



Panduan Pengguna

CloudWatch Log Amazon



CloudWatch Log Amazon: Panduan Pengguna

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

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

Table of Contents

Apa itu Amazon CloudWatch Logs?	1
Fitur	1
AWS Layanan terkait	3
Harga	4
Konsep	4
Penagihan dan biaya	6
Kelas log	7
Fitur yang didukung	7
Memulai	10
Prasyarat	10
Mendaftar untuk Akun AWS	10
Buat pengguna dengan akses administratif	11
Siapkan Antarmuka Baris Perintah	12
Menggunakan agen terpadu CloudWatch	13
Menggunakan CloudWatch agen sebelumnya	13
CloudWatch Prasyarat agen log	14
Quick Start: Menginstal agen di instans EC2 Linux yang sedang berjalan	14
Quick Start: Menginstal agen di instans Linux EC2 saat peluncuran	22
Mulai Cepat: Gunakan CloudWatch Log dengan instans Windows Server 2016	25
Mulai Cepat: Gunakan CloudWatch Log dengan instans Windows Server 2012 dan Windows Server 2008	37
Laporkan status agen CloudWatch Log	47
Mulai agen CloudWatch Log	48
Hentikan agen CloudWatch Log	49
CloudWatch Referensi agen log	49
Mulai Cepat dengan CloudFormation	61
Log konsumsi melalui titik akhir HTTP	63
Perilaku umum	63
Otentikasi token pembawa	64
Opsi 1: Mulai cepat menggunakan AWS konsol	65
Opsi 2: Pengaturan manual	67
Izin kontrol untuk membuat dan menggunakan kunci API CloudWatch Log	70
Memutar kunci API	72
Menanggapi kunci API yang disusupi	73

Praktik keamanan terbaik untuk kunci API	74
Pencatatan penggunaan kunci API dengan CloudTrail	75
Titik akhir OTLP	76
Format permintaan	76
Contoh permintaan	76
Respons	78
Perilaku khusus OTLP	78
Titik akhir HLC	78
Mode masukan	78
Bidang acara (wajib)	79
Bidang waktu (opsional)	79
Content-Type	80
Jenis nilai JSON yang diterima	80
Format titik akhir	80
Format permintaan	81
Contoh permintaan	82
Praktik terbaik	83
Batasan	83
Titik akhir NDJSON	84
Format permintaan	84
Jenis nilai JSON yang diterima	84
Bidang stempel waktu	85
Baris tidak valid	86
Contoh permintaan	86
Respons	86
Praktik terbaik	87
Batasan	88
Titik akhir JSON terstruktur	88
Format permintaan	88
Jenis nilai JSON yang diterima	89
Bidang stempel waktu	89
Contoh permintaan	90
Respons	90
Praktik terbaik	91
Batasan	91
Perbandingan titik akhir konsumsi HTTP	92

Memilih titik akhir	93
Bekerja dengan AWS SDKs	94
Manajemen log	96
Penemuan dan manajemen sumber data	96
Apa itu Sumber Data CloudWatch Log?	97
Cara memulai	97
Bidang sistem	99
Mengakses Sumber Data	99
Hubungan dengan grup log	100
Fitur diaktifkan oleh sumber data	100
Sumber data yang didukung	101
Didukung Layanan AWS untuk sumber data	101
Sumber pihak ketiga yang didukung untuk sumber data	106
Sumber kustom	114
Menganalisis data log dengan Wawasan CloudWatch Log	116
Bahasa kueri yang didukung	119
CloudWatch Bahasa kueri Wawasan Log (Wawasan Log QL)	119
OpenSearch Bahasa Pemrosesan Pipa (PPL)	190
OpenSearch Bahasa Kueri Terstruktur (SQL)	203
Gunakan bahasa alami untuk menghasilkan dan memperbarui kueri Wawasan CloudWatch Log	216
Kueri contoh	217
Memilih untuk tidak menggunakan data Anda untuk perbaikan layanan	219
Log yang didukung dan bidang yang ditemukan	220
Bidang di log JSON	222
Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian	224
Sintaks indeks bidang dan kuota	228
Membuat kebijakan indeks bidang tingkat akun	233
Membuat kebijakan indeks bidang log-group	234
Opsi pemilihan grup log saat membuat kueri	235
Efek menghapus kebijakan indeks bidang	236
Gunakan aspek untuk mengelompokkan dan menjelajahi log	236
Untuk menjalankan kueri berbasis facet	237
Untuk menyimpan kueri berbasis aspek	238
Untuk membuat aspek tingkat akun	238
Manajemen Facet menggunakan APIs	238

Melihat log sekitarnya di Wawasan CloudWatch Log	239
Cara kerja log di sekitarnya	239
Lihat log di sekitarnya	240
Cari di dalam log sekitarnya	241
Analisis pola	242
Memulai dengan analisis pola	243
Detail tentang perintah pola	245
Simpan dan jalankan kembali kueri	246
Menggunakan kueri yang disimpan dengan parameter	252
Tambahkan kueri ke dasbor atau ekspor hasil kueri	254
Lihat kueri atau riwayat kueri yang sedang berjalan	255
Enkripsi hasil kueri dengan AWS Key Management Service	256
Batas	256
Langkah 1: Buat AWS KMS key	256
Langkah 2: Tetapkan izin pada tombol KMS	257
Langkah 3: Kaitkan kunci KMS dengan hasil kueri Anda	259
Langkah 4: Lepaskan kunci dari hasil kueri di akun	259
Hasilkan ringkasan bahasa alami dari hasil kueri CloudWatch Log Insights	259
Cara kerjanya	260
Ketersediaan regional dan pemrosesan data	260
Memulai	261
Izin	261
Privasi data	261
Mengotomatiskan analisis log dengan kueri terjadwal	262
Memahami konsep kueri terjadwal	262
Pemisahan peran IAM	262
Penggunaan lintas wilayah dan lintas akun	265
Menjadwalkan ekspresi dan penanganan zona waktu	266
Memilih bahasa kueri	267
Pemilihan tujuan dan kasus penggunaan	268
Format dan struktur hasil kueri	269
Referensi ekspresi jadwal	270
Praktik terbaik	274
Memulai dengan kueri terjadwal	275
Membuat kueri terjadwal	276
Melihat dan mengelola kueri terjadwal	279

Melihat riwayat eksekusi kueri terjadwal	281
Memperbarui kueri terjadwal	282
Mengkonfigurasi tujuan S3 untuk kueri terjadwal	284
Memberikan hasil ke bucket Amazon S3 di akun yang sama	284
Memberikan hasil ke bucket Amazon S3 di akun lain	285
Mengenkripsi hasil dengan kunci yang dikelola pelanggan AWS KMS	287
Memecahkan masalah kueri terjadwal	289
Eksekusi kueri gagal dengan kesalahan izin	289
Waktu kueri habis	289
Pemrosesan tujuan gagal	290
Kesalahan kueri tidak valid	291
Kesalahan Konkurensi Kueri	291
Deteksi anomali log	292
Tingkat keparahan dan prioritas anomali dan pola	293
Waktu visibilitas anomali	294
Menekan anomali	294
Pertanyaan umum	294
Menggunakan deteksi anomali di Wawasan Log CloudWatch	295
Aktifkan deteksi anomali pada grup log	296
Lihat anomali yang telah ditemukan	297
Buat alarm pada detektor anomali log	300
Metrik yang diterbitkan oleh detektor anomali log	302
Enkripsi detektor anomali dan hasilnya dengan AWS KMS	303
Batas	303
Memecahkan Masalah dengan CloudWatch Log Live Tail	308
Mulai sesi Live Tail menggunakan AWS CLI	309
hanya cetak	309
interaktif	309
Memulai sesi Live Tail di konsol	311
Sentralisasi log lintas wilayah lintas akun	315
Konsep sentralisasi data	315
Menyiapkan sentralisasi log	317
Prasyarat	317
Menyesuaikan nama grup log tujuan	317
Membuat aturan sentralisasi	319
Memodifikasi aturan sentralisasi	322

Melihat aturan sentralisasi	322
Menghapus aturan sentralisasi	323
Pemantauan dan pemecahan masalah aturan sentralisasi	323
Status kesehatan aturan sentralisasi	323
Memantau panggilan API sentralisasi dengan AWS CloudTrail	324
Rekomendasi pemantauan	324
Bekerja dengan grup log dan aliran log	327
Membuat grup log	327
Mengirim log ke grup log	328
Melihat data log	328
Cari data log menggunakan pola filter	329
Cari entri log menggunakan konsol	329
Cari entri log menggunakan AWS CLI	330
Pivot dari metrik ke log	330
Pemecahan masalah	331
Mengubah retensi data log	331
Lindungi grup log dari penghapusan	332
Lindungi grup log dari penghapusan	333
Mengaktifkan perlindungan penghapusan	333
Menandai grup log	334
Dasar-dasar tag	334
Melacak biaya menggunakan penandaan	335
Pembatasan tag	335
Menandai grup log menggunakan AWS CLI	336
Menandai grup log menggunakan API CloudWatch Log	336
Enkripsi data log menggunakan AWS KMS	337
Batas	338
Langkah 1: Buat AWS KMS kunci	256
Langkah 2: Tetapkan izin pada tombol KMS	339
Langkah 3: Tetapkan izin pada prinsipal IAM panggilan	342
Langkah 4: Kaitkan kunci KMS dengan grup log	307
Langkah 5: Pisahkan kunci dari grup log	307
Kunci KMS dan konteks enkripsi	345
Membantu melindungi data log sensitif dengan masking	348
Memahami kebijakan perlindungan data	352
Izin IAM diperlukan untuk membuat atau bekerja dengan kebijakan perlindungan data	354

Buat kebijakan perlindungan data seluruh akun	359
Membuat kebijakan perlindungan data untuk satu grup log	363
Lihat data yang dibuka kedoknya	366
Laporan temuan audit	366
Jenis data yang dapat Anda lindungi	368
Ubah log selama konsumsi	412
Buat dan kelola trafo log	414
Buat kebijakan transformator tingkat akun	415
Mengedit atau menghapus kebijakan transformator tingkat akun	416
Buat transformator log-group-level log dari awal	417
Buat log-group-level transformator dengan menyalin yang sudah ada	418
Edit log-group-level transformator	419
Hapus log-group-level transformator	419
Prosesor tipe parser yang dapat dikonfigurasi	420
parseJSON	421
grok	422
csv	453
parseKeyValue	456
Prosesor bawaan untuk AWS log penjual otomatis	458
ParseWAF	458
ParsePostgres	461
ParseCloudFront	462
ParseRoute53	463
ParseVPC	464
ParseTooCSF	466
Prosesor mutasi string	467
lowerCaseString	467
upperCaseString	468
SplitString	470
SubstituteString	472
TrimString	476
JSON bermutasi prosesor	477
AddKeys	477
DeleteKeys	479
MoveKeys	480
RenameKeys	482

CopyValue	483
listToMap	486
Prosesor konverter tipe data	491
TypeConverter	491
DateTimeConverter	493
Metrik dan kesalahan transformasi	495
Analisis dengan Amazon OpenSearch Service	496
Langkah 1: Buat integrasi dengan OpenSearch Layanan	498
Izin yang diperlukan	498
Buat integrasi	506
Langkah 2: Buat dasbor log vendid	507
Melihat, mengedit, atau menghapus dasbor log terjual	508
Lihat dasbor log yang dijual di CloudWatch Log atau Layanan OpenSearch	508
Berikan akses tampilan dasbor ke peran IAM tambahan atau pengguna IAM	509
Edit konfigurasi dasbor	509
Hapus dasbor log penjual	509
Hapus semua integrasi dasbor log penjual otomatis dengan Layanan OpenSearch	510
Kebijakan IAM untuk pengguna	511
Izin yang dibutuhkan integrasi	513
Akses log dengan Integrasi Tabel S3	515
Memahami Integrasi Tabel S3	515
Komponen Inti	515
Aliran data ke tabel S3	515
Kapan Menggunakan Integrasi Tabel S3	516
Prasyarat	516
Izin IAM	517
Kebijakan kunci KMS (untuk data terenkripsi)	518
Memulai	519
Mengintegrasikan Tabel Amazon S3 Amazon AWS dengan layanan analitik - Menggunakan konsol Amazon S3 (Tautan)	521
Konfigurasi Izin Lake Formation	521
Connect dengan Analytics Tools	522
Filter metrik	523
Konsep	524
Filter sintaks pola untuk filter metrik	525
Mengkonfigurasi nilai metrik untuk filter metrik	526

Menerbitkan dimensi dengan metrik dari peristiwa log	527
Menggunakan nilai dalam peristiwa log untuk menambah nilai metrik	531
Membuat filter metrik	532
Membuat filter metrik untuk grup log	532
Contoh: Hitung peristiwa log	533
Contoh: Hitung kemunculan suatu istilah	535
Contoh: Hitung kode HTTP 404	536
Contoh: Hitung kode HTTP 4xx	539
Contoh: Mengekstraksi bidang dari log Apache dan menetapkan dimensi	540
Daftar filter metrik	542
Menghapus filter metrik	543
Filter langganan	545
Konsep	547
Filter langganan tingkat grup log	548
Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams	549
Contoh 2: Filter berlangganan dengan AWS Lambda	555
Contoh 3: Filter berlangganan dengan Amazon Data Firehose	558
Contoh 4: Filter berlangganan dengan OpenSearch Layanan Amazon	566
Filter berlangganan tingkat akun	566
Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams	567
Contoh 2: Filter berlangganan dengan AWS Lambda	574
Contoh 3: Filter berlangganan dengan Amazon Data Firehose	578
Langganan lintas akun Lintas wilayah	586
Berbagi data log lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams	587
Berbagi data log lintas wilayah lintas akun menggunakan Firehose	607
Langganan tingkat akun lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams	621
Langganan tingkat akun lintas wilayah lintas akun menggunakan Firehose	640
Pencegahan Deputi Bingung	652
Pencegahan rekursi log	653
Filter sintaks pola	655
Ekspresi reguler yang didukung	655
Mencocokkan istilah menggunakan ekspresi reguler	658
Ketentuan kecocokan dalam peristiwa log tidak terstruktur	659
Ketentuan kecocokan dalam acara log JSON	663
Ketentuan kecocokan dalam peristiwa log yang dibatasi ruang	671

Aktifkan logging dari AWS layanan	676
Logging yang membutuhkan izin tambahan [V1]	678
Log dikirim ke CloudWatch Log	679
Log yang dikirim ke Amazon S3	682
Log dikirim ke Firehose	687
Logging yang membutuhkan izin tambahan [V2]	689
Log dikirim ke CloudWatch Log	690
Log yang dikirim ke Amazon S3	694
Log dikirim ke Firehose	698
Jejak dikirim ke X-Ray	701
Izin khusus layanan	705
Izin khusus konsol	706
Contoh pengiriman lintas akun	707
Buat sumber pengiriman	708
Konfigurasi pengiriman ke bucket Amazon S3	708
Konfigurasi pengiriman ke aliran Firehose	711
Pencegahan "confused deputy" lintas layanan	714
Otomatiskan pengaktifan log dengan aturan pengaktifan telemetri	715
Hirarki evaluasi aturan	716
Membuat aturan pemberdayaan	716
Aktifkan logging dari sumber pihak ketiga	717
Integrasi pihak ketiga langsung	717
Sumber tambahan melalui AWS Security Hub CSPM	717
Mengekspor data log ke Amazon S3	723
Konsep	724
Ekspor data log ke Amazon S3 menggunakan konsol	725
Ekspor akun yang sama (konsol)	725
Ekspor lintas akun (konsol)	732
Ekspor data log ke Amazon S3 menggunakan AWS CLI	742
Ekspor akun yang sama (CLI)	742
Ekspor lintas akun (CLI)	749
Jelaskan tugas ekspor (CLI)	759
Batalkan tugas ekspor (CLI)	761
Streaming data ke OpenSearch Layanan	762
Prasyarat	762
Berlangganan grup log ke OpenSearch Layanan	763

Contoh kode	765
Hal-hal mendasar	766
Tindakan	767
Skenario	833
Konfigurasikan Amazon ECS Service Connect	833
Membuat fungsi Lambda pertama Anda	850
Jalankan kueri besar	862
Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda	899
Enkripsi tabel pencarian menggunakan AWS KMS	902
Bagaimana CloudWatch Log menggunakan AWS KMS untuk tabel pencarian	902
Izin yang diperlukan	903
Langkah 1: Buat AWS KMS kunci	904
Langkah 2: Tetapkan izin pada tombol KMS	904
Langkah 3: Kaitkan kunci KMS dengan tabel pencarian	906
Pertimbangan-pertimbangan	907
Keamanan	908
Perlindungan data	909
Enkripsi saat diam	910
Enkripsi saat bergerak	910
Manajemen identitas dan akses	910
Autentikasi	911
Kontrol akses	911
Ikhtisar mengenai pengelolaan akses	912
Menggunakan kebijakan berbasis identitas (kebijakan IAM)	918
Validasi kepatuhan	945
Ketahanan	945
Keamanan infrastruktur	946
Titik akhir VPC antarmuka	946
Ketersediaan	947
Membuat titik akhir VPC untuk Log CloudWatch	947
Menguji koneksi antara VPC dan Log CloudWatch	947
Mengontrol akses ke titik akhir VPC CloudWatch Log	948
Support untuk kunci konteks VPC	949
Logging API dan operasi konsol dengan AWS CloudTrail	950
Informasi pembuatan kueri di CloudTrail	955
Memahami entri file log	956

Memantau penggunaan dengan CloudWatch metrik	958
CloudWatch Metrik log	958
Dimensi untuk metrik CloudWatch Log	963
Metrik dan dimensi transformator log	964
Metrik dan dimensi sentralisasi	965
CloudWatch Metrik penggunaan layanan log	968
Kuota layanan	971
Mengelola kuota layanan CloudWatch Log Anda	983
Riwayat dokumen	985
AWS Glosarium	1002
.....	miii

Apa itu Amazon CloudWatch Logs?

Anda dapat menggunakan Amazon CloudWatch Logs untuk memantau, menyimpan, dan mengakses file log Anda dari instans Amazon Elastic Compute Cloud (Amazon EC2), AWS CloudTrail Route 53, dan sumber lainnya.

CloudWatch Log memungkinkan Anda untuk memusatkan log dari semua sistem, aplikasi, dan AWS layanan yang Anda gunakan, dalam satu layanan yang sangat skalabel. Anda kemudian dapat dengan mudah melihatnya, mencari kode atau pola kesalahan tertentu, memfilternya berdasarkan bidang tertentu, atau mengarsipkannya dengan aman untuk analisis masa depan. CloudWatch Log memungkinkan Anda untuk melihat semua log Anda, terlepas dari sumbernya, sebagai aliran peristiwa tunggal dan konsisten yang diurutkan berdasarkan waktu.

CloudWatch Log juga mendukung kueri log Anda dengan bahasa kueri yang kuat, mengaudit dan menutupi data sensitif di log, dan menghasilkan metrik dari log menggunakan filter atau format log yang disematkan.

CloudWatch Log mendukung dua kelas log. Grup log di kelas CloudWatch log Standar Log mendukung semua fitur CloudWatch Log. Grup log di kelas CloudWatch log Akses Jarang Log dikenakan biaya konsumsi yang lebih rendah dan mendukung subset dari kemampuan kelas Standar. Untuk informasi selengkapnya, lihat [Kelas log](#).

Fitur

- Dua kelas log untuk fleksibilitas — CloudWatch Log menawarkan dua kelas log sehingga Anda dapat memiliki opsi hemat biaya untuk log yang jarang Anda akses. Anda juga memiliki opsi fitur lengkap untuk log yang memerlukan pemantauan waktu nyata atau fitur lainnya. Untuk informasi selengkapnya, lihat [Kelas log](#).
- Kueri data log Anda — Anda dapat menggunakan Wawasan CloudWatch Log untuk mencari dan menganalisis data log Anda secara interaktif. Anda dapat melakukan kueri untuk membantu Anda merespons masalah operasional secara lebih efisien dan efektif. CloudWatch Logs Insights mencakup bahasa kueri yang dibuat khusus dengan beberapa perintah sederhana namun kuat. Kami menyediakan kueri contoh, deskripsi perintah, penyelesaian otomatis kueri, dan penemuan bidang log untuk membantu Anda memulai. Contoh kueri disertakan untuk beberapa jenis log AWS layanan. Untuk memulai, lihat [Menganalisis data log dengan Wawasan CloudWatch Log](#).
- Buat indeks bidang untuk membuat kueri lebih efisien — Anda dapat membuat indeks bidang bidang dalam peristiwa log Anda. Saat Anda kemudian menggunakan indeks bidang dalam

kueri Wawasan CloudWatch Log, kueri akan mencoba melewati pemrosesan peristiwa log yang diketahui tidak menyertakan bidang yang diindeks. Kueri ini mengurangi volume pemindaian kueri Anda, sehingga memungkinkan untuk mengembalikan hasil lebih cepat. Untuk memulai, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

- Deteksi dan debug menggunakan Live Tail — Anda dapat menggunakan Live Tail untuk memecahkan masalah insiden dengan cepat dengan melihat daftar streaming peristiwa log baru saat tertelan. Anda dapat melihat, memfilter, dan menyorot log yang dicerna dalam waktu dekat, membantu Anda mendeteksi dan menyelesaikan masalah dengan cepat. Anda dapat memfilter log berdasarkan istilah yang Anda tentukan, dan juga menyorot log yang berisi istilah tertentu untuk membantu Anda menemukan apa yang Anda cari dengan cepat. Untuk informasi selengkapnya, lihat [Memecahkan Masalah dengan CloudWatch Log Live Tail](#).
- Memantau log dari EC2 instans Amazon — Anda dapat menggunakan CloudWatch Log untuk memantau aplikasi dan sistem menggunakan data log. Misalnya, CloudWatch Log dapat melacak jumlah kesalahan yang terjadi di log aplikasi Anda dan mengirimkan Anda pemberitahuan setiap kali tingkat kesalahan melebihi ambang batas yang Anda tentukan. CloudWatch Log menggunakan data log Anda untuk pemantauan; jadi, tidak ada perubahan kode yang diperlukan. Misalnya, Anda dapat memantau log aplikasi untuk istilah literal tertentu (seperti "NullPointerException") atau menghitung jumlah kemunculan istilah literal pada posisi tertentu dalam data log (seperti kode status "404" dalam log akses Apache). Ketika istilah yang Anda cari ditemukan, CloudWatch Log melaporkan data ke CloudWatch metrik yang Anda tentukan. Data log dienkripsi saat transit dan saat diam. Untuk memulai, lihat [Memulai dengan CloudWatch Log](#).
- Memantau peristiwa yang AWS CloudTrail dicatat - Anda dapat membuat alarm CloudWatch dan menerima pemberitahuan aktivitas API tertentu seperti yang ditangkap oleh CloudTrail dan menggunakan notifikasi untuk melakukan pemecahan masalah. Untuk memulai, lihat [Mengirim CloudTrail Acara ke CloudWatch Log](#) di Panduan AWS CloudTrail Pengguna.
- Audit dan tutupi data sensitif — Jika Anda memiliki data sensitif di log, Anda dapat membantu melindunginya dengan kebijakan perlindungan data. Kebijakan ini memungkinkan Anda mengaudit dan menutupi data sensitif. Jika Anda mengaktifkan perlindungan data, maka secara default, data sensitif yang cocok dengan pengidentifikasi data yang Anda pilih ditutupi. Untuk informasi selengkapnya, lihat [Membantu melindungi data log sensitif dengan masking](#).
- Retensi log — Secara default, log disimpan tanpa batas waktu dan tidak pernah kedaluwarsa. Anda dapat menyesuaikan kebijakan retensi untuk setiap grup log, menjaga retensi yang tak terbatas, atau memilih periode retensi antara 10 tahun dan satu hari.
- Perlindungan penghapusan — Perlindungan yang mencegah penghapusan grup log dan aliran lognya secara tidak sengaja. Saat diaktifkan pada grup log, perlindungan penghapusan memblokir

semua operasi penghapusan hingga dinonaktifkan secara eksplisit. Secara default, perlindungan penghapusan tidak diaktifkan. Fitur opsional ini membantu melindungi data operasional dan kepatuhan yang penting dari penghapusan yang tidak diinginkan, seperti grup log yang berisi data audit, dan log aplikasi produksi untuk pemecahan masalah dan analisis.

- **Arsipkan data log** - Anda dapat menggunakan CloudWatch Log untuk menyimpan data log Anda dalam penyimpanan yang sangat tahan lama. Agen CloudWatch Logs memudahkan untuk dengan cepat mengirim data log yang diputar dan tidak diputar dari host dan ke layanan log. Anda kemudian dapat mengakses data log mentah ketika dibutuhkannya.
- **Kueri DNS Route 53 Log** — Anda dapat menggunakan CloudWatch Log untuk mencatat informasi tentang kueri DNS yang diterima Route 53. Untuk informasi selengkapnya, lihat [Mencatat Log Kueri DNS](#) dalam Panduan Developer Amazon Route 53.
- **Memusatkan log di seluruh akun dan wilayah** — Anda dapat menggunakan Sentralisasi CloudWatch Log untuk menentukan aturan sentralisasi lintas akun dan lintas wilayah yang mereplikasi data log yang tertelan di lingkungan multi-akun dan wilayah ke wilayah dan akun pusat. Anda dapat mengonfigurasi wilayah cadangan dalam akun pusat untuk meningkatkan ketahanan, menentukan pengaturan enkripsi untuk grup log yang baru dibuat di akun pusat, dan membuat filter metrik dan langganan pada peristiwa log dari akun sumber dan wilayah tertentu dengan kemampuan penyaringan yang ditingkatkan.

AWS Layanan terkait

Layanan berikut digunakan bersama dengan CloudWatch Log:

- **AWS CloudTrail** adalah layanan web yang memungkinkan Anda memantau panggilan yang dilakukan ke CloudWatch Logs API untuk akun Anda, termasuk panggilan yang dilakukan oleh Konsol Manajemen AWS, AWS Command Line Interface (AWS CLI), dan layanan lainnya. Saat CloudTrail logging diaktifkan, CloudTrail menangkap panggilan API di akun Anda dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Setiap berkas log dapat berisi satu atau lebih catatan, tergantung pada berapa banyak tindakan yang harus dilakukan untuk memenuhi permintaan. Untuk informasi lebih lanjut tentang AWS CloudTrail, lihat [Apa itu AWS CloudTrail?](#) dalam AWS CloudTrail User Guide. Untuk contoh jenis data yang CloudWatch menulis ke dalam file CloudTrail log, lihat [Logging CloudWatch Logs API dan operasi konsol di AWS CloudTrail](#).
- **AWS Identity and Access Management (IAM)** adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman bagi pengguna Anda. Gunakan IAM untuk mengendalikan orang yang dapat menggunakan sumber daya AWS Anda (otentikasi) dan sumber

daya apa yang dapat digunakan dengan cara apa (otorisasi). Untuk informasi selengkapnya, lihat [Apa yang dimaksud dengan IAM?](#) dalam Panduan Pengguna IAM.

- Amazon Kinesis Data Streams adalah layanan web yang dapat Anda gunakan untuk pengambilan dan agregasi data yang cepat dan berkesinambungan. Jenis data yang digunakan meliputi data log infrastruktur IT, log aplikasi, media sosial, umpan data pasar, dan data clickstream web. Karena waktu respons untuk pengambilan dan pengolahan data secara waktu nyata, pemrosesannya biasanya ringan. Untuk informasi selengkapnya, lihat [Apa yang dimaksud dengan Amazon Kinesis Data Streams?](#) dalam Panduan Developer Amazon Kinesis Data Streams.
- AWS Lambda adalah layanan web yang dapat Anda gunakan untuk membangun aplikasi yang merespons informasi baru dengan cepat. Unggah kode aplikasi Anda sebagai fungsi Lambda dan Lambda menjalankan kode Anda di infrastruktur komputasi dengan ketersediaan tinggi dan melakukan semua administrasi sumber daya komputasi, termasuk pemeliharaan server dan sistem operasi, penyediaan kapasitas dan penskalaan otomatis, deployment patch keamanan, dan pemantauan dan pencatatan kode. Yang perlu Anda lakukan adalah menyediakan kode di salah satu bahasa yang didukung Lambda. Untuk informasi lebih lanjut, lihat [Apa itu AWS Lambda?](#) di Panduan AWS Lambda Pengembang.

Harga

Saat Anda mendaftar AWS, Anda dapat memulai dengan CloudWatch Log secara gratis menggunakan [Tingkat AWS Gratis](#).

Tarif standar berlaku untuk log yang disimpan oleh layanan lain menggunakan CloudWatch Log (misalnya, log aliran VPC Amazon dan log Lambda).

Untuk informasi selengkapnya tentang harga, lihat [CloudWatch Harga Amazon](#).

Untuk informasi selengkapnya tentang cara menganalisis biaya dan penggunaan untuk CloudWatch Log dan CloudWatch, dan untuk praktik terbaik tentang cara mengurangi biaya, lihat [CloudWatch penagihan dan biaya](#).

Konsep Amazon CloudWatch Log

Terminologi dan konsep yang menjadi pusat pemahaman dan penggunaan CloudWatch Log Anda dijelaskan di bawah ini.

Kelas log

CloudWatch Log menawarkan dua kelas grup log. Kelas log Standar adalah opsi berfitur lengkap untuk log yang memerlukan pemantauan waktu nyata atau log yang sering Anda akses. Kelas log Akses Jarang adalah opsi berbiaya lebih rendah untuk log yang Anda akses lebih jarang. Ini mendukung subset dari kemampuan kelas log Standar.

Log acara

Log acara adalah catatan dari beberapa aktivitas yang direkam oleh aplikasi atau sumber daya yang dipantau. Catatan peristiwa CloudWatch log yang dipahami Log berisi dua properti: stempel waktu kapan peristiwa terjadi, dan pesan peristiwa mentah. Pesan kejadian harus dikodekan dalam UTF-8.

Pengaliran Log

Pengaliran log adalah urutan log acara yang berbagi sumber yang sama. Lebih khusus lagi, pengaliran log umumnya dimaksudkan untuk mewakili urutan kejadian yang berasal dari instans aplikasi atau sumber daya yang dipantau. Sebagai contoh, pengaliran log dapat dikaitkan dengan log akses Apache pada host tertentu. Saat Anda tidak lagi membutuhkan aliran log, Anda dapat menghapusnya menggunakan delete-log-stream perintah [aws logs](#).

Grup log

Grup log menentukan grup pengaliran log yang berbagi pengaturan kontrol retensi, pemantauan, dan akses yang sama. Setiap pengaliran log harus termasuk dalam satu grup log. Misalnya, jika Anda memiliki pengaliran log terpisah untuk log akses Apache dari setiap host, Anda dapat mengelompokkan pengaliran log tersebut ke dalam grup log tunggal yang disebut `MyWebsite.com/Apache/access_log`.

Tidak ada batas jumlah pengaliran log yang dapat tergabung dalam satu grup log.

Filter metrik

Anda dapat menggunakan filter metrik untuk mengekstrak pengamatan metrik dari peristiwa yang dicerna dan mengubahnya menjadi titik data dalam CloudWatch metrik. Filter metrik ditetapkan untuk grup log, dan semua filter yang ditetapkan ke grup log diterapkan ke pengaliran log mereka.

Pengaturan retensi

Pengaturan retensi dapat digunakan untuk menentukan berapa lama peristiwa log disimpan di CloudWatch Log. Log acara yang kedaluwarsa dapat dihapus secara otomatis. Sama seperti filter metrik, pengaturan retensi juga ditetapkan ke grup log, dan retensi yang ditetapkan ke grup log diterapkan ke pengaliran log mereka.

Perlindungan penghapusan

Perlindungan penghapusan adalah perlindungan yang mencegah penghapusan grup log dan aliran lognya secara tidak sengaja. Saat diaktifkan pada grup log, perlindungan penghapusan memblokir semua operasi penghapusan hingga dinonaktifkan secara eksplisit. Secara default, perlindungan penghapusan tidak diaktifkan. Fitur opsional ini membantu melindungi data operasional dan kepatuhan yang penting dari penghapusan yang tidak diinginkan, seperti grup log yang berisi data audit, dan log aplikasi produksi untuk pemecahan masalah dan analisis.

Amazon CloudWatch Logs penagihan dan biaya

Untuk informasi terperinci tentang cara menganalisis biaya dan penggunaan untuk CloudWatch Log dan CloudWatch, dan untuk praktik terbaik tentang cara mengurangi biaya, lihat [CloudWatch penagihan dan biaya](#).

Untuk informasi selengkapnya tentang harga, lihat [CloudWatch Harga Amazon](#).

Saat Anda mendaftar AWS, Anda dapat memulai dengan CloudWatch Log secara gratis menggunakan [Tingkat AWS Gratis](#).

Tarif standar berlaku untuk log yang disimpan oleh layanan lain menggunakan CloudWatch Log (misalnya, log aliran VPC Amazon dan log Lambda).

Kelas log

CloudWatch Log menawarkan dua kelas grup log:

- Kelas CloudWatch log Standar Log adalah opsi berfitur lengkap untuk log yang memerlukan pemantauan waktu nyata atau log yang sering Anda akses.
- Class CloudWatch log Logs Infrequent Access adalah kelas log baru yang dapat Anda gunakan untuk mengkonsolidasikan log Anda secara hemat biaya. Kelas log ini menawarkan subset kemampuan CloudWatch Log termasuk konsumsi terkelola, penyimpanan, analisis log lintas akun, dan enkripsi dengan harga konsumsi per GB yang lebih rendah. Kelas log Akses Jarang sangat ideal untuk kueri ad-hoc dan analisis after-the-fact forensik pada log yang jarang diakses.

Note

Untuk biaya, kelas log Akses Standar dan Jarang hanya berbeda dalam biaya konsumsi. Biaya penyimpanan dan biaya Wawasan CloudWatch Log sama di setiap kelas log.

Untuk informasi selengkapnya tentang harga CloudWatch Log, lihat [CloudWatch Harga Amazon](#).

Important

Setelah grup log dibuat, kelas lognya tidak dapat diubah.

Fitur yang didukung

Tabel berikut mencantumkan fitur untuk setiap kelas log.

Fitur	Standar	Akses Jarang
Tertelan dan penyimpanan log yang dikelola sepenuhnya	Ya ✓	Ya ✓
Fitur lintas akun	Ya ✓	Ya ✓
Enkripsi dengan AWS KMS	Ya ✓	Ya ✓

Fitur	Standar	Akses Jarang
CloudWatch Perintah kueri Log Insights	Ya ✓	Ya ✓ (Kebanyakan perintah — lihat Log Insights Perintah QL didukung di kelas log.)
CloudWatch Logs Insights menemukan bidang	Ya ✓	Ya ✓
Segi	Ya ✓	Tidak
Menggunakan CloudWatch Pipeline untuk mengubah log	Ya ✓	Ya ✓
Ekspor ke Amazon S3	Ya ✓	Ya ✓
Integrasi Tabel S3	Ya ✓	Ya ✓
Pertanyaan Terjadwal	Ya ✓	Ya ✓
Menggunakan OpenSearch PPL atau OpenSearch SQL untuk melakukan kueri di Wawasan CloudWatch Log;	Ya ✓	Ya ✓
Bantuan kueri bahasa alami	Ya ✓	Tidak
CloudWatch Deteksi Anomali Log	Ya ✓	Tidak
Ekor Hidup	Ya ✓	Tidak
Pengindeksan lapangan	Ya ✓	Tidak
Bandingkan dengan rentang waktu sebelumnya	Ya ✓	Tidak
Filter berlangganan	Ya ✓	Tidak

Fitur	Standar	Akses Jarang	
GetLogEvents dan operasi FilterLogEvents API	Ya ✓	Tidak didukung. Gunakan Wawasan CloudWatch Log untuk melihat peristiwa log yang disimpan dalam grup log di kelas log Akses Jarang.	
Filter metrik	Ya ✓	Tidak	
Kontainer Insights log ingestion	Ya ✓	Tidak	
Konsumsi log Lambda Insights	Ya ✓	Tidak	
Perlindungan data sensitif dengan masking	Ya ✓	Ya ✓	
Format metrik tertanam	Ya ✓	Tidak	

Note

Selain dua kelas log ini, ada kelas Delivery log. Gunakan kelas Delivery log hanya untuk mengirimkan AWS Lambda log untuk disimpan di Amazon S3 atau Amazon Data Firehose. Peristiwa log di grup log di kelas Pengiriman disimpan CloudWatch Logs selama dua hari. Periode retensi ini tetap dan tidak dapat diubah. Kelas log ini tidak menawarkan CloudWatch Logs kemampuan kaya seperti kueri CloudWatch Logs Wawasan.

Memulai dengan CloudWatch Log

Untuk mengumpulkan log dari instans Amazon EC2 dan server lokal ke dalam CloudWatch Log, gunakan agen terpadu. CloudWatch Ini memungkinkan Anda untuk mengumpulkan log dan metrik lanjutan dengan satu agen. Ini menawarkan dukungan di seluruh sistem operasi, termasuk server yang menjalankan Windows Server. Agen ini juga memberikan performa yang lebih baik.

Jika Anda menggunakan CloudWatch agen terpadu untuk mengumpulkan CloudWatch metrik, ini memungkinkan pengumpulan metrik sistem tambahan, untuk visibilitas tamu. Ini juga mendukung pengumpulan metrik khusus menggunakan StatsD atau collectd.

Untuk informasi selengkapnya, lihat [Menginstal CloudWatch Agen](#) di Panduan CloudWatch Pengguna Amazon.

Agen CloudWatch Logs lama, yang hanya mendukung kumpulan log dari server yang menjalankan Linux, tidak digunakan lagi dan tidak lagi didukung. Untuk informasi tentang migrasi dari agen CloudWatch Log lama ke agen terpadu, lihat [Membuat file konfigurasi CloudWatch agen dengan wizard](#).

Daftar Isi

- [Prasyarat](#)
- [Gunakan CloudWatch agen terpadu untuk memulai dengan CloudWatch Log](#)
- [Gunakan CloudWatch agen sebelumnya untuk memulai dengan CloudWatch Log](#)
- [Mulai Cepat: Gunakan CloudFormation untuk memulai dengan CloudWatch Log](#)

Prasyarat

Untuk menggunakan Amazon CloudWatch Logs, Anda memerlukan AWS akun. AWS Akun Anda memungkinkan Anda menggunakan layanan (misalnya, Amazon EC2) untuk menghasilkan log yang dapat Anda lihat di CloudWatch konsol, antarmuka berbasis web. Selain itu, Anda dapat menginstal dan mengkonfigurasi AWS Command Line Interface (AWS CLI).

Mendaftar untuk Akun AWS

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

Untuk mendaftar untuk Akun AWS

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

Bagian dari prosedur pendaftaran melibatkan menerima panggilan telepon atau pesan teks dan memasukkan kode verifikasi pada keypad telepon.

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

AWS mengirim Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk <https://aws.amazon.com/ke/> dan memilih Akun Saya.

Buat pengguna dengan akses administratif

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

Amankan Anda Pengguna root akun AWS

1. Masuk ke [Konsol Manajemen AWS](#) sebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

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

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

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

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

Untuk mendapatkan petunjuk, silakan lihat [Mengaktifkan AWS IAM Identity Center](#) di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

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

Masuk sebagai pengguna dengan akses administratif

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

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

Tetapkan akses ke pengguna tambahan

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

Untuk petunjuknya, lihat [Membuat set izin](#) di Panduan AWS IAM Identity Center Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

Untuk petunjuk, lihat [Menambahkan grup](#) di Panduan AWS IAM Identity Center Pengguna.

Siapkan Antarmuka Baris Perintah

Anda dapat menggunakan AWS CLI untuk melakukan operasi CloudWatch Log.

Untuk informasi tentang cara menginstal dan mengkonfigurasi AWS CLI, lihat [Menyiapkan dengan Antarmuka Baris AWS Perintah](#) di Panduan AWS Command Line Interface Pengguna.

Gunakan CloudWatch agen terpadu untuk memulai dengan CloudWatch Log

Untuk informasi selengkapnya tentang penggunaan CloudWatch agen terpadu untuk memulai CloudWatch Log, lihat [Mengumpulkan Metrik dan Log dari Instans Amazon EC2 dan Server Lokal dengan CloudWatch Agen di Panduan Pengguna Amazon](#). CloudWatch Anda menyelesaikan langkah-langkah yang tercantum dalam bagian ini untuk menginstal, mengonfigurasi, dan memulai agen. Jika Anda tidak menggunakan agen untuk juga mengumpulkan CloudWatch metrik, Anda dapat mengabaikan bagian apa pun yang merujuk ke metrik.

Jika saat ini Anda menggunakan agen CloudWatch Log lama dan ingin bermigrasi menggunakan agen terpadu baru, sebaiknya gunakan wizard yang disertakan dalam paket agen baru. Wizard ini dapat membaca file konfigurasi agen CloudWatch Log Anda saat ini dan mengatur CloudWatch agen untuk mengumpulkan log yang sama. Untuk informasi selengkapnya tentang wizard, lihat [Membuat File Konfigurasi CloudWatch Agen dengan Wizard](#) di Panduan CloudWatch Pengguna Amazon.

Gunakan CloudWatch agen sebelumnya untuk memulai dengan CloudWatch Log

Important

CloudWatch menyertakan CloudWatch agen terpadu yang dapat mengumpulkan log dan metrik dari instans EC2 dan server lokal. Agen logs-only yang lebih lama tidak digunakan lagi dan tidak lagi didukung.

Untuk informasi tentang migrasi dari agen khusus log yang lebih lama ke agen terpadu, lihat [Membuat file konfigurasi CloudWatch agen dengan wizard](#).

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log lama untuk pelanggan yang masih menggunakannya.

Dengan menggunakan agen CloudWatch Log, Anda dapat mempublikasikan data log dari instans Amazon EC2 yang menjalankan Linux atau Windows Server, dan peristiwa yang dicatat dari. AWS CloudTrail Sebaiknya gunakan agen CloudWatch terpadu untuk mempublikasikan data log Anda. Untuk informasi selengkapnya tentang agen baru, lihat [Mengumpulkan Metrik dan Log dari Instans Amazon EC2 dan Server Lokal dengan CloudWatch Agen di Panduan Pengguna Amazon](#). CloudWatch

Daftar Isi

- [CloudWatch Prasyarat agen log](#)
- [Mulai Cepat: Instal dan konfigurasi agen CloudWatch Log pada instans Linux EC2 yang sedang berjalan](#)
- [Mulai Cepat: Instal dan konfigurasi agen CloudWatch Log pada instans Linux EC2 saat diluncurkan](#)
- [Mulai Cepat: Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2016 untuk mengirim log ke Log menggunakan agen CloudWatch Log CloudWatch](#)
- [Mulai Cepat: Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2012 dan Windows Server 2008 untuk mengirim log ke Log CloudWatch](#)
- [Laporkan status agen CloudWatch Log](#)
- [Mulai agen CloudWatch Log](#)
- [Hentikan agen CloudWatch Log](#)
- [CloudWatch Referensi agen log](#)

CloudWatch Prasyarat agen log

Agan CloudWatch Logs memerlukan Python versi 2.7, 3.0, atau 3.3, dan salah satu versi Linux berikut:

- Amazon Linux versi 2014.03.02 atau yang lebih baru. Amazon Linux 2 tidak didukung
- Ubuntu Server versi 12.04, 14.04, atau 16.04
- CentOS versi 6, 6.3, 6.4, 6.5, atau 7.0
- Red Hat Enterprise Linux (RHEL) versi 6.5 atau 7.0
- Debian 8.0

Mulai Cepat: Instal dan konfigurasi agen CloudWatch Log pada instans Linux EC2 yang sedang berjalan

Important

Agan log yang lebih tua tidak digunakan lagi. CloudWatch menyertakan agen terpadu yang dapat mengumpulkan log dan metrik dari instans EC2 dan server lokal. Untuk informasi selengkapnya, lihat [Memulai dengan CloudWatch Log](#).

Untuk informasi tentang migrasi dari agen CloudWatch Log lama ke agen terpadu, lihat [Membuat file konfigurasi CloudWatch agen dengan wizard](#).

Agan log yang lebih lama hanya mendukung Python versi 2.6 hingga 3.5. Selain itu, agen CloudWatch Log yang lebih lama tidak mendukung Layanan Metadata Instans Versi 2 (IMDSv2). Jika server Anda menggunakan IMDSv2, Anda harus menggunakan agen terpadu yang lebih baru, bukan agen CloudWatch Log yang lebih lama.

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log lama untuk pelanggan yang masih menggunakannya.

Tip

CloudWatch menyertakan agen terpadu baru yang dapat mengumpulkan log dan metrik dari instans EC2 dan server lokal. Jika Anda belum menggunakan agen CloudWatch Log yang lebih lama, kami sarankan Anda menggunakan agen terpadu CloudWatch yang lebih baru. Untuk informasi selengkapnya, lihat [Memulai dengan CloudWatch Log](#).

Selain itu, agen lama tidak mendukung Instance Metadata Service Version 2 (IMDSv2). Jika server Anda menggunakan IMDSv2, Anda harus menggunakan agen terpadu yang lebih baru, bukan agen CloudWatch Log yang lebih lama.

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log yang lebih tua.

Konfigurasi agen CloudWatch Log yang lebih lama pada instans Linux EC2 yang sedang berjalan

Anda dapat menggunakan penginstal agen CloudWatch Log pada instans EC2 yang ada untuk menginstal dan mengonfigurasi agen CloudWatch Log. Setelah instalasi selesai, log secara otomatis mengalir dari instans ke pengaliran log yang Anda buat saat menginstal agen. Agen mengonfirmasi bahwa itu telah dimulai dan tetap berjalan sampai Anda menonaktifkannya.

Selain menggunakan agen, Anda juga dapat mempublikasikan data log menggunakan AWS CLI, CloudWatch Logs SDK, atau CloudWatch Logs API. Yang paling cocok untuk menerbitkan data di baris perintah atau melalui skrip. CloudWatch Logs SDK paling cocok untuk menerbitkan data log langsung dari aplikasi atau membuat aplikasi penerbitan log Anda sendiri.

Langkah 1: Konfigurasi peran IAM atau pengguna Anda untuk CloudWatch Log

Agan CloudWatch Log mendukung peran dan pengguna IAM. Jika instans Anda sudah memiliki IAM role yang terkait dengannya, pastikan Anda menyertakan kebijakan IAM di bawah ini. Jika Anda belum memiliki IAM role yang ditetapkan ke instans, Anda dapat menggunakan kredensial IAM untuk langkah berikutnya atau Anda dapat menetapkan IAM role ke instans tersebut. Untuk informasi selengkapnya, lihat [Melampirkan IAM Role ke Instans](#).

Untuk mengonfigurasi peran IAM atau pengguna Anda untuk CloudWatch Log

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Peran.
3. Pilih peran dengan memilih nama peran (jangan mencentang kotak di samping nama).
4. Pilih Attach Policies (Lampirkan Kebijakan), Create Policy (Buat Kebijakan).

Tab atau jendela peramban baru akan terbuka.

5. Pilih tab JSON dan ketik dokumen kebijakan JSON berikut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

6. Setelah Anda selesai, pilih Tinjau kebijakan. Validator Kebijakan melaporkan kesalahan sintaksis.

7. Di halaman Review Policy (Tinjau Kebijakan), ketikkan Name (Nama) dan Description (Deskripsi) (optional) untuk kebijakan yang sedang Anda buat. Tinjau Summary (Ringkasan) kebijakan untuk melihat izin yang diberikan oleh kebijakan Anda. Kemudian pilih Buat kebijakan untuk menyimpan pekerjaan Anda.
8. Tutup tab atau jendela peramban, dan kembali ke halaman Add permissions (Tambahkan izin) untuk peran Anda. Pilih Refresh (Segarkan), lalu pilih kebijakan baru untuk dilampirkan ke peran Anda.
9. Pilih Lampirkan Kebijakan.

Langkah 2: Instal dan konfigurasikan CloudWatch Log pada instans Amazon EC2 yang ada

Proses untuk menginstal agen CloudWatch Log berbeda tergantung pada apakah instans Amazon EC2 Anda menjalankan Amazon Linux, Ubuntu, CentOS, atau Red Hat. Gunakan langkah-langkah yang sesuai untuk versi Linux di instans Anda.

Untuk menginstal dan mengonfigurasi CloudWatch Log pada instans Amazon Linux yang ada

Dimulai dengan Amazon Linux AMI 2014.09, agen CloudWatch Logs tersedia sebagai instalasi RPM dengan paket awslogs. Versi sebelumnya dari Amazon Linux dapat mengakses paket awslogs dengan memperbarui instans dengan perintah `sudo yum update -y`. Dengan menginstal paket awslogs sebagai RPM alih-alih menggunakan penginstal CloudWatch Log, instans Anda menerima pembaruan dan tambalan paket reguler AWS tanpa harus menginstal ulang agen Log secara manual. CloudWatch

 Warning

Jangan memperbarui agen CloudWatch Log menggunakan metode instalasi RPM jika sebelumnya Anda menggunakan skrip Python untuk menginstal agen. Melakukannya dapat menyebabkan masalah konfigurasi yang mencegah agen CloudWatch Log mengirim log Anda CloudWatch.

1. Hubungkan ke instans Amazon Linux Anda. Untuk informasi selengkapnya, lihat [Connect to Your Instance](#) di Panduan Pengguna Amazon EC2.

Untuk informasi selengkapnya tentang masalah koneksi, lihat [Pemecahan Masalah Menghubungkan ke Instans Anda](#) di Panduan Pengguna Amazon EC2.

2. Perbarui instans Amazon Linux Anda untuk mengambil perubahan terbaru dalam repositori paket.

```
sudo yum update -y
```

3. Instal `awslogs` paket. Ini adalah metode yang direkomendasikan untuk menginstal `awslogs` di instans Amazon Linux.

```
sudo yum install -y awslogs
```

4. Edit file `/etc/awslogs/awslogs.conf` untuk mengonfigurasi log yang akan dilacak. Untuk informasi selengkapnya tentang mengedit file ini, lihat [CloudWatch Referensi agen log](#).
5. Secara default, `/etc/awslogs/awsccli.conf` menunjuk ke Wilayah `us-east-1`. Untuk mendorong log Anda ke Wilayah yang berbeda, edit file `awsccli.conf` dan tentukan Wilayah tersebut.
6. Mulai layanan `awslogs`.

```
sudo service awslogs start
```

Jika Anda menjalankan Amazon Linux 2, mulai layanan `awslogs` dengan perintah berikut.

```
sudo systemctl start awslogsd
```

7. (Opsional) Periksa file `/var/log/awslogs.log` untuk kesalahan yang dicatat saat memulai layanan.
8. (Opsional) Jalankan perintah berikut untuk memulai layanan `awslogs` pada setiap boot sistem.

```
sudo chkconfig awslogs on
```

Jika Anda menjalankan Amazon Linux 2, gunakan perintah berikut untuk memulai layanan pada setiap boot sistem.

```
sudo systemctl enable awslogsd.service
```

9. Anda akan melihat grup log dan aliran log yang baru dibuat di CloudWatch konsol setelah agen berjalan selama beberapa saat.

Untuk informasi selengkapnya, lihat [Lihat data log yang dikirim ke CloudWatch Log](#).

Untuk menginstal dan mengkonfigurasi CloudWatch Log pada Server Ubuntu, CentOS, atau contoh Red Hat yang ada

Jika Anda menggunakan AMI yang menjalankan Ubuntu Server, CentOS, atau Red Hat, gunakan prosedur berikut untuk menginstal agen CloudWatch Log secara manual pada instance Anda.

1. Connect ke instans EC2 Anda. Untuk informasi selengkapnya, lihat [Connect to Your Instance](#) di Panduan Pengguna Amazon EC2.

Untuk informasi selengkapnya tentang masalah koneksi, lihat [Pemecahan Masalah Menghubungkan ke Instans Anda](#) di Panduan Pengguna Amazon EC2.

2. Jalankan penginstal agen CloudWatch Log menggunakan salah satu dari dua opsi. Anda dapat menjalankannya langsung dari internet, atau mengunduh file dan menjalankannya secara mandiri.

 Note

Jika Anda menjalankan CentOS 6.x, Red Hat 6.x, atau Ubuntu 12.04, gunakan langkah-langkah untuk mengunduh dan menjalankan penginstal mandiri. Menginstal agen CloudWatch Log langsung dari internet tidak didukung pada sistem ini.

 Note

Di Ubuntu, jalankan `apt-get update` sebelum menjalankan perintah di bawah ini.

Untuk menjalankannya secara langsung dari internet, gunakan arahan berikut dan ikuti arahannya:

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-setup.py -O
```

```
sudo python ./awslogs-agent-setup.py --region us-east-1
```

Jika perintah sebelumnya tidak berfungsi, coba hal berikut:

```
sudo python3 ./awslogs-agent-setup.py --region us-east-1
```

Untuk mengunduh dan menjalankannya secara mandiri, gunakan perintah berikut dan ikuti arahnya:

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-setup.py -O
```

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/AgentDependencies.tar.gz -O
```

```
tar xvf AgentDependencies.tar.gz -C /tmp/
```

```
sudo python ./awslogs-agent-setup.py --region us-east-1 --dependency-path /tmp/AgentDependencies
```

Anda dapat menginstal agen CloudWatch Log dengan menentukan us-east-1, us-west-1, us-west-2, ap-south-1, ap-south-1, ap-northeast-2, ap-southeast-2, ap-southeast-1, ap-southeast-1, ap-southeast-2, ap-northeast-1 Wilayah eu-central-1, eu-west-1, atau sa-east-1, atau sa-east-1.

Note

Untuk informasi selengkapnya tentang versi saat ini dan riwayat versi `awslogs-agent-setup`, lihat [CHANGELOG.txt](#).

Installer agen CloudWatch Log memerlukan informasi tertentu selama penyiapan. Sebelum memulai, Anda perlu mengetahui berkas log mana yang akan dipantau dan format stempel waktunya. Anda juga harus menyiapkan informasi berikut.

Item	Deskripsi
AWS ID kunci akses	Tekan Enter jika menggunakan IAM role. Jika tidak, masukkan ID kunci AWS akses Anda.

Item	Deskripsi
AWS kunci akses rahasia	Tekan Enter jika menggunakan IAM role. Jika tidak, masukkan kunci akses AWS rahasia Anda.
Nama Wilayah default	Tekan Enter. Default-nya adalah us-east-2. Anda dapat mengatur ini menjadi us-east-1, us-west-1, us-west-2, ap-south-1, ap-northeast-2, ap-southeast-1, ap-southeast-2, ap-northeast-1, eu-central-1, eu-west-1, atau sa-east-1.
Format output default	Kosongkan dan tekan Enter.
Jalur berkas log untuk diunggah	Lokasi file yang berisi data log untuk dikirim. Penginstal akan menyarankan jalur untuk Anda.
Nama Grup Log tujuan	Nama untuk grup log Anda. Penginstal akan menyarankan nama grup log untuk Anda.
Nama Pengaliran Log tujuan	Secara default, ini adalah nama host. Penginstal akan menyarankan nama host untuk Anda.
Format stempel waktu	Tentukan format stempel waktu dalam berkas log yang ditentukan. Pilih custom (khusus) untuk menentukan format Anda sendiri.
Posisi awal	Cara data diunggah. Atur ini menjadi start_of_file untuk mengunggah segala sesuatu dalam file data. Atur menjadi end_of_file untuk mengunggah hanya data yang baru ditambahkan.

Setelah menyelesaikan langkah-langkah ini, penginstal menanyakan tentang konfigurasi berkas log lainnya. Anda dapat menjalankan proses sebanyak yang Anda inginkan untuk setiap berkas log. Jika Anda tidak memiliki berkas log lagi untuk dipantau, pilih N saat diminta oleh penginstal untuk menyiapkan log lain. Untuk informasi selengkapnya tentang pengaturan di file konfigurasi agen, lihat [CloudWatch Referensi agen log](#).

Note

Mengonfigurasi beberapa sumber log untuk mengirim data ke satu pengaliran log tidaklah didukung.

3. Anda akan melihat grup log dan aliran log yang baru dibuat di CloudWatch konsol setelah agen berjalan selama beberapa saat.

Untuk informasi selengkapnya, lihat [Lihat data log yang dikirim ke CloudWatch Log](#).

Mulai Cepat: Instal dan konfigurasikan agen CloudWatch Log pada instans Linux EC2 saat diluncurkan

Tip

Agen CloudWatch Log lama yang dibahas di bagian ini sedang menuju penghentian. Kami sangat menyarankan agar Anda menggunakan CloudWatch agen terpadu baru yang dapat mengumpulkan log dan metrik. Selain itu, agen CloudWatch Log yang lebih lama memerlukan Python 3.3 atau yang lebih lama, dan versi ini tidak diinstal pada instans EC2 baru secara default. Untuk informasi selengkapnya tentang CloudWatch agen terpadu, lihat [Menginstal CloudWatch Agen](#).

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log yang lebih tua.

Menginstal agen CloudWatch Logs yang lebih lama pada instans Linux EC2 saat diluncurkan

Anda dapat menggunakan data pengguna Amazon EC2, fitur Amazon EC2 yang memungkinkan informasi parametrik diteruskan ke instans saat diluncurkan, untuk menginstal dan mengonfigurasi agen Log pada instance CloudWatch tersebut. Untuk meneruskan informasi instalasi dan konfigurasi agen CloudWatch Log ke Amazon EC2, Anda dapat menyediakan file konfigurasi di lokasi jaringan seperti bucket Amazon S3.

Mengonfigurasi beberapa sumber log untuk mengirim data ke satu pengaliran log tidaklah didukung.

Prasyarat

Buat file konfigurasi agen yang menjelaskan semua grup log dan pengaliran log Anda. Ini adalah file teks yang menjelaskan berkas log yang akan dipantau serta grup log dan pengaliran log untuk mengunggah berkas log. Agen mengonsumsi file konfigurasi ini dan mulai memantau dan mengunggah semua berkas log yang dijelaskan di dalamnya. Untuk informasi selengkapnya tentang pengaturan di file konfigurasi agen, lihat [CloudWatch Referensi agen log](#).

Berikut ini adalah sampel dari file konfigurasi agen untuk Amazon Linux 2

```
[general]
state_file = /var/lib/awslogs/state/agent-state

[/var/log/messages]
file = /var/log/messages
log_group_name = /var/log/messages
log_stream_name = {instance_id}
datetime_format = %b %d %H:%M:%S
```

Berikut ini adalah sampel dari file konfigurasi agen untuk Ubuntu

```
[general]
state_file = /var/awslogs/state/agent-state

[/var/log/syslog]
file = /var/log/syslog
log_group_name = /var/log/syslog
log_stream_name = {instance_id}
datetime_format = %b %d %H:%M:%S
```

Untuk mengonfigurasi IAM role

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi, pilih Policies (Kebijakan), Create Policy (Buat Kebijakan).
3. Di halaman Create Policy (Buat Kebijakan), untuk Create Your Own Policy (Buat Kebijakan Anda Sendiri), pilih Select (Pilih). Untuk informasi selengkapnya tentang membuat kebijakan khusus, lihat [Kebijakan IAM untuk Amazon EC2](#) di Panduan Pengguna Amazon EC2.
4. Di halaman Review Policy (Tinjau Kebijakan), untuk Policy Name (Nama Kebijakan), ketikkan nama untuk kebijakan tersebut.
5. Untuk Policy Document (Dokumen Kebijakan), tempelkan kebijakan berikut:

JSON

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

```

        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents",
            "logs:DescribeLogStreams"
        ],
        "Resource": [
            "arn:aws:logs:*:*:*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject"
        ],
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
    }
]
}

```

6. Pilih Buat Kebijakan.
7. Di panel navigasi, pilih Roles (Peran), Create New Role (Buat Peran Baru).
8. Di halaman Set Role Name (Tetapkan Nama Peran), ketik nama untuk peran tersebut, lalu pilih Next Step (Langkah Selanjutnya).
9. Di halaman Select Role Type (Pilih Jenis Peran), pilih Select (Pilihan) di samping Amazon EC2.
10. Di halaman Attach Policy (Lampirkan Kebijakan), di header tabel, pilih Policy Type (Jenis Kebijakan), Customer Managed (Dikelola Pelanggan).
11. Pilih kebijakan IAM yang sudah Anda buat, lalu pilih Next Step (Langkah Selanjutnya).
12. Pilih Buat Peran.

Untuk informasi selengkapnya tentang pengguna dan kebijakan, lihat [Pengguna dan Grup IAM dan Mengelola Kebijakan IAM](#) di Panduan Pengguna IAM.

Untuk meluncurkan instance baru dan mengaktifkan CloudWatch Log

1. Buka konsol Amazon EC2 di <https://console.aws.amazon.com/ec2/>

2. Pilih Luncurkan Instans.

Untuk informasi selengkapnya, lihat [Meluncurkan Instans](#) di Panduan Pengguna Amazon EC2.

3. Di halaman Langkah 1: Pilih Amazon Machine Image (AMI) pilih tipe instans Linux yang akan diluncurkan, lalu di halaman Langkah 2: Pilih Tipe Instans, pilih Selanjutnya: Konfigurasi Detail Instans.

Pastikan bahwa [cloud-init](#) termasuk dalam Amazon Machine Image (AMI) Anda. Amazon Linux AMIs, dan AMIs untuk Ubuntu dan RHEL sudah menyertakan cloud-init, tetapi CentOS dan lainnya AMIs mungkin tidak. AWS Marketplace

4. Di halaman Langkah 3: Konfigurasi Detail Instans, untuk IAM role, pilih IAM role yang sudah Anda buat.
5. Di Advanced Details (Detail Lanjutan), untuk User data (Data pengguna), tempelkan skrip berikut ke dalam kotak. Kemudian perbarui skrip tersebut dengan mengubah nilai opsi -c menjadi lokasi file konfigurasi agen Anda:

```
#!/bin/bash
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
setup.py -O
chmod +x ./awslogs-agent-setup.py
./awslogs-agent-setup.py -n -r us-east-1 -c s3://amzn-s3-demo-bucket/my-config-file
```

6. Buat perubahan lain pada instans, tinjau pengaturan peluncuran Anda, lalu pilih Launch (Luncurkan).
7. Anda akan melihat grup log dan aliran log yang baru dibuat di CloudWatch konsol setelah agen berjalan selama beberapa saat.

Untuk informasi selengkapnya, lihat [Lihat data log yang dikirim ke CloudWatch Log](#).

Mulai Cepat: Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2016 untuk mengirim log ke Log menggunakan agen CloudWatch Log CloudWatch

Tip

CloudWatch menyertakan agen terpadu baru yang dapat mengumpulkan log dan metrik dari instans EC2 dan server lokal. Kami menyarankan Anda menggunakan agen terpadu

CloudWatch yang lebih baru. Untuk informasi selengkapnya, lihat [Memulai dengan CloudWatch Log](#).

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log yang lebih tua.

Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2016 untuk mengirim log ke Log menggunakan agen CloudWatch Log yang lebih lama CloudWatch

Ada beberapa metode yang dapat Anda gunakan untuk mengaktifkan instance yang menjalankan Windows Server 2016 untuk mengirim CloudWatch log ke Log. Langkah-langkah di bagian ini menggunakan Systems Manager Run Command. Untuk informasi tentang metode lain yang mungkin, lihat [Mengirim Log, Acara, dan Penghitung Kinerja ke Amazon CloudWatch](#).

Langkah-langkah

- [Unduh file konfigurasi contoh](#)
- [Konfigurasi file JSON untuk CloudWatch](#)
- [Membuat peran IAM untuk Systems Manager](#)
- [Memverifikasi prasyarat Systems Manager](#)
- [Memverifikasi Akses Internet](#)
- [Aktifkan CloudWatch Log menggunakan Systems Manager Run Command](#)

Unduh file konfigurasi contoh

Unduh file contoh berikut ke komputer Anda: [AWS.EC2.Windows.CloudWatch.json](#).

Konfigurasi file JSON untuk CloudWatch

Anda menentukan log mana yang akan dikirim CloudWatch dengan menentukan pilihan Anda dalam file konfigurasi. Proses untuk membuat file ini dan menentukan pilihan Anda dapat memakan waktu 30 menit atau lebih untuk diselesaikan. Setelah Anda menyelesaikan tugas ini satu kali, Anda dapat menggunakan kembali file konfigurasi di semua instans Anda.

Langkah-langkah

- [Langkah 1: Aktifkan CloudWatch Log](#)
- [Langkah 2: Konfigurasi pengaturan untuk CloudWatch](#)

- [Langkah 3: Konfigurasi data untuk mengirim](#)
- [Langkah 4: Konfigurasi kontrol aliran](#)
- [Langkah 5: Simpan konten JSON](#)

Langkah 1: Aktifkan CloudWatch Log

Di bagian atas file JSON, ubah "false" menjadi "true" untuk IsEnabled:

```
"IsEnabled": true,
```

Langkah 2: Konfigurasikan pengaturan untuk CloudWatch

Tentukan kredensial, Wilayah, nama grup log, dan namespace pengaliran log. Hal ini memungkinkan instance untuk mengirim data log ke CloudWatch Log. Untuk mengirim data log yang sama ke lokasi yang berbeda, Anda dapat menambahkan bagian tambahan dengan unik IDs (misalnya, "CloudWatchLogs2" dan CloudWatchLogs 3") dan Wilayah yang berbeda untuk setiap ID.

Untuk mengkonfigurasi pengaturan untuk mengirim data log ke CloudWatch Log

1. Dalam file JSON, temukan bagian CloudWatchLogs.

```
{
  "Id": "CloudWatchLogs",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "AccessKey": "",
    "SecretKey": "",
    "Region": "us-east-1",
    "LogGroup": "Default-Log-Group",
    "LogStream": "{instance_id}"
  }
},
```

2. Biarkan bidang AccessKey dan SecretKey tetap kosong. Anda mengonfigurasi kredensial menggunakan IAM role.
3. Untuk Region, ketik Wilayah untuk mengirim data log (misalnya, us-east-2).
4. Untuk LogGroup, ketik nama untuk grup log Anda. Nama ini muncul di layar Grup Log di CloudWatch konsol.

5. Untuk LogStream, ketik pengaliran log tujuan. Nama ini muncul di layar Grup Log > Streams di CloudWatch konsol.

Jika Anda menggunakan `{instance_id}`, yaitu default-nya, nama pengaliran log adalah ID instans dari instans ini.

Jika Anda menentukan nama aliran log yang belum ada, CloudWatch Log secara otomatis membuatnya untuk Anda. Anda dapat menentukan nama pengaliran log menggunakan string literal, variabel yang telah ditetapkan `{instance_id}`, `{hostname}`, dan `{ip_address}`, atau kombinasinya.

Langkah 3: Konfigurasi data untuk mengirim

Anda dapat mengirim data log peristiwa, data Event Tracing for Windows (ETW), dan data log lainnya ke CloudWatch Log.

Untuk mengirim data log peristiwa aplikasi Windows ke CloudWatch Log

1. Dalam file JSON, temukan bagian `ApplicationEventLog`.

```
{
  "Id": "ApplicationEventLog",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Application",
    "Levels": "1"
  }
},
```

2. Untuk `Levels`, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:
 - **1** - Unggah hanya pesan kesalahan.
 - **2** - Unggah hanya pesan peringatan.
 - **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim data log keamanan ke CloudWatch Log

1. Dalam file JSON, temukan bagian `SecurityEventLog`.

```
{
  "Id": "SecurityEventLog",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Security",
    "Levels": "7"
  }
},
```

2. Untuk `Levels`, ketik **7** untuk mengunggah semua pesan.

Untuk mengirim data log peristiwa sistem ke CloudWatch Log

1. Dalam file JSON, temukan bagian `SystemEventLog`.

```
{
  "Id": "SystemEventLog",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "System",
    "Levels": "7"
  }
},
```

2. Untuk `Levels`, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:

- **1** - Unggah hanya pesan kesalahan.
- **2** - Unggah hanya pesan peringatan.

- **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim jenis data log peristiwa lainnya ke CloudWatch Log

1. Dalam file JSON, tambahkan bagian baru. Setiap bagian harus memiliki Id yang unik.

```
{
  "Id": "Id-name",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Log-name",
    "Levels": "7"
  }
},
```

2. Untuk Id, ketik nama untuk log yang akan diunggah (misalnya, **WindowsBackup**).
3. Untuk LogName, ketik nama log yang akan diunggah. Anda dapat menemukan nama log sebagai berikut.
 - a. Buka Event Viewer.
 - b. Di panel navigasi, pilih Applications and Services Logs (Log Aplikasi dan Layanan).
 - c. Buka log, lalu pilih Actions (Tindakan), Properties (Properti).
4. Untuk Levels, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:
 - **1** - Unggah hanya pesan kesalahan.
 - **2** - Unggah hanya pesan peringatan.
 - **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim Event Tracing untuk data Windows ke CloudWatch Log

ETW (Event Tracing for Windows) menyediakan mekanisme pencatatan log yang efisien dan terperinci yang dapat digunakan aplikasi untuk menuliskan log. Setiap ETW dikendalikan oleh manajer sesi yang dapat memulai dan menghentikan sesi pencatatan. Setiap sesi memiliki penyedia dan satu atau beberapa konsumen.

1. Dalam file JSON, temukan bagian ETW.

```
{
  "Id": "ETW",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Microsoft-Windows-WinINet/Analytic",
    "Levels": "7"
  }
},
```

2. Untuk LogName, ketik nama log yang akan diunggah.
3. Untuk Levels, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:
 - **1** - Unggah hanya pesan kesalahan.
 - **2** - Unggah hanya pesan peringatan.
 - **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim log kustom (file log berbasis teks apa pun) ke Log CloudWatch

1. Dalam file JSON, temukan bagian CustomLogs.

```
{
  "Id": "CustomLogs",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
```

```
"Parameters": {  
  "LogDirectoryPath": "C:\\\\CustomLogs\\\\",  
  "TimestampFormat": "MM/dd/yyyy HH:mm:ss",  
  "Encoding": "UTF-8",  
  "Filter": "",  
  "CultureName": "en-US",  
  "TimeZoneKind": "Local",  
  "LineCount": "5"  
},  
}
```

2. Untuk `LogDirectoryPath`, ketik jalur tempat log disimpan di instans Anda.
3. Untuk `TimestampFormat`, ketik format stempel waktu yang akan digunakan. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Custom Date and Time Format Strings](#) di MSDN.

 Important

Berkas log sumber Anda harus memiliki stempel waktu di awal setiap baris log dan harus ada spasi setelah stempel waktu.

4. Untuk `Encoding`, ketik pengodean file yang akan digunakan (misalnya, UTF-8). Untuk daftar nilai yang didukung, lihat topik [Encoding Class](#) di MSDN.

 Note

Gunakan nama pengodean, bukan nama tampilan.

5. (Opsional) Untuk `Filter`, ketik prefiks nama log. Biarkan parameter ini kosong untuk memantau semua file. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [FileSystemWatcherFilter Properti](#) di MSDN.
6. (Opsional) Untuk `CultureName`, ketik lokal tempat stempel waktu dicatat. Jika `CultureName` kosong, default-nya adalah lokal yang sama yang saat ini digunakan oleh instans Windows Anda. Untuk informasi selengkapnya, lihat kolom Language tag di tabel dalam topik [Product Behavior](#) di MSDN.

 Note

Nilai `div`, `div-MV`, `hu`, dan `hu-HU` tidak didukung.

7. (Opsional) Untuk `TimeZoneKind`, ketik `Local` atau `UTC`. Anda dapat mengatur ini untuk memberikan informasi zona waktu ketika tidak ada informasi zona waktu yang disertakan dalam stempel waktu log Anda. Jika parameter ini dibiarkan kosong dan jika stempel waktu Anda tidak menyertakan informasi zona waktu, CloudWatch Log default ke zona waktu lokal. Parameter ini diabaikan jika stempel waktu Anda sudah berisi informasi zona waktu.
8. (Opsional) Untuk `LineCount`, ketik jumlah baris di header untuk mengidentifikasi berkas log. Sebagai contoh, berkas log IIS memiliki header yang hampir identik. Anda bisa memasukkan `5`, yang akan membaca tiga baris pertama header berkas log untuk mengidentifikasinya. Dalam berkas log IIS, baris ketiga adalah tanggal dan stempel waktu, tetapi stempel waktu tidak selalu dijamin akan berbeda antara berkas log. Untuk alasan ini, sebaiknya sertakan setidaknya satu baris data log aktual untuk membuat sidik jari berkas log secara unik.

Untuk mengirim data log IIS ke CloudWatch Log

1. Dalam file JSON, temukan bagian `IISLog`.

```
{
  "Id": "IISLogs",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogDirectoryPath": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
    "TimestampFormat": "yyyy-MM-dd HH:mm:ss",
    "Encoding": "UTF-8",
    "Filter": "",
    "CultureName": "en-US",
    "TimeZoneKind": "UTC",
    "LineCount": "5"
  }
},
```

2. Untuk `LogDirectoryPath`, ketik folder tempat IIS log disimpan untuk situs individual (misalnya, `C:\\inetpub\\logs\\LogFiles\\W3SVC1`).

 Note

Hanya format log W3C yang didukung. Format IIS, NCSA, dan Custom tidak didukung.

3. Untuk `TimestampFormat`, ketik format stempel waktu yang akan digunakan. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Custom Date and Time Format Strings](#) di MSDN.
4. Untuk `Encoding`, ketik pengodean file yang akan digunakan (misalnya, UTF-8). Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Encoding Class](#) di MSDN.

 Note

Gunakan nama pengodean, bukan nama tampilan.

5. (Opsional) Untuk `Filter`, ketik prefiks nama log. Biarkan parameter ini kosong untuk memantau semua file. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [FileSystemWatcherFilter Properti](#) di MSDN.
6. (Opsional) Untuk `CultureName`, ketik lokal tempat stempel waktu dicatat. Jika `CultureName` kosong, default-nya adalah lokal yang sama yang saat ini digunakan oleh instans Windows Anda. Untuk informasi selengkapnya tentang nilai yang didukung, lihat kolom `Language` tag dalam tabel di topik [Product Behavior](#) di MSDN.

 Note

Nilai `div`, `div-MV`, `hu`, dan `hu-HU` tidak didukung.

7. (Opsional) Untuk `TimeZoneKind`, masukkan `Local` atau `UTC`. Anda dapat mengatur ini untuk memberikan informasi zona waktu ketika tidak ada informasi zona waktu yang disertakan dalam stempel waktu log Anda. Jika parameter ini dibiarkan kosong dan jika stempel waktu Anda tidak menyertakan informasi zona waktu, CloudWatch Log default ke zona waktu lokal. Parameter ini diabaikan jika stempel waktu Anda sudah berisi informasi zona waktu.
8. (Opsional) Untuk `LineCount`, ketik jumlah baris di header untuk mengidentifikasi berkas log. Sebagai contoh, berkas log IIS memiliki header yang hampir identik. Anda bisa memasukkan `5`, yang akan membaca lima baris pertama header berkas log untuk mengidentifikasinya. Dalam berkas log IIS, baris ketiga adalah tanggal dan stempel waktu, tetapi stempel waktu tidak selalu dijamin akan berbeda antara berkas log. Untuk alasan ini, sebaiknya sertakan setidaknya satu baris data log aktual untuk membuat sidik jari berkas log secara unik.

Langkah 4: Konfigurasi kontrol aliran

Setiap tipe data harus memiliki tujuan yang sesuai di bagian Flows. Misalnya, untuk mengirim log kustom, log ETW, dan log sistem ke CloudWatch Log, tambahkan (CustomLogs, ETW, SystemEventLog), CloudWatchLogs ke Flows bagian.

Warning

Menambahkan langkah yang tidak valid akan memblokir aliran. Misalnya, jika Anda menambahkan langkah metrik disk, tetapi instans Anda tidak memiliki disk, semua langkah dalam aliran akan diblokir.

Anda dapat mengirim berkas log yang sama ke lebih dari satu tujuan. Misalnya, untuk mengirim log aplikasi ke dua tujuan yang berbeda yang Anda tetapkan di bagian CloudWatchLogs, tambahkan ApplicationEventLog, (CloudWatchLogs, CloudWatchLogs2) ke bagian Flows.

Untuk mengonfigurasi kontrol aliran

1. Di file AWS.EC2.Windows.CloudWatch.json, temukan bagian Flows.

```
"Flows": {
  "Flows": [
    "PerformanceCounter,CloudWatch",
    "(PerformanceCounter,PerformanceCounter2), CloudWatch2",
    "(CustomLogs, ETW, SystemEventLog),CloudWatchLogs",
    "CustomLogs, CloudWatchLogs2",
    "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"
  ]
}
```

2. Untuk Flows, tambahkan setiap tipe data yang akan diunggah (misalnya, ApplicationEventLog) dan tujuannya (misalnya, CloudWatchLogs).

Langkah 5: Simpan konten JSON

Anda sekarang sudah selesai mengedit file JSON. Simpan file, dan tempel isi file ke editor teks di jendela lain. Anda akan membutuhkan isi file di langkah selanjutnya dalam prosedur ini.

Membuat peran IAM untuk Systems Manager

IAM role untuk kredensial instans diperlukan ketika Anda menggunakan Systems Manager Run Command. Peran ini memungkinkan Systems Manager untuk melakukan tindakan di instans. Untuk informasi selengkapnya, lihat [Mengonfigurasi Peran Keamanan untuk Systems Manager](#) di Panduan Pengguna AWS Systems Manager . Untuk informasi tentang cara melampirkan peran IAM ke instans yang ada, lihat [Melampirkan Peran IAM ke Instans di Panduan](#) Pengguna Amazon EC2.

Memverifikasi prasyarat Systems Manager

Sebelum Anda menggunakan Systems Manager Run Command untuk mengonfigurasi integrasi dengan CloudWatch Log, verifikasi bahwa instance Anda memenuhi persyaratan minimum. Untuk informasi selengkapnya, silakan lihat [Prasyarat Systems Manager](#) di Panduan Pengguna AWS Systems Manager .

Memverifikasi Akses Internet

Instans Amazon EC2 Windows Server dan instans terkelola harus memiliki akses internet keluar untuk mengirim data log dan peristiwa ke CloudWatch. Untuk informasi selengkapnya tentang cara mengonfigurasi akses internet, silakan lihat [Gateway Internet](#) dalam Panduan Pengguna VPC Amazon.

Aktifkan CloudWatch Log menggunakan Systems Manager Run Command

Run Command memungkinkan Anda mengelola konfigurasi instans Anda sesuai permintaan. Anda menentukan dokumen Manajer Sistem, menentukan parameter, dan mengeksekusi perintah pada satu atau beberapa instans. SSM agent di instans memproses perintah dan mengonfigurasi instans seperti yang ditentukan.

Untuk mengkonfigurasi integrasi dengan CloudWatch Log menggunakan Run Command

1. Buka konsol Amazon EC2 di <https://console.aws.amazon.com/ec2/>
2. Buka konsol SSM di <https://console.aws.amazon.com/systems-manager/>.
3. Di panel navigasi, pilih Jalankan Perintah.
4. Pilih Run a command (Jalankan perintah).
5. Untuk dokumen Command, pilih AWS- ConfigureCloudWatch.
6. Untuk instance Target, pilih instance yang akan diintegrasikan dengan CloudWatch Log. Jika Anda tidak melihat instans dalam daftar ini, instans tersebut mungkin tidak dikonfigurasi untuk

Run Command. Untuk informasi selengkapnya, lihat [Prasyarat Systems Manager di Panduan Pengguna](#) Amazon EC2.

7. Untuk Status, pilih Enabled (Diaktifkan).
8. Untuk Properties (Properti), salin dan tempel konten JSON yang Anda buat dalam tugas sebelumnya.
9. Selesaikan bidang opsional yang lainnya dan pilih Run (Jalankan).

Gunakan prosedur berikut untuk melihat hasil eksekusi perintah di konsol Amazon EC2.

Untuk melihat output perintah di konsol

1. Pilih perintah.
2. Pilih tab Output.
3. Pilih View Output (Lihat Output). Halaman output perintah menampilkan hasil eksekusi perintah Anda.

Mulai Cepat: Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2012 dan Windows Server 2008 untuk mengirim log ke Log CloudWatch



Tip

CloudWatch menyertakan agen terpadu baru yang dapat mengumpulkan log dan metrik dari instans EC2 dan server lokal. Kami menyarankan Anda menggunakan agen terpadu CloudWatch yang lebih baru. Untuk informasi selengkapnya, lihat [Memulai dengan CloudWatch Log](#).

Sisa bagian ini menjelaskan penggunaan agen CloudWatch Log yang lebih tua.

Aktifkan instans Amazon EC2 Anda yang menjalankan Windows Server 2012 dan Windows Server 2008 untuk mengirim log ke Log CloudWatch

Gunakan langkah-langkah berikut untuk mengaktifkan instance Anda yang menjalankan Windows Server 2012 dan Windows Server 2008 untuk mengirim CloudWatch log ke Log.

Unduh file konfigurasi contoh

Unduh file JSON contoh berikut ke komputer Anda: [AWS.EC2.Windows.CloudWatch.json](#). Anda akan mengeditnya dalam langkah-langkah berikut.

Konfigurasi file JSON untuk CloudWatch

Anda menentukan log mana yang akan dikirim CloudWatch dengan menentukan pilihan Anda di file konfigurasi JSON. Proses untuk membuat file ini dan menentukan pilihan Anda dapat memakan waktu 30 menit atau lebih untuk diselesaikan. Setelah Anda menyelesaikan tugas ini satu kali, Anda dapat menggunakan kembali file konfigurasi di semua instans Anda.

Langkah-langkah

- [Langkah 1: Aktifkan CloudWatch Log](#)
- [Langkah 2: Konfigurasi pengaturan untuk CloudWatch](#)
- [Langkah 3: Konfigurasi data untuk mengirim](#)
- [Langkah 4: Konfigurasi kontrol aliran](#)

Langkah 1: Aktifkan CloudWatch Log

Di bagian atas file JSON, ubah "false" menjadi "true" untuk `IsEnabled`:

```
"IsEnabled": true,
```

Langkah 2: Konfigurasi pengaturan untuk CloudWatch

Tentukan kredensial, Wilayah, nama grup log, dan namespace pengaliran log. Hal ini memungkinkan instance untuk mengirim data log ke CloudWatch Log. Untuk mengirim data log yang sama ke lokasi yang berbeda, Anda dapat menambahkan bagian tambahan dengan unik IDs (misalnya, "CloudWatchLogs2" dan CloudWatchLogs 3") dan Wilayah yang berbeda untuk setiap ID.

Untuk mengkonfigurasi pengaturan untuk mengirim data log ke CloudWatch Log

1. Dalam file JSON, temukan bagian `CloudWatchLogs`.

```
{
  "Id": "CloudWatchLogs",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
```

```

    "AccessKey": "",
    "SecretKey": "",
    "Region": "us-east-1",
    "LogGroup": "Default-Log-Group",
    "LogStream": "{instance_id}"
  }
},

```

2. Biarkan bidang AccessKey dan SecretKey tetap kosong. Anda mengonfigurasi kredensial menggunakan IAM role.
3. Untuk Region, ketik Wilayah untuk mengirim data log (misalnya, us-east-2).
4. Untuk LogGroup, ketik nama untuk grup log Anda. Nama ini muncul di layar Grup Log di CloudWatch konsol.
5. Untuk LogStream, ketik pengaliran log tujuan. Nama ini muncul di layar Grup Log > Streams di CloudWatch konsol.

Jika Anda menggunakan `{instance_id}`, yaitu default-nya, nama pengaliran log adalah ID instans dari instans ini.

Jika Anda menentukan nama aliran log yang belum ada, CloudWatch Log secara otomatis membuatnya untuk Anda. Anda dapat menentukan nama pengaliran log menggunakan string literal, variabel yang telah ditetapkan `{instance_id}`, `{hostname}`, dan `{ip_address}`, atau kombinasinya.

Langkah 3: Konfigurasi data untuk mengirim

Anda dapat mengirim data log peristiwa, data Event Tracing for Windows (ETW), dan data log lainnya ke CloudWatch Log.

Untuk mengirim data log peristiwa aplikasi Windows ke CloudWatch Log

1. Dalam file JSON, temukan bagian `ApplicationEventLog`.

```

{
  "Id": "ApplicationEventLog",
  "FullName":
"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Application",
    "Levels": "1"
  }
}

```

```
}
},
```

2. Untuk **Levels**, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:

- **1** - Unggah hanya pesan kesalahan.
- **2** - Unggah hanya pesan peringatan.
- **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim data log keamanan ke CloudWatch Log

1. Dalam file JSON, temukan bagian **SecurityEventLog**.

```
{
  "Id": "SecurityEventLog",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Security",
    "Levels": "7"
  }
},
```

2. Untuk **Levels**, ketik **7** untuk mengunggah semua pesan.

Untuk mengirim data log peristiwa sistem ke CloudWatch Log

1. Dalam file JSON, temukan bagian **SystemEventLog**.

```
{
  "Id": "SystemEventLog",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
```

```

      "LogName": "System",
      "Levels": "7"
    }
  },

```

2. Untuk `Levels`, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:

- **1** - Unggah hanya pesan kesalahan.
- **2** - Unggah hanya pesan peringatan.
- **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim jenis data log peristiwa lainnya ke CloudWatch Log

1. Dalam file JSON, tambahkan bagian baru. Setiap bagian harus memiliki `Id` yang unik.

```

{
  "Id": "Id-name",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Log-name",
    "Levels": "7"
  }
},

```

2. Untuk `Id`, ketik nama untuk log yang akan diunggah (misalnya, **WindowsBackup**).
3. Untuk `LogName`, ketik nama log yang akan diunggah. Anda dapat menemukan nama log sebagai berikut.
 - a. Buka Event Viewer.
 - b. Di panel navigasi, pilih Applications and Services Logs (Log Aplikasi dan Layanan).
 - c. Buka log, lalu pilih Actions (Tindakan), Properties (Properti).

4. Untuk **Levels**, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:

- **1** - Unggah hanya pesan kesalahan.
- **2** - Unggah hanya pesan peringatan.
- **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim Event Tracing untuk data Windows ke CloudWatch Log

ETW (Event Tracing for Windows) menyediakan mekanisme pencatatan log yang efisien dan terperinci yang dapat digunakan aplikasi untuk menuliskan log. Setiap ETW dikendalikan oleh manajer sesi yang dapat memulai dan menghentikan sesi pencatatan. Setiap sesi memiliki penyedia dan satu atau beberapa konsumen.

1. Dalam file JSON, temukan bagian ETW.

```
{
  "Id": "ETW",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogName": "Microsoft-Windows-WinINet/Analytic",
    "Levels": "7"
  }
},
```

2. Untuk **LogName**, ketik nama log yang akan diunggah.

3. Untuk **Levels**, tentukan jenis pesan yang akan diunggah. Anda dapat menentukan salah satu nilai berikut:

- **1** - Unggah hanya pesan kesalahan.
- **2** - Unggah hanya pesan peringatan.
- **4** - Unggah hanya pesan informasi.

Anda dapat menggabungkan nilai-nilai untuk menyertakan lebih dari satu jenis pesan. Misalnya, nilai **3** mengunggah pesan kesalahan (**1**) dan pesan peringatan (**2**). Nilai **7** mengunggah pesan kesalahan (**1**), pesan peringatan (**2**), dan pesan informasi (**4**).

Untuk mengirim log khusus (file log berbasis teks apa pun) ke Log CloudWatch

1. Dalam file JSON, temukan bagian CustomLogs.

```
{
  "Id": "CustomLogs",
  "FullName":
  "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
    "LogDirectoryPath": "C:\\\\CustomLogs\\\\",
    "TimestampFormat": "MM/dd/yyyy HH:mm:ss",
    "Encoding": "UTF-8",
    "Filter": "",
    "CultureName": "en-US",
    "TimeZoneKind": "Local",
    "LineCount": "5"
  }
},
```

2. Untuk LogDirectoryPath, ketik jalur tempat log disimpan di instans Anda.
3. Untuk TimestampFormat, ketik format stempel waktu yang akan digunakan. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Custom Date and Time Format Strings](#) di MSDN.

 Important

Berkas log sumber Anda harus memiliki stempel waktu di awal setiap baris log dan harus ada spasi setelah stempel waktu.

4. Untuk Encoding, ketik pengodean file yang akan digunakan (misalnya, UTF-8). Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Encoding Class](#) di MSDN.

 Note

Gunakan nama pengodean, bukan nama tampilan.

5. (Opsional) Untuk `Filter`, ketik prefiks nama log. Biarkan parameter ini kosong untuk memantau semua file. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [FileSystemWatcherFilter Properti](#) di MSDN.
6. (Opsional) Untuk `CultureName`, ketik lokal tempat stempel waktu dicatat. Jika `CultureName` kosong, default-nya adalah lokal yang sama yang saat ini digunakan oleh instans Windows Anda. Untuk informasi selengkapnya tentang nilai yang didukung, lihat kolom `Language` tag dalam tabel di topik [Product Behavior](#) di MSDN.

 Note

Nilai `div`, `div-MV`, `hu`, dan `hu-HU` tidak didukung.

7. (Opsional) Untuk `TimeZoneKind`, ketik `Local` atau `UTC`. Anda dapat mengatur ini untuk memberikan informasi zona waktu ketika tidak ada informasi zona waktu yang disertakan dalam stempel waktu log Anda. Jika parameter ini dibiarkan kosong dan jika stempel waktu Anda tidak menyertakan informasi zona waktu, CloudWatch Log default ke zona waktu lokal. Parameter ini diabaikan jika stempel waktu Anda sudah berisi informasi zona waktu.
8. (Opsional) Untuk `LineCount`, ketik jumlah baris di header untuk mengidentifikasi berkas log. Sebagai contoh, berkas log IIS memiliki header yang hampir identik. Anda bisa memasukkan `5`, yang akan membaca tiga baris pertama header berkas log untuk mengidentifikasinya. Dalam berkas log IIS, baris ketiga adalah tanggal dan stempel waktu, tetapi stempel waktu tidak selalu dijamin akan berbeda antara berkas log. Untuk alasan ini, sebaiknya sertakan setidaknya satu baris data log aktual untuk membuat sidik jari berkas log secara unik.

Untuk mengirim data log IIS ke CloudWatch Log

1. Dalam file JSON, temukan bagian `IISLog`.

```
{
  "Id": "IISLogs",
  "FullName":
    "AWS.EC2.Windows.CloudWatch.CustomLog.CustomLogInputComponent,AWS.EC2.Windows.CloudWatch",
  "Parameters": {
```

```
"LogDirectoryPath": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",  
"TimestampFormat": "yyyy-MM-dd HH:mm:ss",  
"Encoding": "UTF-8",  
"Filter": "",  
"CultureName": "en-US",  
"TimeZoneKind": "UTC",  
"LineCount": "5"  
}  
,
```

2. Untuk `LogDirectoryPath`, ketik folder tempat IIS log disimpan untuk situs individual (misalnya, `C:\\inetpub\\logs\\LogFiles\\W3SVC1`).

 Note

Hanya format log W3C yang didukung. Format IIS, NCSA, dan Custom tidak didukung.

3. Untuk `TimestampFormat`, ketik format stempel waktu yang akan digunakan. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Custom Date and Time Format Strings](#) di MSDN.
4. Untuk `Encoding`, ketik pengodean file yang akan digunakan (misalnya, UTF-8). Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [Encoding Class](#) di MSDN.

 Note

Gunakan nama pengodean, bukan nama tampilan.

5. (Opsional) Untuk `Filter`, ketik prefiks nama log. Biarkan parameter ini kosong untuk memantau semua file. Untuk informasi selengkapnya tentang nilai yang didukung, lihat topik [FileSystemWatcherFilter Properti](#) di MSDN.
6. (Opsional) Untuk `CultureName`, ketik lokal tempat stempel waktu dicatat. Jika `CultureName` kosong, default-nya adalah lokal yang sama yang saat ini digunakan oleh instans Windows Anda. Untuk informasi selengkapnya tentang nilai yang didukung, lihat kolom Language tag dalam tabel di topik [Product Behavior](#) di MSDN.

 Note

Nilai `div`, `div-MV`, `hu`, dan `hu-HU` tidak didukung.

7. (Opsional) Untuk `TimeZoneKind`, masukkan `Local` atau `UTC`. Anda dapat mengatur ini untuk memberikan informasi zona waktu ketika tidak ada informasi zona waktu yang disertakan dalam stempel waktu log Anda. Jika parameter ini dibiarkan kosong dan jika stempel waktu Anda tidak menyertakan informasi zona waktu, CloudWatch Log default ke zona waktu lokal. Parameter ini diabaikan jika stempel waktu Anda sudah berisi informasi zona waktu.
8. (Opsional) Untuk `LineCount`, ketik jumlah baris di header untuk mengidentifikasi berkas log. Sebagai contoh, berkas log IIS memiliki header yang hampir identik. Anda bisa memasukkan `5`, yang akan membaca lima baris pertama header berkas log untuk mengidentifikasinya. Dalam berkas log IIS, baris ketiga adalah tanggal dan stempel waktu, tetapi stempel waktu tidak selalu dijamin akan berbeda antara berkas log. Untuk alasan ini, sebaiknya sertakan setidaknya satu baris data log aktual untuk membuat sidik jari berkas log secara unik.

Langkah 4: Konfigurasi kontrol aliran

Setiap tipe data harus memiliki tujuan yang sesuai di bagian `Flows`. Misalnya, untuk mengirim log kustom, log ETW, dan log sistem ke CloudWatch Log, tambahkan (`CustomLogs`, `ETW`, `SystemEventLog`), `CloudWatchLogs` ke `Flows` bagian.

Warning

Menambahkan langkah yang tidak valid akan memblokir aliran. Misalnya, jika Anda menambahkan langkah metrik disk, tetapi instans Anda tidak memiliki disk, semua langkah dalam aliran akan diblokir.

Anda dapat mengirim berkas log yang sama ke lebih dari satu tujuan. Misalnya, untuk mengirim log aplikasi ke dua tujuan yang berbeda yang Anda tetapkan di bagian `CloudWatchLogs`, tambahkan `ApplicationEventLog`, (`CloudWatchLogs`, `CloudWatchLogs2`) ke bagian `Flows`.

Untuk mengonfigurasi kontrol aliran

1. Di file `AWS.EC2.Windows.CloudWatch.json`, temukan bagian `Flows`.

```
"Flows": {
  "Flows": [
    "PerformanceCounter,CloudWatch",
    "(PerformanceCounter,PerformanceCounter2), CloudWatch2",
    "(CustomLogs, ETW, SystemEventLog),CloudWatchLogs",
```

```
"CustomLogs, CloudWatchLogs2",  
  "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"  
]  
}
```

2. Untuk Flows, tambahkan setiap tipe data yang akan diunggah (misalnya, `ApplicationEventLog`) dan tujuannya (misalnya, `CloudWatchLogs`).

Anda sekarang sudah selesai mengedit file JSON. Anda akan menggunakannya di langkah berikutnya.

Memulai agen

Untuk mengaktifkan instans Amazon EC2 yang menjalankan Windows Server 2012 atau Windows Server 2008 untuk mengirim CloudWatch log ke Log, gunakan layanan EC2 Config (`EC2Config.exe`). Instance Anda harus memiliki EC2 Config 4.0 atau yang lebih baru, dan Anda dapat menggunakan prosedur ini.

Untuk mengkonfigurasi CloudWatch menggunakan EC2 Config 4.x

1. Periksa pengodean file `AWS.EC2.Windows.CloudWatch.json` yang Anda edit sebelumnya dalam prosedur ini. Hanya pengodean UTF-8 tanpa BOM yang didukung. Kemudian simpan file di folder berikut di instans Windows Server 2008 - 2012 R2: `C:\Program Files\Amazon\SSM\Plugins\awsCloudWatch\`.
2. Mulai atau mulai ulang agen SSM (`AmazonSSMAgent.exe`) menggunakan panel kontrol Layanan Windows atau menggunakan PowerShell perintah berikut:

```
PS C:\> Restart-Service AmazonSSMAgent
```

Setelah agen SSM restart, ia mendeteksi file konfigurasi dan mengkonfigurasi instance untuk integrasi. CloudWatch Jika Anda mengubah parameter dan pengaturan dalam file konfigurasi lokal, Anda perlu memulai ulang SSM agent untuk mengikuti perubahannya. Untuk menonaktifkan CloudWatch integrasi pada instance, ubah `IsEnabled` ke `false` dan simpan perubahan Anda dalam file konfigurasi.

Laporkan status agen CloudWatch Log

Gunakan prosedur berikut untuk melaporkan status agen CloudWatch Log pada instans EC2 Anda.

Untuk melaporkan status agen

1. Hubungkan ke instans EC2 Anda. Untuk informasi selengkapnya, lihat [Connect to Your Instance](#) di Panduan Pengguna Amazon EC2.

Untuk informasi selengkapnya tentang masalah koneksi, lihat [Memecahkan Masalah Menyambung ke Instans Anda](#) di Panduan Pengguna Amazon EC2

2. Di jendela perintah, jalankan perintah berikut:

```
sudo service awslogs status
```

Jika Anda menjalankan Amazon Linux 2, ketik perintah berikut:

```
sudo service awslogsd status
```

3. Periksa file `/var/log/awslogs.log` untuk kesalahan, peringatan, atau masalah apa pun dengan agen CloudWatch Log.

Mulai agen CloudWatch Log

Jika agen CloudWatch Log pada instans EC2 Anda tidak memulai secara otomatis setelah instalasi, atau jika Anda menghentikan agen, Anda dapat menggunakan prosedur berikut untuk memulai agen.

Untuk memulai agen

1. Connect ke instans EC2 Anda. Untuk informasi selengkapnya, lihat [Connect to Your Instance](#) di Panduan Pengguna Amazon EC2.

Untuk informasi selengkapnya tentang masalah koneksi, lihat [Pemecahan Masalah Menghubungkan ke Instans Anda](#) di Panduan Pengguna Amazon EC2.

2. Di jendela perintah, jalankan perintah berikut:

```
sudo service awslogs start
```

Jika Anda menjalankan Amazon Linux 2, ketik perintah berikut:

```
sudo service awslogsd start
```

Hentikan agen CloudWatch Log

Gunakan prosedur berikut untuk menghentikan agen CloudWatch Log pada instans EC2 Anda.

Untuk menghentikan agen

1. Connect ke instans EC2 Anda. Untuk informasi selengkapnya, lihat [Connect to Your Instance](#) di Panduan Pengguna Amazon EC2.

Untuk informasi selengkapnya tentang masalah koneksi, lihat [Pemecahan Masalah Menghubungkan ke Instans Anda](#) di Panduan Pengguna Amazon EC2.

2. Di jendela perintah, jalankan perintah berikut:

```
sudo service awslogs stop
```

Jika Anda menjalankan Amazon Linux 2, ketik perintah berikut:

```
sudo service awslogsd stop
```

CloudWatch Referensi agen log

Important

Bagian ini adalah referensi bagi mereka yang menggunakan agen Log usang. CloudWatch Jika menggunakan Instance Metadata Service Version 2 (IMDSv2), Anda harus menggunakan agen terpadu CloudWatch yang baru. Namun, meskipun Anda tidak menggunakan IMDSv2, kami sangat menyarankan untuk menggunakan CloudWatch agen terpadu yang lebih baru daripada agen Log yang tidak digunakan lagi. CloudWatch Untuk informasi tentang CloudWatch agen terpadu yang lebih baru, lihat [Mengumpulkan metrik dan log dari instans Amazon EC2 dan server lokal](#) dengan agen. CloudWatch Untuk informasi tentang migrasi dari agen CloudWatch Log yang tidak digunakan lagi ke agen terpadu, [Buat file konfigurasi CloudWatch agen dengan wizard](#).

Agan CloudWatch Log menyediakan cara otomatis untuk mengirim data log ke Log dari CloudWatch instans Amazon EC2. Agen meliputi komponen berikut:

- Plug-in untuk AWS CLI yang mendorong data log ke CloudWatch Log.
- Skrip (daemon) yang memulai proses untuk mendorong data ke Log. CloudWatch
- Tugas cron yang memastikan bahwa daemon selalu berjalan.

File konfigurasi agen

File konfigurasi agen CloudWatch Log menjelaskan informasi yang dibutuhkan oleh agen CloudWatch Log. Bagian [general] file konfigurasi agen mendefinisikan konfigurasi umum yang berlaku untuk semua pengaliran log. Bagian [logstream] menentukan informasi yang diperlukan untuk mengirim file lokal ke pengaliran log jarak jauh. Anda dapat memiliki lebih dari satu bagian [logstream], tetapi masing-masing harus memiliki nama yang unik dalam file konfigurasi, misalnya [logstream1], [logstream2], dan seterusnya. Nilai [logstream] beserta baris pertama data dalam berkas log akan menentukan identitas berkas log.

```
[general]
state_file = value
logging_config_file = value
use_gzip_http_content_encoding = [true | false]

[logstream1]
log_group_name = value
log_stream_name = value
datetime_format = value
time_zone = [LOCAL|UTC]
file = value
file_fingerprint_lines = integer | integer-integer
multi_line_start_pattern = regex | {datetime_format}
initial_position = [start_of_file | end_of_file]
encoding = [ascii|utf_8|..]
buffer_duration = integer
batch_count = integer
batch_size = integer

[logstream2]
...
```

state_file

Menentukan tempat file state disimpan.

logging_config_file

(Opsional) Menentukan lokasi file konfigurasi pencatatan agen. Jika Anda tidak menentukan file konfigurasi pencatatan agen di sini, file default `awslogs.conf` akan digunakan. Lokasi file default adalah `/var/awslogs/etc/awslogs.conf` jika Anda menginstal agen dengan skrip, dan `/etc/awslogs/awslogs.conf` jika Anda menginstal agen dengan rpm. File ini dalam format file konfigurasi Python (<https://docs.pylogging-config-fileformatthon.org/2/library/logging.config.html#>). Pencatat log dengan nama berikut dapat disesuaikan.

```
cwlogs.push
cwlogs.push.reader
cwlogs.push.publisher
cwlogs.push.event
cwlogs.push.batch
cwlogs.push.stream
cwlogs.push.watcher
```

Contoh di bawah ini mengubah tingkat pembaca dan penerbit menjadi WARNING sementara nilai default-nya adalah INFO.

```
[loggers]
keys=root,cwlogs,reader,publisher

[handlers]
keys=consoleHandler

[formatters]
keys=simpleFormatter

[logger_root]
level=INFO
handlers=consoleHandler

[logger_cwlogs]
level=INFO
handlers=consoleHandler
qualname=cwlogs.push
propagate=0

[logger_reader]
level=WARNING
handlers=consoleHandler
```

```

qualname=cwlogs.push.reader
propagate=0

[logger_publisher]
level=WARNING
handlers=consoleHandler
qualname=cwlogs.push.publisher
propagate=0

[handler_consoleHandler]
class=logging.StreamHandler
level=INFO
formatter=simpleFormatter
args=(sys.stderr,)

[formatter_simpleFormatter]
format=%(asctime)s - %(name)s - %(levelname)s - %(process)d - %(threadName)s -
%(message)s

```

use_gzip_http_content_encoding

Saat disetel ke true (default), aktifkan pengkodean konten http gzip untuk mengirim muatan terkompresi ke Log. CloudWatch Ini mengurangi penggunaan CPU, menurunkan NetworkOut, dan mengurangi latensi put. Untuk menonaktifkan fitur ini, tambahkan `use_gzip_http_content_encoding = false` ke bagian [umum] dari file konfigurasi agen Logs, lalu restart agen. CloudWatch

Note

Pengaturan ini hanya tersedia di awscli-cwlogs versi 1.3.3 dan yang lebih baru.

log_group_name

Menentukan grup log tujuan. Jika belum ada, grup log akan dibuat secara otomatis. Nama grup log dapat berisi antara 1 dan 512 karakter. Karakter yang diperbolehkan meliputi a–z, A–Z, 0–9, '_' (garis bawah), '-' (tanda hubung), '/' (garis miring), dan '.' (titik).

log_stream_name

Menentukan pengaliran log tujuan. Anda dapat menggunakan string literal atau variabel yang telah ditetapkan ({instance_id}, {hostname}, {ip_address}), atau kombinasi keduanya untuk menentukan nama pengaliran log. Jika belum ada, pengaliran log akan dibuat secara otomatis.

datetime_format

Menentukan bagaimana stempel waktu diekstraksi dari log. Stempel waktu digunakan untuk mengambil log acara dan menghasilkan metrik. Waktu saat ini akan digunakan untuk setiap log acara jika `datetime_format` tidak disediakan. Jika nilai `datetime_format` yang diberikan tidak valid untuk pesan log tertentu, stempel waktu dari log acara terakhir dengan stempel waktu yang berhasil diurai akan digunakan. Jika tidak ada log acara sebelumnya, waktu saat ini akan digunakan.

Kode `datetime_format` yang umum tercantum di bawah ini. Anda juga dapat menggunakan kode `datetime_format` yang didukung oleh Python, `datetime.strptime()`. Pengimbangan zona waktu (`%z`) juga didukung meskipun itu tidak didukung sebelum python 3.2, `[+-]HHMM` tanpa titik dua(`:`). Untuk informasi selengkapnya, lihat [Perilaku `strptime\(\)` dan `strptime\(\)`](#).

`%y`: Tahun tanpa abad sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 00, 01,..., 99

`%Y`: Tahun dengan abad sebagai angka desimal. 1970, 1988, 2001, 2013

`%b`: Bulan sebagai nama singkat lokal. Jan, Feb, ..., Des (id_ID);

`%B`: Bulan sebagai nama lengkap lokal. Januari, Februari, ..., Desember (id_ID);

`%m`: Bulan sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 01, 02,..., 12

`%d`: Tanggal dalam bulan sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 01, 02,..., 31

`%H`: Jam (24 jam) sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 00, 01,..., 23

`%I`: Jam (12 jam) sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 01, 02,..., 12

`%p`: Istilah lokal yang setara dengan AM atau PM.

`%M`: Menit sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 00, 01,..., 59

`%S`: Detik sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 00, 01,..., 59

%f: Mikrosekon sebagai angka desimal yang ditambah angka nol di depan angka satu digit (zero-padded). 000000,..., 999999

%z: Pengimbangan UTC dalam bentuk +HHMM atau -HHMM. +0000, -0400, +1030

Contoh format:

Syslog: '%b %d %H:%M:%S', e.g. Jan 23 20:59:29

Log4j: '%d %b %Y %H:%M:%S', e.g. 24 Jan 2014 05:00:00

ISO8601: '%Y-%m-%dT%H:%M:%S%z', e.g. 2014-02-20T05:20:20+0000

time_zone

Menentukan zona stempel waktu log acara. Dua nilai yang didukung adalah UTC dan LOCAL. Default-nya adalah LOCAL, yang digunakan jika zona waktu tidak dapat disimpulkan berdasarkan `datetime_format`.

berkas

Menentukan file log yang ingin Anda push ke CloudWatch Log. File dapat menunjuk ke file tertentu atau beberapa file (menggunakan wildcard seperti `/var/log/system.log*`). Hanya file terbaru yang didorong ke CloudWatch Log berdasarkan waktu modifikasi file. Kami sarankan Anda menggunakan wildcard untuk menentukan serangkaian file dengan jenis yang sama, seperti `access_log.2014-06-01-01`, `access_log.2014-06-01-02`, dan seterusnya, tetapi bukan beberapa jenis file, seperti `access_log_80` dan `access_log_443`. Untuk menentukan beberapa jenis file, tambahkan entri pengaliran log lain ke file konfigurasi agar setiap jenis berkas log pergi ke pengaliran log yang berbeda. File terkompresi tidak didukung.

file_fingerprint_lines

Menentukan rentang baris untuk mengidentifikasi file. Nilai yang valid adalah satu angka atau dua angka yang dibatasi dengan tanda hubung, seperti '1', '2-5'. Nilai default-nya adalah '1' sehingga baris pertama digunakan untuk menghitung sidik jari. Garis sidik jari tidak dikirim ke CloudWatch Log kecuali semua baris yang ditentukan tersedia.

multi_line_start_pattern

Menentukan pola untuk mengidentifikasi awal pesan log. Pesan log dibuat dari baris yang sesuai dengan pola dan baris berikutnya yang tidak cocok dengan pola. Nilai yang valid adalah ekspresi reguler atau `{datetime_format}`. Jika menggunakan `{datetime_format}`, pilihan `datetime_format`

harus ditentukan. Nilai default-nya adalah '^[\s]' sehingga semua baris yang dimulai dengan karakter yang bukan merupakan spasi kosong akan menutup pesan log sebelumnya dan memulai pesan log baru.

initial_position

Menentukan tempat untuk memulai membaca data (start_of_file atau end_of_file). Default-nya adalah start_of_file. Ini hanya digunakan jika tidak ada keadaan yang dipertahankan untuk pengaliran log tersebut.

encoding

Menentukan pengodean berkas log agar file dapat dibaca dengan benar. Default-nya adalah utf_8. Pengodean yang didukung oleh Python codecs.decode() dapat digunakan di sini.

Warning

Jika Anda menentukan pengodean yang salah, mungkin akan ada kehilangan data karena karakter yang tidak dapat didekode diganti dengan karakter lain.

Berikut adalah beberapa pengodean umum:

```
ascii, big5, big5hkscs, cp037, cp424, cp437, cp500, cp720, cp737,
cp775, cp850, cp852, cp855, cp856, cp857, cp858, cp860, cp861, cp862,
cp863, cp864, cp865, cp866, cp869, cp874, cp875, cp932, cp949, cp950,
cp1006, cp1026, cp1140, cp1250, cp1251, cp1252, cp1253, cp1254, cp1255,
cp1256, cp1257, cp1258, euc_jp, euc_jis_2004, euc_jisx0213, euc_kr,
gb2312, gbk, gb18030, hz, iso2022_jp, iso2022_jp_1, iso2022_jp_2,
iso2022_jp_2004, iso2022_jp_3, iso2022_jp_ext, iso2022_kr, latin_1,
iso8859_2, iso8859_3, iso8859_4, iso8859_5, iso8859_6, iso8859_7,
iso8859_8, iso8859_9, iso8859_10, iso8859_13, iso8859_14, iso8859_15,
iso8859_16, johab, koi8_r, koi8_u, mac_cyrillic, mac_greek, mac_iceland,
mac_latin2, mac_roman, mac_turkish, ptcp154, shift_jis, shift_jis_2004,
shift_jisx0213, utf_32, utf_32_be, utf_32_le, utf_16, utf_16_be,
utf_16_le, utf_7, utf_8, utf_8_sig
```

buffer_duration

Menentukan durasi waktu untuk pembuatan batch log acara. Nilai minimumnya adalah 5000ms dan nilai default-nya adalah 5000ms.

batch_count

Menentukan jumlah maks log acara dalam batch, maksimum 10000. Nilai default-nya adalah 10000.

batch_size

Menentukan ukuran maks log acara dalam batch, dalam byte, maksimal 1048576 byte. Nilai default-nya adalah 1048576. Ukuran ini dihitung sebagai jumlah semua pesan kejadian dalam UTF-8, ditambah 26 byte untuk setiap log acara.

Menggunakan agen CloudWatch Log dengan proxy HTTP

Anda dapat menggunakan agen CloudWatch Log dengan proxy HTTP.

Note

Proksi HTTP didukung awslogs-agent-setup dalam.py versi 1.3.8 atau yang lebih baru.

Untuk menggunakan agen CloudWatch Log dengan proxy HTTP

1. Lakukan salah satu tindakan berikut:

a. Untuk instalasi baru agen CloudWatch Log, jalankan perintah berikut:

```
curl https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-setup.py -O
```

```
sudo python awslogs-agent-setup.py --region us-east-1 --http-proxy http://your/proxy --https-proxy http://your/proxy --no-proxy 169.254.169.254
```

Untuk mempertahankan akses ke layanan metadata Amazon EC2 di instans EC2, gunakan `--no-proxy 169.254.169.254` (disarankan). Untuk informasi selengkapnya, lihat [Metadana Instans dan Data Pengguna](#) di Panduan Pengguna Amazon EC2.

Dalam nilai untuk `http-proxy` dan `https-proxy`, Anda menentukan seluruh URL.

b. Untuk instalasi agen CloudWatch Log yang ada, edit `/var/awslogs/etc/proxy.conf`, dan tambahkan proxy Anda:

```
HTTP_PROXY=  
HTTPS_PROXY=  
NO_PROXY=
```

2. Restart agen agar perubahan diterapkan:

```
sudo service awslogs restart
```

Jika Anda menggunakan Amazon Linux 2, gunakan perintah berikut untuk me-restart agen:

```
sudo service awslogsd restart
```

Membagi file konfigurasi agen CloudWatch Log

Jika Anda menggunakan `awslogs-agent-setup.py` versi 1.3.8 atau yang lebih baru dengan `awscli-cwlogs` 1.3.3 atau yang lebih baru, Anda dapat mengimpor konfigurasi aliran yang berbeda untuk berbagai komponen secara independen satu sama lain dengan membuat file konfigurasi tambahan di direktori `/var/awslogs/etc/config`. Ketika agen CloudWatch Log dimulai, itu mencakup konfigurasi aliran apa pun dalam file konfigurasi tambahan ini. Properti konfigurasi di bagian [umum] harus didefinisikan dalam file konfigurasi utama (`/var/awslogs/etc/awslogs.conf`) and are ignored in any additional configuration files found in `/var/awslogs/etc/config/`.

Jika Anda tidak memiliki direktori `/var/awslogs/etc/config/` karena Anda menginstal agen dengan rpm, Anda dapat menggunakan direktori `/etc/awslogs/config/` sebagai gantinya.

Restart agen agar perubahan diterapkan:

```
sudo service awslogs restart
```

Jika Anda menggunakan Amazon Linux 2, gunakan perintah berikut untuk me-restart agen:

```
sudo service awslogsd restart
```

CloudWatch FAQ agen log

Apa jenis rotasi file yang didukung?

Mekanisme rotasi file berikut didukung:

- Mengganti nama berkas log yang ada dengan akhiran numerik, kemudian membuat ulang berkas log kosong asli. Misalnya, `/var/log/syslog.log` is renamed `/var/log/syslog.log.1`. If `/var/log/syslog.log.1` already exists from a previous rotation, it is renamed `/var/log/syslog.log.2`.
- Memotong berkas log asli di tempat setelah membuat salinan. Misalnya, `/var/log/syslog.log` is copied to `/var/log/syslog.log.1` and `/var/log/syslog.log` terpotong. Mungkin akan ada kehilangan data untuk kasus ini, jadi berhati-hatilah dalam menggunakan mekanisme rotasi file ini.
- Membuat file baru dengan pola umum seperti yang lama. Misalnya, `/var/log/syslog.log.2014-01-01` remains and `/var/log/syslog.log.2014-01-02` dibuat.

Sidik jari (ID sumber) file dihitung dengan hashing kunci pengaliran log dan baris pertama dari konten file. Untuk menggantikan perilaku ini, pilihan `file_fingerprint_lines` dapat digunakan. Ketika rotasi file terjadi, file baru seharusnya memiliki konten baru dan file lama tidak seharusnya memiliki tambahan konten; agen mendorong file baru setelah selesai membaca file lama.

Bagaimana cara menentukan versi agen yang saya gunakan?

Jika Anda menggunakan skrip penyiapan untuk menginstal agen CloudWatch Logs, Anda dapat menggunakan `/var/awslogs/bin/awslogs-version.sh` untuk memeriksa versi agen yang Anda gunakan. Versi agen dan dependensi utamanya akan dicetak. Jika Anda menggunakan yum untuk menginstal agen CloudWatch Log, Anda dapat menggunakan “yum info awslogs” dan “yum info aws-cli-plugin-cloudwatch -logs” untuk memeriksa versi agen dan plugin Logs. CloudWatch

Bagaimana entri log dikonversi menjadi log acara?

Log acara berisi dua properti: stempel waktu ketika peristiwa terjadi, dan pesan log mentah. Secara default, semua baris yang dimulai dengan karakter yang bukan spasi kosong akan menutup pesan log sebelumnya, jika ada, dan memulai pesan log baru. Untuk menggantikan perilaku ini, `multi_line_start_pattern` dapat digunakan dan setiap baris yang cocok dengan pola akan memulai pesan log baru. Pola bisa berupa regex atau `{datetime_format}`. Sebagai contoh, jika baris pertama dari setiap pesan log berisi stempel waktu, seperti `'2014-01-02T13:13:01Z'`, `multi_line_start_pattern` dapat diatur ke `'\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}Z'`. Untuk menyederhanakan konfigurasi, variabel `{datetime_format}` dapat digunakan jika `datetime_format` option ditentukan. Untuk contoh yang sama, jika `datetime_format` diatur ke `'%Y-%m-%dT%H:%M:%S%z'`, `multi_line_start_pattern` dapat berupa `{datetime_format}`.

Waktu saat ini akan digunakan untuk setiap log acara jika `datetime_format` tidak disediakan. Jika `datetime_format` yang diberikan tidak valid untuk pesan log tertentu, stempel waktu dari log acara terakhir dengan stempel waktu yang berhasil diurai akan digunakan. Jika tidak ada log acara sebelumnya, waktu saat ini akan digunakan. Pesan peringatan akan dicatat ketika peristiwa log kembali ke waktu saat ini atau waktu log acara sebelumnya.

Stempel waktu digunakan untuk mengambil log acara dan menghasilkan metrik, jadi jika Anda menentukan format yang salah, log acara mungkin tidak bisa diambil dan akan menghasilkan metrik yang salah.

Bagaimana batch log acara dibuat?

Suatu batch akan menjadi penuh dan dipublikasikan ketika salah satu dari persyaratan berikut terpenuhi:

1. Parameter jumlah waktu `buffer_duration` telah berlalu sejak log acara pertama ditambahkan.
2. Kurang dari `batch_size` log acara telah terakumulasi, tetapi menambahkan log acara baru akan melampaui `batch_size`.
3. Jumlah log acara telah mencapai `batch_count`.
4. Log acara dari batch tidak berlangsung lebih dari 24 jam, tetapi menambahkan log acara baru akan melampaui batas 24 jam.

Apa yang menyebabkan entri log, log acara, atau batch dilewati atau dipotong?

Untuk mematuhi batasan operasi `PutLogEvents`, masalah berikut dapat menyebabkan log acara atau batch dilewati.

 Note

Agan CloudWatch Logs menulis peringatan ke lognya saat data dilewati.

1. Jika ukuran log acara melebihi 256 KB, log acara akan dilewati sepenuhnya.
2. Jika stempel waktu log acara menyatakan waktu yang lebih dari 2 jam mendatang, log acara akan dilewati.
3. Jika stempel waktu log acara menyatakan waktu yang lebih dari 14 hari yang lampau, log acara akan dilewati.
4. Jika log acara lebih tua dari periode retensi grup log, seluruh batch akan dilewati.
5. Jika batch log acara dalam satu permintaan `PutLogEvents` mencakup lebih dari 24 jam, operasi `PutLogEvents` akan gagal.

Apakah menghentikan agen akan menyebabkan kehilangan data/duplikat?

Tidak, selama file state tersedia dan tidak ada rotasi file yang terjadi sejak terakhir dijalankan. Agen CloudWatch Log dapat memulai dari tempat berhenti dan terus mendorong data log.

Dapatkah saya mengarahkan berkas log yang berbeda dari host yang sama atau berbeda ke pengaliran log yang sama?

Mengonfigurasi beberapa sumber log untuk mengirim data ke satu pengaliran log tidaklah didukung.

Panggilan API apa yang dilakukan agen (atau tindakan apa yang harus saya tambahkan ke kebijakan IAM saya)?

Agen CloudWatch Log memerlukan izin untuk melakukan `CreateLogGroup`, `CreateLogStream`, `DescribeLogStreams`, `DescribeLogGroupd`, `PutLogEvents` dan `PutRetentionPolicy` tindakan. Jika Anda menggunakan agen terbaru, `DescribeLogStreams` tidak diperlukan. Lihat contoh kebijakan IAM di bawah ini.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams",
        "logs:DescribeLogGroups",
        "logs:PutRetentionPolicy"
      ],
      "Resource": [
        "arn:aws:logs:*:*:*"
      ]
    }
  ]
}
```

Saya tidak ingin agen CloudWatch Log membuat grup log atau aliran log secara otomatis. Bagaimana cara mencegah agen membuat grup log dan pengaliran log?

Dalam kebijakan IAM, Anda dapat membatasi agen hanya ke operasi berikut:
`DescribeLogStreams`, `PutLogEvents`.

Sebelum Anda mencabut izin `CreateLogGroup` dan `CreateLogStream` dari agen, pastikan untuk membuat grup log dan pengaliran log yang Anda inginkan untuk digunakan oleh agen. Agen log tidak dapat membuat pengaliran log dalam grup log yang telah Anda buat kecuali memiliki izin `CreateLogGroup` dan `CreateLogStream`.

Log apa yang harus saya lihat saat memecahkan masalah?

Log penginstalan agen berada di `/var/log/awslogs-agent-setup.log` dan log agen berada di `/var/log/awslogs.log`.

Mulai Cepat: Gunakan CloudFormation untuk memulai dengan CloudWatch Log

AWS CloudFormation memungkinkan Anda untuk mendeskripsikan dan menyediakan AWS sumber daya Anda dalam format JSON. Keuntungan dari metode ini termasuk mampu mengelola kumpulan AWS sumber daya sebagai satu unit, dan dengan mudah mereplikasi AWS sumber daya Anda di seluruh Wilayah.

Saat Anda menyediakan AWS penggunaan CloudFormation, Anda membuat templat yang menjelaskan AWS sumber daya yang akan digunakan. Contoh berikut adalah cuplikan templat yang membuat grup log dan filter metrik yang menghitung 404 kemunculan dan mengirimkan jumlah ini ke grup log.

```
"WebServerLogGroup": {
  "Type": "AWS::Logs::LogGroup",
  "Properties": {
    "RetentionInDays": 7
  }
},

"404MetricFilter": {
  "Type": "AWS::Logs::MetricFilter",
  "Properties": {
    "LogGroupName": {
      "Ref": "WebServerLogGroup"
    },
    "FilterPattern": "[ip, identity, user_id, timestamp, request, status_code = 404, size, ...]",
    "MetricTransformations": [
      {
```

```
        "MetricValue": "1",
        "MetricNamespace": "test/404s",
        "MetricName": "test404Count"
      }
    ]
  }
}
```

Ini adalah contoh dasar. Anda dapat mengatur penerapan CloudWatch Log yang jauh lebih kaya menggunakan CloudFormation. Untuk informasi selengkapnya tentang contoh templat, lihat [Cuplikan Templat CloudWatch Log Amazon](#) di AWS CloudFormation Panduan Pengguna. Untuk informasi selengkapnya tentang memulai, lihat [Memulai AWS CloudFormation](#) dalam Panduan Pengguna AWS CloudFormation .

Log konsumsi melalui titik akhir HTTP

Amazon CloudWatch Logs menyediakan endpoint HTTP yang memungkinkan Anda mengirim log langsung ke CloudWatch Log menggunakan permintaan HTTP POST sederhana. Titik akhir ini mendukung otentikasi token sigV4 dan pembawa.

Important

Sebaiknya gunakan otentikasi SiGv4 untuk semua beban kerja produksi yang memungkinkan integrasi AWS SDK. SiGv4 menggunakan kredensial jangka pendek dan memberikan postur keamanan terkuat. Otentikasi token pembawa (kunci API) ditujukan untuk skenario di mana SigV4 tidak layak, seperti penerusan log pihak ketiga yang tidak mendukung integrasi SDK. AWS Untuk informasi selengkapnya, lihat [Alternatif untuk kunci akses jangka panjang](#) di Panduan Pengguna IAM.

CloudWatch Log mendukung titik akhir konsumsi HTTP berikut:

Titik akhir	Jalan	Content-Type	Format
OpenTelemetry Logs	/v1/logs	application/json atau application/x-protobuf	OTLP JSON atau Protobuf
HLC Logs	/services/collector/event	application/json	Format HLC
ND-JSON Logs	/ingest/bulk	application/json atau application/x-ndjson	JSON yang dibatasi baris baru
Structured JSON Logs	/ingest/json	application/json	JSON objek atau array

Perilaku umum

Semua titik akhir konsumsi HTTP berbagi perilaku berikut:

Autentikasi

Semua titik akhir mendukung otentikasi SIgV4 dan token pembawa:

- SiGv4 (disarankan) - AWS Penandatanganan Versi Tanda Tangan Standar 4. Gunakan SiGv4 setiap kali aplikasi atau infrastruktur Anda mendukung AWS SDK atau dapat menandatangani permintaan. SiGv4 menggunakan kredensial jangka pendek dan merupakan metode otentikasi yang paling aman.
- Token pembawa — Gunakan `Authorization: Bearer <ACWL token>` header.
 - Token harus berupa token pembawa ACWL yang valid. Untuk petunjuk penyiapan, lihat [the section called “Otentikasi token pembawa”](#).
 - Memerlukan izin `logs:PutLogEvents` dan `logs:CallWithBearerToken` IAM.

Grup log dan aliran log

- Disediakan melalui header: `x-aws-log-group` dan `x-aws-log-stream`
- Parameter kueri juga `?logGroup=<name>&logStream=<name>` didukung pada semua titik akhir kecuali OTLP.
- Anda tidak dapat menggunakan parameter kueri dan header untuk parameter yang sama.
- Baik grup log dan aliran log diperlukan.

Respons

- Sukses: HTTP 200 dengan tubuh `{}`
- Kesalahan validasi: HTTP 400
- Kegagalan autentikasi: HTTP 401

Menyiapkan otentikasi token pembawa

Sebelum Anda dapat mengirim log menggunakan otentikasi token pembawa dengan salah satu titik akhir penyerapan HTTP, Anda perlu:

- Buat pengguna IAM dengan izin CloudWatch Log
- Hasilkan kredensial khusus layanan (token pembawa)
- Buat grup log dan aliran log

- Aktifkan otentikasi token pembawa pada grup log

Important

Sebaiknya gunakan otentikasi SiGv4 dengan kredensi jangka pendek untuk semua beban kerja jika memungkinkan. SiGv4 memberikan postur keamanan terkuat. Batasi penggunaan kunci API (token pembawa) ke skenario di mana otentikasi berbasis kredensial jangka pendek tidak layak dilakukan. Ketika Anda siap untuk memasukkan CloudWatch Log ke dalam aplikasi dengan persyaratan keamanan yang lebih besar, Anda harus beralih ke kredensi jangka pendek. Untuk informasi selengkapnya, lihat [Alternatif untuk kunci akses jangka panjang](#) di Panduan Pengguna IAM.

Opsi 1: Mulai cepat menggunakan AWS konsol

AWS Management Console menyediakan alur kerja yang efisien untuk menghasilkan kunci API untuk akses titik akhir HTTP.

Untuk mengatur akses titik akhir HTTP menggunakan konsol

1. Masuk ke Konsol AWS Manajemen.
2. Arahkan CloudWatch ke > Pengaturan > Log.
3. Di bagian API Keys, pilih Generate API key.
4. Untuk kedaluwarsa kunci API, lakukan salah satu hal berikut:
 - Pilih durasi kedaluwarsa kunci API 1, 5, 30, 90, atau 365 hari.
 - Pilih Durasi kustom untuk menentukan tanggal kedaluwarsa kunci API kustom.
 - Pilih Jangan pernah kedaluwarsa (tidak disarankan).
5. Pilih Hasilkan kunci API.

Konsol secara otomatis:

- Membuat pengguna IAM baru dengan izin yang sesuai
- Melampirkan kebijakan terkelola [CloudWatchLogsAPIKeyAccess](#) (termasuk `logs:PutLogEvents` dan `logs:CallWithBearerToken` izin)
- Menghasilkan kredensi khusus layanan (kunci API)

6. Salin dan simpan kredensial yang ditampilkan dengan aman:

- ID Kunci API (ID kredensi khusus layanan)
- Rahasia Kunci API (Token pembawa)

Important

Simpan Rahasia Kunci API segera. Hal itu tidak dapat diambil lagi nanti. Jika Anda kehilangannya, Anda harus membuat kunci API baru.

7. Buat grup log dan aliran log tempat log Anda akan disimpan:

```
# Create the log group
aws logs create-log-group \
  --log-group-name /aws/hlc-logs/my-application \
  --region us-east-1

# Create the log stream
aws logs create-log-stream \
  --log-group-name /aws/hlc-logs/my-application \
  --log-stream-name application-stream-001 \
  --region us-east-1
```

8. Aktifkan otentikasi token pembawa pada grup log:

```
aws logs put-bearer-token-authentication \
  --log-group-identifier /aws/hlc-logs/my-application \
  --bearer-token-authentication-enabled \
  --region us-east-1
```

Verifikasi konfigurasi:

```
aws logs describe-log-groups \
  --log-group-name-prefix /aws/hlc-logs/my-application \
  --region us-east-1
```

Izin disertakan: Pengguna IAM yang dibuat secara otomatis akan memiliki izin berikut:

- `logs:PutLogEvents`— Kirim peristiwa log ke CloudWatch Log

- `logs:CallWithBearerToken`— Otentikasi menggunakan token pembawa
- `kms:Describe*`, `kms:GenerateDataKey*`, `kms:Decrypt` — Akses grup log terenkripsi KMS (dengan kondisi membatasi layanan log)

Opsi 2: Pengaturan manual

Jika Anda lebih suka kontrol lebih atas konfigurasi IAM atau perlu menyesuaikan izin, Anda dapat mengatur akses titik akhir HTTP secara manual.

Langkah 1: Buat pengguna IAM

Buat pengguna IAM yang akan digunakan untuk log ingestion:

1. Masuk ke AWS Management Console dan navigasikan ke IAM.
2. Pada panel navigasi kiri, pilih Pengguna.
3. Pilih Create user (Buat pengguna).
4. Masukkan nama pengguna (misalnya, `cloudwatch-logs-hlc-user`).
5. Pilih Berikutnya.
6. Lampirkan salah satu kebijakan IAM berikut:

Opsi A: Gunakan kebijakan terkelola (disarankan)

Lampirkan kebijakan terkelola [CloudWatchLogsAPIKeyAccess](#).

Opsi B: Buat kebijakan khusus

Buat dan lampirkan kebijakan IAM berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "LogsAPIs",
      "Effect": "Allow",
      "Action": [
        "logs:CallWithBearerToken",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

```

    },
    {
      "Sid": "KMSAPIs",
      "Effect": "Allow",
      "Action": [
        "kms:Describe*",
        "kms:GenerateDataKey*",
        "kms:Decrypt"
      ],
      "Condition": {
        "StringEquals": {
          "kms:ViaService": [
            "logs.*.amazonaws.com"
          ]
        }
      },
      "Resource": "arn:aws:kms:*:*:key/*"
    }
  ]
}

```

7. Pilih Berikutnya dan kemudian Buat pengguna.

Note

Izin KMS diperlukan jika Anda berencana untuk mengirim log ke grup log terenkripsi KMS. Kondisi ini membatasi akses KMS ke hanya kunci yang digunakan melalui layanan CloudWatch Log.

Langkah 2: Hasilkan kredensial khusus layanan (kunci API)

Buat kunci API CloudWatch Log menggunakan [CreateServiceSpecificCredential](#) API. Anda juga dapat menggunakan perintah [create-service-specific-credential](#) CLI. Untuk usia kredensial, Anda dapat menentukan nilai antara 1—36600 hari. Jika Anda tidak menentukan usia kredensi, kunci API tidak akan kedaluwarsa.

Untuk membuat kunci API dengan masa kedaluwarsa 30 hari:

```

aws iam create-service-specific-credential \
  --user-name cloudwatch-logs-hlc-user \

```

```
--service-name logs.amazonaws.com \  
--credential-age-days 30
```

Responsnya adalah [ServiceSpecificCredential](#) objek. `ServiceCredentialSecret` nilainya adalah kunci API CloudWatch Log Anda (token pembawa).

Important

Simpan `ServiceCredentialSecret` nilainya dengan aman, karena Anda tidak dapat mengambilnya nanti. Jika Anda kehilangannya, Anda harus membuat kunci API baru.

Langkah 3: Buat grup log dan aliran log

Buat grup log dan aliran log tempat log Anda akan disimpan:

```
# Create the log group  
aws logs create-log-group \  
  --log-group-name /aws/hlc-logs/my-application \  
  --region us-east-1  
  
# Create the log stream  
aws logs create-log-stream \  
  --log-group-name /aws/hlc-logs/my-application \  
  --log-stream-name application-stream-001 \  
  --region us-east-1
```

Langkah 4: Aktifkan otentikasi token pembawa

Aktifkan otentikasi token pembawa pada grup log:

```
aws logs put-bearer-token-authentication \  
  --log-group-identifier /aws/hlc-logs/my-application \  
  --bearer-token-authentication-enabled \  
  --region us-east-1
```

Verifikasi konfigurasi:

```
aws logs describe-log-groups \  
  --log-group-name-prefix /aws/hlc-logs/my-application \  
  --region us-east-1
```

Izin kontrol untuk membuat dan menggunakan kunci API CloudWatch Log

Pembuatan dan penggunaan kunci API CloudWatch Log dikendalikan oleh tindakan dan kunci kondisi di layanan CloudWatch Log dan IAM.

Mengontrol pembuatan kunci API CloudWatch Log

CreateServiceSpecificCredentialTindakan [iam:](#) mengontrol pembuatan kunci khusus layanan (seperti kunci API CloudWatch Log). Anda dapat memberikan cakupan tindakan ini kepada pengguna IAM sebagai sumber daya untuk membatasi pengguna yang kuncinya dapat dibuat.

Anda dapat menggunakan tombol kondisi berikut untuk memaksakan kondisi pada izin untuk `iam:CreateServiceSpecificCredential` tindakan:

- [iam: ServiceSpecificCredentialAgeDays](#) — Memungkinkan Anda menentukan, dalam kondisi, waktu kedaluwarsa kunci dalam beberapa hari. Misalnya, Anda dapat menggunakan kunci kondisi ini untuk hanya mengizinkan pembuatan kunci API yang kedaluwarsa dalam 90 hari.
- [iam: ServiceSpecificCredentialServiceName](#) — Memungkinkan Anda menentukan, dalam kondisi, nama layanan. Misalnya, Anda dapat menggunakan kunci kondisi ini untuk hanya mengizinkan pembuatan kunci API untuk CloudWatch Log dan bukan layanan lainnya.

Mengontrol penggunaan kunci API CloudWatch Log

`logs:CallWithBearerToken`Tindakan ini mengontrol penggunaan kunci API CloudWatch Log. Untuk mencegah identitas menggunakan kunci API CloudWatch Logs, lampirkan kebijakan yang menolak `logs:CallWithBearerToken` tindakan tersebut ke pengguna IAM yang terkait dengan kunci tersebut.

Contoh kebijakan

Mencegah identitas membuat dan menggunakan kunci API CloudWatch Log

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyCWAPIKeys",
      "Effect": "Deny",
      "Action": [
        "iam:CreateServiceSpecificCredential",
```

```

        "logs:CallWithBearerToken"
      ],
      "Resource": "*"
    }
  ]
}

```

Warning

Kebijakan ini akan mencegah pembuatan kredensi untuk semua AWS layanan yang mendukung pembuatan kredensi khusus layanan. Untuk informasi selengkapnya, lihat [Kredensi khusus layanan](#) untuk pengguna IAM.

Mencegah identitas menggunakan kunci API CloudWatch Log

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "logs:CallWithBearerToken",
      "Resource": "*"
    }
  ]
}

```

Izinkan pembuatan kunci CloudWatch Log hanya jika mereka kedaluwarsa dalam 90 hari

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceSpecificCredential",
      "Resource": "arn:aws:iam::123456789012:user/username",
      "Condition": {
        "StringEquals": {
          "iam:ServiceSpecificCredentialServiceName": "logs.amazonaws.com"
        },
        "NumericLessThanEquals": {
          "iam:ServiceSpecificCredentialAgeDays": "90"
        }
      }
    }
  ]
}

```

```
}  
  }  
    }  
      ]  
        }
```

Memutar kunci API

Memutar kunci API Anda secara teratur mengurangi risiko akses yang tidak sah. Sebaiknya buat jadwal rotasi yang sesuai dengan kebijakan keamanan organisasi Anda.

Proses rotasi

Untuk memutar kunci API tanpa mengganggu pengiriman log, ikuti prosedur ini:

1. Buat kredensi baru (sekunder) untuk pengguna IAM:

```
aws iam create-service-specific-credential \  
  --user-name cloudwatch-logs-hlc-user \  
  --service-name logs.amazonaws.com \  
  --credential-age-days 90
```

2. (Opsional) Simpan kredensi baru AWS Secrets Manager untuk pengambilan aman dan rotasi otomatis.
3. Impor kredensi baru ke portal vendor Anda atau perbarui konfigurasi aplikasi Anda untuk menggunakan kunci API baru.
4. Setel kredensi asli menjadi tidak aktif:

```
aws iam update-service-specific-credential \  
  --user-name cloudwatch-logs-hlc-user \  
  --service-specific-credential-id ACCA1234EXAMPLE1234 \  
  --status Inactive
```

5. Verifikasi bahwa pengiriman log tidak terpengaruh dengan memantau IncomingBytes metrik untuk grup log Anda. CloudWatch Untuk informasi selengkapnya, lihat [Memantau dengan CloudWatch metrik](#).
6. Setelah mengonfirmasi pengiriman yang berhasil dengan kunci baru, hapus kredensi sebelumnya:

```
aws iam delete-service-specific-credential \  

```

```
--service-specific-credential-id ACCA1234EXAMPLE1234
```

Memantau kedaluwarsa kunci

Untuk memeriksa tanggal pembuatan dan status kunci API yang ada, gunakan [list-service-specific-credentials](#) perintah:

```
aws iam list-service-specific-credentials \  
  --user-name cloudwatch-logs-hlc-user \  
  --service-name logs.amazonaws.com
```

Tanggapan tersebut mencakup CreateDate dan Status untuk setiap kredensi. Gunakan informasi ini untuk mengidentifikasi kunci yang mendekati kedaluwarsa atau telah aktif lebih lama dari yang diizinkan oleh kebijakan rotasi Anda.

Menanggapi kunci API yang disusupi

Jika Anda mencurigai bahwa kunci API telah disusupi, segera lakukan langkah-langkah berikut:

1. Nonaktifkan kunci segera untuk mencegah penggunaan tidak sah lebih lanjut:

```
aws iam update-service-specific-credential \  
  --user-name cloudwatch-logs-hlc-user \  
  --service-specific-credential-id ACCA1234EXAMPLE1234 \  
  --status Inactive
```

2. Tinjau CloudTrail log untuk menentukan ruang lingkup akses yang tidak sah. Lihat [the section called “Pencatatan penggunaan kunci API dengan CloudTrail”](#) cara mengaktifkan audit penggunaan kunci API.
3. Buat kunci pengganti mengikuti proses rotasi yang dijelaskan dalam [the section called “Proses rotasi”](#).
4. Hapus kunci yang disusupi setelah penggantian dilakukan:

```
aws iam delete-service-specific-credential \  
  --service-specific-credential-id ACCA1234EXAMPLE1234
```

5. Lampirkan kebijakan penolakan jika Anda perlu segera memblokir semua akses token pembawa untuk pengguna IAM saat Anda menyelidiki:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "logs:CallWithBearerToken",
    "Resource": "*"
  }
}
```

Note

Untuk melakukan tindakan ini melalui API, Anda harus mengautentikasi dengan AWS kredensi dan bukan dengan kunci API CloudWatch Log.

Anda juga dapat menggunakan operasi IAM API berikut untuk mengelola kunci yang dikompromikan:

- [ResetServiceSpecificCredential](#)— Setel ulang kunci untuk menghasilkan kata sandi baru tanpa menghapus kredensialnya. Kuncinya tidak boleh kedaluwarsa.

Praktik keamanan terbaik untuk kunci API

Ikuti praktik terbaik berikut ini untuk melindungi kunci API CloudWatch Log Anda:

- Jangan pernah menyematkan kunci API dalam kode sumber. Jangan kode keras kunci API dalam kode aplikasi atau komit ke sistem kontrol versi. Jika kunci secara tidak sengaja berkomitmen ke repositori publik, pemindaian AWS otomatis dapat menandainya dan Anda harus segera memutar kunci tersebut.
- Gunakan manajer rahasia. Simpan kunci API di [AWS Secrets Manager](#) atau solusi manajemen rahasia yang setara. Ini memungkinkan kontrol akses terpusat, pencatatan audit, dan rotasi otomatis.
- Tetapkan kedaluwarsa pada semua tombol. Selalu tentukan `--credential-age-days` nilai saat membuat kunci API. Untuk menerapkan masa pakai kunci maksimum di seluruh organisasi Anda, gunakan kunci kondisi `iam:ServiceSpecificCredentialAgeDays` IAM. Sebagai contoh, lihat [the section called “Izinkan pembuatan kunci CloudWatch Log hanya jika mereka kedaluwarsa dalam 90 hari”](#).

- Terapkan izin hak istimewa paling sedikit. Cakupan izin pengguna IAM hanya untuk grup log dan tindakan yang diperlukan. Gunakan kebijakan [CloudWatchLogsAPIKeyAkses](#) terkelola sebagai titik awal dan batasi lebih lanjut sesuai kebutuhan.
- Aktifkan CloudTrail logging. Audit penggunaan kunci API dengan mengaktifkan peristiwa CloudTrail data untuk `AWS::Logs::LogGroupAuthorization`. Lihat [the section called “Pencatatan penggunaan kunci API dengan CloudTrail”](#).
- Monitor dengan IAM Access Analyzer. Gunakan [IAM Access Analyzer](#) untuk mengidentifikasi kredensi yang tidak digunakan dan kebijakan yang terlalu permisif yang terkait dengan pengguna IAM kunci API Anda.
- Putar tombol secara teratur. Tetapkan jadwal rotasi dan ikuti proses yang dijelaskan dalam [the section called “Memutar kunci API”](#).

Pencatatan penggunaan kunci API dengan CloudTrail

Anda dapat menggunakan AWS CloudTrail untuk mencatat peristiwa data untuk penggunaan kunci API CloudWatch Log. CloudWatch Log memancarkan peristiwa `AWS::Logs::LogGroupAuthorization` data untuk `CallWithBearerToken` panggilan, memungkinkan Anda untuk mengaudit kapan dan bagaimana kunci API digunakan untuk mengirim log.

Untuk mengaktifkan CloudTrail logging untuk penggunaan kunci API CloudWatch Log:

Note

Bucket S3 yang Anda tentukan untuk jejak harus memiliki kebijakan bucket yang memungkinkan CloudTrail untuk menulis file log ke dalamnya. Untuk informasi selengkapnya, lihat [kebijakan bucket Amazon S3](#) untuk CloudTrail

1. Buat jejak:

```
aws cloudtrail create-trail \  
  --name cloudwatch-logs-api-key-audit \  
  --s3-bucket-name my-cloudtrail-bucket \  
  --region us-east-1
```

2. Konfigurasikan pemilih acara lanjutan untuk menangkap peristiwa otorisasi grup CloudWatch log Log:

```
aws cloudtrail put-event-selectors \  
  --region us-east-1 \  
  --trail-name cloudwatch-logs-api-key-audit \  
  --advanced-event-selectors '[{  
    "Name": "CloudWatch Logs API key authorization events",  
    "FieldSelectors": [  
      { "Field": "eventCategory", "Equals": ["Data"] },  
      { "Field": "resources.type", "Equals":  
        ["AWS::Logs::LogGroupAuthorization"] }  
    ]  
  }]]'
```

3. Mulai pencatatan jejak:

```
aws cloudtrail start-logging \  
  --name cloudwatch-logs-api-key-audit \  
  --region us-east-1
```

Mengirim log menggunakan titik akhir OTLP (Log) OpenTelemetry

Endpoint Logs (/v1/logs) menerima data OpenTelemetry log OpenTelemetry Protocol (OTLP) baik dalam pengkodean JSON atau Protobuf. Untuk informasi rinci tentang titik akhir OTLP, termasuk konfigurasi dan penggunaan, lihat [Mengirim metrik dan jejak ke with. CloudWatch OpenTelemetry](#)

Jika Anda menggunakan otentikasi token pembawa, selesaikan langkah-langkah penyiapan [the section called “Otentikasi token pembawa”](#) sebelum melanjutkan.

Format permintaan

- Metode: POST
- Tipe Konten: `application/json` atau `application/x-protobuf`
- Grup log: hanya `x-aws-log-group` header (parameter kueri tidak didukung)
- Aliran log: `x-aws-log-stream` header

Contoh permintaan

```
curl -X POST "https://logs.<region>.amazonaws.com/v1/logs" \  

```

```

-H "Authorization: Bearer ACWL<token>" \
-H "Content-Type: application/json" \
-H "x-aws-log-group: MyLogGroup" \
-H "x-aws-log-stream: MyLogStream" \
-d '{
  "resourceLogs": [
    {
      "resource": {
        "attributes": [
          {
            "key": "service.name",
            "value": { "stringValue": "my-service" }
          }
        ]
      },
      "scopeLogs": [
        {
          "scope": {
            "name": "my-library",
            "version": "1.0.0"
          },
          "logRecords": [
            {
              "timeUnixNano": "1741900000000000000",
              "severityNumber": 9,
              "severityText": "INFO",
              "body": {
                "stringValue": "User logged in successfully"
              },
              "attributes": [
                {
                  "key": "user.id",
                  "value": { "stringValue": "12345" }
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}'

```

Respons

Sukses (semua acara diterima):

```
HTTP 200 OK
{}
```

Keberhasilan sebagian (beberapa acara ditolak):

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1, \"expiredLogEventCount\": 1}"
  }
}
```

Ketika permintaan Content-Type adalah `application/x-protobuf`, respon dikembalikan sebagai pesan `ExportLogsServiceResponse` protobuf serial dengan bidang yang sama.

Perilaku khusus OTLP

Perilaku berikut khusus untuk titik akhir OTLP dan tidak ada pada titik akhir konsumsi HTTP lainnya:

- Retry-After header - Termasuk pada 503 dan 429 tanggapan untuk menunjukkan kapan klien harus mencoba lagi.

Mengirim log menggunakan titik akhir HLC (Log HLC)

Titik akhir HLC Logs (`/services/collector/event`) didasarkan pada format HTTP Log Collector (HLC).

Jika Anda menggunakan otentikasi token pembawa, selesaikan langkah-langkah penyiapan [the section called “Otentikasi token pembawa”](#) sebelum melanjutkan.

Mode masukan

Setiap peristiwa adalah objek JSON dengan "event" bidang wajib. Bidang metadata opsional: "time", "host", "source", "sourcetype". "index"

Acara tunggal:

```
{"event":"Hello world!","time":1486683865.0}
```

JSON array acara:

```
[
  {"event":"msg1","time":1486683865.0},
  {"event":"msg2","time":1486683866.0}
]
```

Acara bergabung/batch (tidak ada pembungkus array):

```
{"event":"msg1","time":1486683865.0}{"event":"msg2","time":1486683866.0}
```

Bidang acara (wajib)

"event" Bidang itu wajib. Nilainya dapat berupa jenis JSON:

```
{"event":"a string message"}
{"event":{"message":"structured data","severity":"INFO"}}
{"event":42}
{"event":true}
```

Objek tanpa "event" bidang dilewati secara diam-diam:

```
{"message":"this is skipped – no event field"}
```

Bidang waktu (opsional)

"time" Bidang ini dalam detik epoch (bukan milidetik), dengan desimal opsional untuk presisi sub-detik.

format	Contoh	Ditafsirkan sebagai
Desimal	"time":1486683865.500	1486683865500 ms

format	Contoh	Ditafsirkan sebagai
Bilangan Bulat	"time":1486683865	1486683865000 ms
String (mengapung)	"time":"1486683865.500"	1486683865500 ms
String (bilangan bulat)	"time":"1486683865"	1486683865000 ms
Hilang	(tidak ada bidang waktu)	Waktu server saat ini
Tidak valid	"time":"invalid"	Waktu server saat ini

Content-Type

Hanya `application/json` diterima.

Jenis nilai JSON yang diterima

Tipe tingkat atas	Perilaku
Objek dengan "event"	Diterima
Objek tanpa "event"	Dilewati
Array objek	Setiap elemen diproses secara individual
Objek yang digabungkan	Setiap objek diproses secara individual
Primitif (string, angka, boolean, null)	Dilewati

Format titik akhir

URL titik akhir HLC mengikuti format ini:

```
https://logs.<region>.amazonaws.com/services/collector/event?
logGroup=<name>&logStream=<name>[&entityName=<name>&entityEnvironment=<environment>]
```

Parameter yang diperlukan:

- `<region>`— AWS Wilayah (misalnya, `us-east-1`, `eu-west-1`)
- `logGroup`— Nama grup log yang dikodekan URL
- `logStream`— Nama aliran log yang dikodekan URL

Parameter opsional:

Anda dapat secara opsional mengaitkan peristiwa log Anda dengan Service entitas dengan menyertakan parameter kueri berikut. Karena log yang dikirim melalui titik akhir HLC adalah telemetri khusus, mereka tidak secara otomatis terkait dengan entitas. Dengan menyediakan parameter ini, CloudWatch Log membuat entitas dengan `KeyAttributes.Type` set ke Service dan mengaitkannya dengan peristiwa log Anda. Ini memungkinkan fitur terkait Jelajahi CloudWatch untuk menghubungkan log ini dengan telemetri lain (metrik, jejak, dan log) dari layanan yang sama, sehingga memudahkan pemecahan masalah dan memantau aplikasi Anda di berbagai jenis sinyal. Untuk informasi selengkapnya tentang entitas dan telemetri terkait, lihat [Menambahkan informasi terkait ke telemetri kustom](#).

- `entityName`— Nama entitas layanan untuk dikaitkan dengan peristiwa log. Nilai ini disimpan sebagai entitas `KeyAttributes.Name` (misalnya, `my-application` atau `api.myservice.com`).
- `entityEnvironment`— Lingkungan tempat layanan di-host atau milik apa. Nilai ini disimpan sebagai entitas `KeyAttributes.Environment` (misalnya, `productionec2:default`, atau `eks:my-cluster/default`).

Format permintaan

Kirim log menggunakan HTTP POST dengan header dan badan berikut:

Header:

- `Authorization: Bearer <your-bearer-token>`
- `Content-Type: application/json`

Format tubuh:

Badan permintaan harus dalam format JSON dengan array peristiwa:

```
{
  "event": [
    {
      "time": 1730141374.001,
      "event": "Application started successfully",
      "host": "web-server-1",
      "source": "application.log",
      "severity": "info"
    },
    {
      "time": 1730141374.457,
      "event": "User login successful",
      "host": "web-server-1",
      "source": "auth.log",
      "user": "john.doe"
    }
  ]
}
```

Deskripsi bidang:

- **time**— Stempel waktu epoch Unix dalam hitungan detik, dengan desimal opsional untuk presisi sub-detik (opsional)
- **event**— Pesan log atau data peristiwa (wajib)
- **host**— Nama host sumber atau pengenalan (opsional)
- **source**— Pengidentifikasi sumber log (opsional)

Bidang kustom tambahan dapat dimasukkan sesuai kebutuhan.

Contoh permintaan

```
curl -X POST "https://logs.<region>.amazonaws.com/services/collector/event?
logGroup=MyLogGroup&logStream=MyStream" \
-H "Authorization: Bearer ACWL<token>" \
-H "Content-Type: application/json" \
-d '{"event":{"message":"User logged
in","user_id":"u-123"},"time":1486683865.0,"host":"web-01","source":"auth-service"}'
```


Praktik terbaik

Acara batching

Untuk kinerja dan efisiensi yang lebih baik:

- Batch beberapa peristiwa dalam satu permintaan bila memungkinkan
- Ukuran batch yang disarankan: 10—100 acara per permintaan
- Ukuran permintaan maksimal: 1 MB

Penanganan kesalahan

Menerapkan penanganan kesalahan yang tepat dalam aplikasi Anda. Kode status HTTP umum:

- 200 OK— Log berhasil dicerna
- 400 Bad Request— Format atau parameter permintaan tidak valid
- 401 Unauthorized— Token pembawa tidak valid atau kedaluwarsa
- 403 Forbidden— Izin tidak mencukupi
- 404 Not Found— Grup log atau aliran tidak ada
- 429 Too Many Requests— Batas tarif terlampaui
- 500 Internal Server Error— Kesalahan layanan (coba lagi dengan backoff eksponensial)

Batasan

- Ukuran acara maksimal: 256 KB per acara
- Ukuran permintaan maksimal: 1 MB
- Maksimum acara per permintaan: 10.000
- Nama grup log harus mengikuti konvensi penamaan CloudWatch Log
- Otentikasi token pembawa harus diaktifkan pada grup log jika otentikasi token pembawa digunakan.

Mengirim log menggunakan titik akhir NDJSON (ND-JSON Logs)

Titik akhir ND-JSON Logs (/ingest/bulk) menerima log dalam format [NDJSON \(Newline Delimited JSON\)](#). Setiap baris berisi persis satu nilai JSON, dipisahkan oleh karakter baris baru.

Jika Anda menggunakan otentikasi token pembawa, selesaikan langkah-langkah penyiapan [the section called “Otentikasi token pembawa”](#) sebelum melanjutkan.

Format permintaan

Kirim satu nilai JSON per baris, dipisahkan oleh \n (LF) atau \r\n (CRLF). Baris kosong diabaikan secara diam-diam.

```
{"timestamp":1771007942000,"message":"event one","level":"INFO"}
{"timestamp":1771007943000,"message":"event two","level":"ERROR"}
{"timestamp":1771007944000,"message":"event three","level":"DEBUG"}
```

Keduanya application/json dan application/x-ndjson diterima sebagai Content-Type.

Jenis nilai JSON yang diterima

Per spesifikasi NDJSON (RFC 8259), nilai JSON yang valid diterima pada setiap baris.

Objek JSON (paling umum):

```
{"timestamp":1771007942000,"message":"User logged in","service":"auth"}
{"timestamp":1771007943000,"error":"Connection timeout","service":"api"}
```

Array JSON (diratakan menjadi peristiwa individual):

```
[{"timestamp":1000,"message":"a"}, {"timestamp":2000,"message":"b"}]
```

Baris tunggal ini menghasilkan 2 peristiwa. Setiap elemen array menjadi peristiwa log terpisah.

Nilai primitif:

```
"a plain string log message"
42
true
null
```

Setiap primitif menjadi acaranya sendiri dengan stempel waktu server saat ini.

Jenis campuran:

```
{ "timestamp": 1771007942000, "message": "structured event" }
"unstructured string message"
42
{ "timestamp": 1771007943000, "error": "something failed" }
```

Semua 4 baris diterima sebagai acara yang valid.

Konten baris	Perilaku
Objek JSON	Diterima, stempel waktu diekstraksi jika ada
Array JSON	Diratakan - setiap elemen menjadi acara yang terpisah
Array kosong []	Diterima, menghasilkan 0 acara
String JSON	Diterima sebagai pesan acara
Nomor JSON	Diterima sebagai pesan acara
Boolean JSON	Diterima sebagai pesan acara
JSON nol	Diterima sebagai pesan acara
JSON tidak valid	Dilewati (dihitung, pemrosesan berlanjut)
Baris kosong	Diabaikan (tidak dihitung sebagai dilewati)

Bidang stempel waktu

"timestamp" Bidang ini dalam milidetik epoch (bukan detik).

format	Contoh	Ditafsirkan sebagai
Numerik (millis)	"timestamp":1771007942000	1771007942000 ms
Hilang	(tidak ada bidang stempel waktu)	Waktu server saat ini
Non-numerik	"timestamp":"invalid"	Waktu server saat ini
Garis non-objek	"hello", 42, true	Waktu server saat ini

Baris tidak valid

Baris yang tidak valid JSON dilewati dan dihitung secara diam-diam. Pemrosesan berlanjut dengan baris berikutnya.

```
{ "message": "valid event" }
this is not valid json
{ "message": "another valid event" }
```

Hasil: 2 acara tertelan, 1 dilewati. Pengembalian HTTP 200.

Jika semua baris tidak valid, kembali HTTP 400 dengan. "All events were invalid"

Contoh permintaan

```
curl -X POST "https://logs.<region>.amazonaws.com/ingest/bulk?
logGroup=MyLogGroup&logStream=MyStream" \
-H "Authorization: Bearer ACWL<token>" \
-H "Content-Type: application/x-ndjson" \
-d '{ "timestamp":1771007942000,"message":"User logged in","level":"INFO"}
{ "timestamp":1771007943000,"message":"Query took 42ms","level":"DEBUG"}
{ "timestamp":1771007944000,"error":"Connection refused","level":"ERROR"}'
```

Respons

Sukses (semua acara diterima):

```
HTTP 200 OK
```

```
{}
```

Keberhasilan sebagian (beberapa acara ditolak):

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1, \"expiredLogEventCount\": 1}"
  }
}
```

`rejectedLogRecordsBidang` adalah jumlah total peristiwa yang ditolak. `errorMessageBidang` berisi rincian yang dikodekan JSON dengan alasan penolakan:

- `tooOldLogEventCount`— Acara dengan stempel waktu yang lebih tua dari periode retensi
- `tooNewLogEventCount`— Acara dengan stempel waktu terlalu jauh di masa depan
- `expiredLogEventCount`— Acara yang kedaluwarsa selama pemrosesan

Praktik terbaik

Acara batching

Untuk kinerja dan efisiensi yang lebih baik:

- Batch beberapa peristiwa dalam satu permintaan bila memungkinkan
- Ukuran batch yang disarankan: 10—100 acara per permintaan
- Ukuran permintaan maksimal: 1 MB

Penanganan kesalahan

Menerapkan penanganan kesalahan yang tepat dalam aplikasi Anda. Kode status HTTP umum:

- `200 OK`— Log berhasil dicerna
- `400 Bad Request`— Format atau parameter permintaan tidak valid
- `401 Unauthorized`— Token pembawa tidak valid atau kedaluwarsa
- `403 Forbidden`— Izin tidak mencukupi

- 404 Not Found— Grup log atau aliran tidak ada
- 429 Too Many Requests— Batas tarif terlampaui
- 500 Internal Server Error— Kesalahan layanan (coba lagi dengan backoff eksponensial)

Batasan

- Ukuran acara maksimal: 256 KB per acara
- Ukuran permintaan maksimal: 1 MB
- Maksimum acara per permintaan: 10.000
- Nama grup log harus mengikuti konvensi penamaan CloudWatch Log
- Otentikasi token pembawa harus diaktifkan pada grup log jika otentikasi token pembawa digunakan.

Mengirim log menggunakan titik akhir JSON Terstruktur (Log JSON Terstruktur)

Titik akhir JSON Logs Terstruktur (/ingest/json) menerima JSON standar — baik objek JSON tunggal atau array objek JSON. Endpoint ini dirancang untuk data log terstruktur di mana setiap peristiwa adalah objek JSON.

Jika Anda menggunakan otentikasi token pembawa, selesaikan langkah-langkah penyiapan [the section called “Otentikasi token pembawa”](#) sebelum melanjutkan.

Format permintaan

Hanya application/json diterima sebagai Content-Type.

Objek JSON tunggal:

```
{"timestamp":1771007942000,"message":"single event","level":"INFO"}
```

JSON array objek:

```
[  
  {"timestamp":1771007942000,"message":"event one","level":"INFO"},  
  ...  
]
```

```
{ "timestamp":1771007943000,"message":"event two","level":"ERROR"}
]
```

Jenis nilai JSON yang diterima

Titik akhir ini ketat — hanya objek JSON yang diterima sebagai peristiwa.

Input	Perilaku
Objek JSON tunggal	Diterima sebagai salah satu acara
JSON array objek	Setiap objek menjadi peristiwa yang terpisah
Array kosong []	Diterima, menghasilkan 0 acara
Non-objek dalam array (string, angka, dll.)	Dilewati
Primitif tingkat atas ("hello",) 42	Dilewati
Objek yang digabungkan {...}{...}	Hanya objek pertama yang diuraikan

Contoh - array dengan tipe campuran:

```
[
  {"timestamp":1771007942000,"message":"valid object"},
  "just a string",
  42,
  {"timestamp":1771007943000,"message":"another valid object"}
]
```

Hasil: 2 peristiwa tertelan (objek), 2 dilewati (string dan angka).

Bidang stempel waktu

"timestamp" Bidang ini dalam milidetik epoch, sama seperti titik akhir NDJSON.

format	Contoh	Ditafsirkan sebagai
Numerik (millis)	"timestamp":1771007942000	1771007942000 ms
Hilang	(tidak ada bidang stempel waktu)	Waktu server saat ini
Non-numerik	"timestamp":"invalid"	Waktu server saat ini

Contoh permintaan

```
curl -X POST "https://logs.<region>.amazonaws.com/ingest/json?
logGroup=MyLogGroup&logStream=MyStream" \
-H "Authorization: Bearer ACWL<token>" \
-H "Content-Type: application/json" \
-d '[{"timestamp":1771007942000,"message":"User logged in","user_id":"u-123"},
{"timestamp":1771007943000,"message":"Order placed","order_id":"o-456"}]'
```

Respons

Sukses (semua acara diterima):

```
HTTP 200 OK
{}
```

Keberhasilan sebagian (beberapa acara ditolak):

```
{
  "partialSuccess": {
    "rejectedLogRecords": 5,
    "errorMessage": "{\"tooOldLogEventCount\": 3, \"tooNewLogEventCount\": 1,
    \"expiredLogEventCount\": 1}"
  }
}
```

`rejectedLogRecordsBidang` adalah jumlah total peristiwa yang ditolak. `errorMessageBidang` berisi rincian yang dikodekan JSON dengan alasan penolakan:

- `tooOldLogEventCount`— Acara dengan stempel waktu yang lebih tua dari periode retensi

- `tooNewLogEventCount`— Acara dengan stempel waktu terlalu jauh di masa depan
- `expiredLogEventCount`— Acara yang kedaluwarsa selama pemrosesan

Praktik terbaik

Acara batching

Untuk kinerja dan efisiensi yang lebih baik:

- Batch beberapa peristiwa dalam satu permintaan bila memungkinkan
- Ukuran batch yang disarankan: 10—100 acara per permintaan
- Ukuran permintaan maksimal: 1 MB

Penanganan kesalahan

Menerapkan penanganan kesalahan yang tepat dalam aplikasi Anda. Kode status HTTP umum:

- `200 OK`— Log berhasil dicerna
- `400 Bad Request`— Format atau parameter permintaan tidak valid
- `401 Unauthorized`— Token pembawa tidak valid atau kedaluwarsa
- `403 Forbidden`— Izin tidak mencukupi
- `404 Not Found`— Grup log atau aliran tidak ada
- `429 Too Many Requests`— Batas tarif terlampaui
- `500 Internal Server Error`— Kesalahan layanan (coba lagi dengan backoff eksponensial)

Batasan

- Ukuran acara maksimal: 256 KB per acara
- Ukuran permintaan maksimal: 1 MB
- Maksimum acara per permintaan: 10.000
- Nama grup log harus mengikuti konvensi penamaan CloudWatch Log
- Otentikasi token pembawa harus diaktifkan pada grup log jika otentikasi token pembawa digunakan.

Perbandingan titik akhir konsumsi HTTP

Fitur	Log HLC	Log ND-JSON	Log JSON Terstruktur	OpenTelemetry Log
Jalan	/services/collector/event	/ingest/bulk	/ingest/json	/v1/logs
Content-Type	application/json	application/json atau application/x-ndjson	application/json	application/json atau application/x-protobuf
Bidang stempel waktu	"time"(detik)	"timestamp" (milidetik)	"timestamp" (milidetik)	"timeUnixNano" (nanodetik)
Bidang yang diperlukan	"event"	Tidak ada	Tidak ada	Struktur OTLP () "resourceLogs"
Respon keberhasilan sebagian	Tidak	Ya	Ya	Ya
Dukungan parameter kueri	Ya	Ya	Ya	Tidak (hanya header)
Metadata entitas	Ya	Ya	Ya	Tidak
Menerima primitif	Tidak	Ya	Tidak	Tidak
Penguraian berbasis garis	Tidak	Ya	Tidak	Tidak
Dukungan Protobuf	Tidak	Tidak	Tidak	Ya

Fitur	Log HLC	Log ND-JSON	Log JSON Terstruktur	OpenTelemetry Log
Retry-After header	Tidak	Tidak	Tidak	Ya

Memilih titik akhir

- Menggunakan format HLC? Gunakan Log HLC. Muatan HLC Anda yang ada berfungsi dengan sedikit perubahan.
- line-by-lineLog streaming? Gunakan Log ND-JSON. Terbaik untuk saluran pipa log yang memancarkan satu peristiwa per baris. Paling fleksibel - menerima jenis nilai JSON apa pun.
- Mengirim muatan JSON terstruktur? Gunakan Log JSON Terstruktur. Terbaik untuk aplikasi yang menghasilkan objek atau array JSON yang terbentuk dengan baik.
- Sudah menggunakan OpenTelemetry? Gunakan OpenTelemetry Log. Menerima format OTLP JSON atau Protobuf dan mendukung respons keberhasilan sebagian dengan semantik coba lagi.

Menggunakan CloudWatch Log dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDKs) 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 untuk C++	AWS SDK untuk C++ contoh kode
AWS CLI	AWS CLI contoh kode
AWS SDK untuk Go	AWS SDK untuk Go contoh kode
AWS SDK untuk Java	AWS SDK untuk Java contoh kode
AWS SDK untuk JavaScript	AWS SDK untuk JavaScript contoh kode
AWS SDK untuk Kotlin	AWS SDK untuk Kotlin contoh kode
AWS SDK untuk .NET	AWS SDK untuk .NET contoh kode
AWS SDK untuk PHP	AWS SDK untuk PHP contoh kode
Alat AWS untuk PowerShell	Alat AWS untuk PowerShell contoh kode
AWS SDK untuk Python (Boto3)	AWS SDK untuk Python (Boto3) contoh kode
AWS SDK untuk Ruby	AWS SDK untuk Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk CloudWatch Log, lihat [Contoh kode untuk CloudWatch Log menggunakan AWS SDKs](#).

 Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

Manajemen log

CloudWatch Log menyediakan kemampuan manajemen log tingkat lanjut yang membantu Anda mengatur, mengubah, dan menganalisis data log Anda dengan lebih efektif. Fitur-fitur ini termasuk [sentralisasi data lintas akun dan lintas wilayah](#), [penemuan data otomatis dan manajemen skema](#), [transformasi log selama konsumsi](#), dan [analitik yang disempurnakan dengan aspek](#) untuk eksplorasi log interaktif.

Topik

- [Penemuan dan manajemen sumber data](#)
- [Fitur diaktifkan oleh sumber data](#)
- [Sumber data yang didukung](#)

Penemuan dan manajemen sumber data

CloudWatch Log secara otomatis menemukan dan mengkategorikan data log Anda berdasarkan sumber dan jenis data, sehingga lebih mudah untuk memahami dan mengelola log Anda dalam skala besar. Fitur ini menyediakan penemuan skema untuk sumber yang AWS dijual seperti Amazon VPC Flow Logs,, dan Route 53 CloudTrail, serta alat keamanan pihak ketiga.

Konsol Manajemen Log menyediakan tampilan tingkat tinggi dari log Anda yang diatur berdasarkan sumber dan jenis data, bukan hanya grup log. Organisasi ini membantu Anda:

- Lihat log yang dikategorikan berdasarkan AWS layanan, sumber pihak ketiga (seperti Okta atau CrowdStrike), dan sumber kustom
- Memahami skema dan struktur data log Anda secara otomatis
- Buat kebijakan indeks bidang berdasarkan bidang skema yang ditemukan
- Kelola log dengan lebih efisien di berbagai sumber data
- Log kueri oleh sumber data yang berbeda

Saat Anda [mengaktifkan pencatatan CloudWatch Log untuk AWS layanan yang didukung](#), CloudWatch Log secara otomatis menerapkan skema yang sesuai ke log Anda. Aplikasi skema otomatis ini membantu menjaga konsistensi dan memberikan wawasan langsung ke dalam struktur log Anda.

Apa itu Sumber Data CloudWatch Log?

CloudWatch Sumber Data Log adalah fitur yang menyediakan cara baru untuk mengatur dan mengkategorikan data log Anda berdasarkan sumber yang menghasilkan log. Sementara CloudWatch Log secara tradisional menggunakan grup log untuk mengatur log, Sumber Data menawarkan lapisan organisasi tambahan yang mengelompokkan log berdasarkan layanan asal dan jenis log mereka.

Bagaimana Sumber Data bekerja

Sumber Data menyediakan organisasi log berbasis layanan dan penemuan yang disederhanakan di seluruh infrastruktur Anda AWS . Anda dapat dengan mudah menemukan log dari layanan tertentu dan memfilter berdasarkan jenis log tanpa perlu mengetahui nama atau struktur grup log individu.

Untuk sumber pihak ketiga dan opsional untuk sumber log aplikasi, Sumber Data bekerja dengan CloudWatch pipeline untuk mengkategorikan log Anda. Saat mengonfigurasi pipeline untuk menyerap dan mengubah log, Anda menentukan nama dan jenis sumber data. CloudWatch Log kemudian secara otomatis mengkategorikan semua log yang diproses pipa. Untuk informasi selengkapnya, lihat [CloudWatch saluran pipa](#) di Panduan CloudWatch Pengguna Amazon.

Sumber Data mengkategorikan log menggunakan dua pengidentifikasi kunci:

- Nama Sumber Data: AWS Layanan, sumber pihak ketiga, atau aplikasi yang menghasilkan log (misalnya, Route 53, Amazon VPC, Okta SSO CloudTrail, atau Falcon). CrowdStrike
- Jenis Sumber Data: Jenis log spesifik yang dihasilkan oleh layanan tersebut.

Skema mendefinisikan struktur data log, termasuk bidang apa yang ada dan bagaimana informasi diatur. Sumber data tunggal dapat menghasilkan beberapa jenis log dengan skema dan tujuan yang berbeda. Misalnya, sumber AWS CloudTrail data memiliki dua jenis: peristiwa manajemen (yang melacak operasi bidang kontrol seperti membuat atau menghapus sumber daya) dan peristiwa data (yang melacak operasi bidang data seperti akses objek S3). Setiap jenis memiliki skema yang berbeda karena mereka menangkap berbagai jenis informasi.

Cara memulai

CloudWatch Log mengkategorikan log Anda ke sumber data berdasarkan asalnya. Metode ini tergantung pada jenis log yang Anda kerjakan:

Layanan AWS log

Log dari Layanan AWS yang [didukung](#) secara otomatis dikelompokkan berdasarkan sumber data tanpa konfigurasi apa pun yang diperlukan. CloudWatch Log mengenali log ini dan menerapkan nama dan jenis sumber data yang sesuai berdasarkan layanan asal.

Log pihak ketiga

Log pihak ketiga memerlukan saluran pipa untuk kategorisasi sumber data. Saat mengonfigurasi pipeline untuk menyerap log dari sumber pihak ketiga yang didukung seperti Microsoft Office 365, Okta CrowdStrike, atau Palo Alto Networks, Anda menentukan [nama sumber data dan ketik](#) dalam konfigurasi pipeline. CloudWatch Log secara otomatis mengkategorikan semua log yang diproses pipeline menggunakan pengidentifikasi tersebut.

Pipelines secara opsional dapat mengubah log pihak ketiga menjadi format Open Cybersecurity Schema Framework (OCSF) untuk analisis peristiwa keamanan standar. Ketika transformasi OCSF diaktifkan, nama dan jenis sumber data secara otomatis ditentukan berdasarkan pemetaan skema OCSF. Tanpa transformasi OCSF, Anda menentukan nama sumber data dan mengetik konfigurasi pipeline.

Log aplikasi

Untuk log aplikasi kustom, Anda dapat mengkategorikannya berdasarkan sumber data menggunakan salah satu metode berikut:

- Tag grup log - Tambahkan tag ke grup log Anda menggunakan kunci `cw:datasource:name` dan `cw:datasource:type` untuk menentukan nama dan jenis sumber data masing-masing untuk semua log yang dicerna dalam grup log. Nilai tag dapat mencapai 64 karakter dan mungkin hanya berisi huruf kecil, angka dan garis bawah. Mereka harus mulai dengan huruf atau angka dan mereka mungkin tidak mengandung garis bawah ganda (`__`).
- Konfigurasi saluran pipa - Konfigurasikan informasi sumber data melalui pipeline pemrosesan log saat menelan log aplikasi Anda.

Note

Nama sumber data tidak dapat dimulai dengan “aws” atau “amazon” untuk menghindari konflik dengan log AWS layanan.

Bidang sistem

CloudWatch Log secara otomatis menambahkan tiga bidang sistem ke log yang dikategorikan berdasarkan sumber data. Bidang ini berfungsi sebagai aspek default:

- `@data_source_name`- Berisi nama sumber data, atau “Tidak Diketahui” jika tidak ditentukan
- `@data_source_type`- Berisi jenis sumber data, atau “Tidak Diketahui” jika tidak ditentukan
- `@data_format`- Menunjukkan format data log

Ketika nama atau jenis sumber data tidak dapat ditentukan, bidang ini disetel ke “Tidak Diketahui”. Sumber data dengan nilai “Tidak Diketahui” masih terlihat dalam aspek dan di tabel sumber data di bawah “Manajemen Log” di Konsol, memungkinkan Anda mengidentifikasi log yang tidak dikategorikan dan dari grup log mana mereka berasal.

`@data_format` Bidang dapat berisi salah satu nilai berikut:

- `Default`- Log tertelan tanpa modifikasi.
- `Custom`- Log diproses melalui prosesor pipa atau log yang dicerna ke dalam grup log dengan `name/type` tag sumber data.
- `OCSF-<version>`- Log diproses dengan prosesor OCSF (Open Cybersecurity Schema Framework) di pipeline.
- `AWS-OTEL-LOG-V<version>`- OpenTelemetry log dicerna melalui titik akhir CloudWatch OTLP.
- `AWS-OTEL-TRACE-V<version>`- OpenTelemetry jejak tertelan melalui titik akhir CloudWatch OTLP.

Bidang sistem ini memungkinkan Anda untuk memfilter dan menanyakan log Anda berdasarkan sumber dan formatnya, sehingga lebih mudah untuk bekerja dengan log dari berbagai asal dan saluran pemrosesan.

Mengakses Sumber Data

Konsol

Di konsol CloudWatch Log, Anda menggunakan tab Manajemen Log untuk mengakses sumber data Anda. CloudWatch Log secara otomatis mengkonsolidasikan data log Anda berdasarkan sumber dan tipe data, terus menemukan data yang baru dicerna. Dari daftar sumber data, Anda dapat membuat saluran pipa, menentukan indeks bidang dan aspek.

AWS CLI

Gunakan perintah berikut untuk membuat daftar sumber data dan jenis log yang berbeda di akun Anda:

```
aws logs list-aggregate-log-group-summaries --group-by DATA_SOURCE_NAME_AND_TYPE
```

Hubungan dengan grup log

Sumber data melengkapi daripada mengganti grup log. Log Anda terus disimpan dalam grup log seperti sebelumnya, tetapi sekarang mereka juga secara otomatis ditandai dengan informasi sumber data. Organisasi ganda ini memungkinkan Anda untuk:

- Gunakan grup log untuk kebijakan kontrol akses dan penyimpanan berbutir halus
- Gunakan sumber data untuk penemuan dan analisis log berbasis layanan
- Log kueri menggunakan salah satu metode organisasi tergantung pada kebutuhan Anda

Sumber data membuatnya lebih mudah untuk bekerja dengan log dalam skala besar dengan memberikan tampilan layanan-sentris dari data log Anda di seluruh infrastruktur Anda. AWS

Fitur diaktifkan oleh sumber data

Sumber data memungkinkan pemrosesan log tingkat lanjut dan kemampuan analitik melalui penemuan lapangan dan struktur data yang konsisten.

- **Aspek:** Aspek adalah bidang log yang diindeks yang menyediakan pemfilteran dan analisis interaktif tanpa menulis kueri. CloudWatch Log secara otomatis membuat aspek untuk nama dan jenis sumber data, dan Anda dapat membuat kebijakan aspek pada bidang yang ditemukan untuk mempercepat pemecahan masalah. Aspek menampilkan distribusi dan hitungan nilai dalam Wawasan CloudWatch Log, sehingga mudah untuk mengidentifikasi pola melalui eksplorasi. point-and-click
- **Pipelines:** Buat pipeline transformasi yang berlaku untuk semua log dari nama dan jenis sumber data tertentu. Ini memungkinkan Anda untuk menentukan aturan pemrosesan yang konsisten untuk log dari sumber yang sama.
- **Penemuan lapangan:** CloudWatch Log secara otomatis menemukan bidang dan tipe datanya untuk setiap nama sumber data dan kombinasi tipe berdasarkan prosesor pipeline. Untuk log AWS

terkelola, struktur bidang sudah ditentukan sebelumnya. Untuk log aplikasi, sebaiknya pertahankan format log yang konsisten untuk memaksimalkan kompatibilitas dengan alat analitik seperti tabel Amazon S3 yang memerlukan struktur bidang yang terdefinisi dengan baik.

Anda dapat melihat daftar lengkap bidang dan tipenya untuk sumber data apa pun menggunakan GetLogFields API:

```
aws logs get-log-fields --data-source-name <name> --data-source-type <type>
```

Penemuan dan konsistensi lapangan ini memungkinkan analisis dan integrasi tingkat lanjut, karena alat eksternal dapat bekerja dengan struktur bidang yang dapat diprediksi saat memproses data log Anda.

Sumber data yang didukung

CloudWatch Log mendukung berbagai sumber data termasuk AWS layanan, keamanan pihak ketiga dan alat TI, dan sumber aplikasi khusus. Halaman ini menyediakan katalog lengkap dari semua sumber data yang didukung. Untuk informasi tentang bagaimana sumber data ditemukan dan dikelola, lihat [the section called “Penemuan dan manajemen sumber data”](#).

Topik

- [Didukung Layanan AWS untuk sumber data](#)
- [Sumber pihak ketiga yang didukung untuk sumber data](#)
- [Sumber kustom](#)

Didukung Layanan AWS untuk sumber data

Tabel berikut mencantumkan Layanan AWS yang secara otomatis dikategorikan oleh CloudWatch Log sebagai sumber data:

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
amazon_api_gateway	access
amazon_bedrock_agentcore	browser_usage

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
amazon_bedrock_agentcore	code_interpreter_application
amazon_bedrock_agentcore	code_interpreter_usage
amazon_bedrock_agentcore	gateway_application
amazon_bedrock_agentcore	identity_workload_application
amazon_bedrock_agentcore	memory_application
amazon_bedrock_agentcore	online_evaluation_config
amazon_bedrock_agentcore	runtime_application
amazon_bedrock_agentcore	runtime_usage
amazon_bedrock_agents	application
amazon_bedrock_agents	event
amazon_bedrock_knowledge_bases	application
amazon_cloudfront	access
amazon_cloudfront	connection
amazon_cloudwatch	rum_app_monitor
amazon_cognito	user_pool
amazon_ec2	verified_access
amazon_eks	api_server
amazon_eks	audit
amazon_eks	authenticator
amazon_eks	controller_manager

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
amazon_eks	scheduler
amazon_elasticache	cluster
amazon_eventbridge	eventbus_error
amazon_eventbridge	eventbus_info
amazon_eventbridge	pipes_execution
amazon_interactive_video_service	chat
amazon_managed_prometheus	scraper
amazon_managed_prometheus	workspace
amazon_msk	broker
amazon_msk	connect
amazon_opensearch_service	pipeline
amazon_q_business	events
amazon_q_business	sync_job
amazon_q_connect	events
amazon_route53	global_resolver_query
amazon_route53	hosted_zones
amazon_route53	profiles_resolver_query
amazon_route53	resolver_query
amazon_sagemaker	workteam_activity
amazon_ses	ingress_endpoints

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
amazon_ses	rule_sets
amazon_ses	traffic_policy
amazon_vpc	flow
amazon_vpc	route_server_peer
amazon_vpc_lattice	access
amazon_vpc_lattice	resource_access
amazon_workmail	access_control
amazon_workmail	authentication
amazon_workmail	personal_access
amazon_workmail	workmail_access
amazon_workmail	workmail_availability
aws_b2b_data_interchange	execution
aws_backup	data_access
aws_backup	hypervisor
aws_clean_rooms	analysis
aws_client_vpn	connection
aws_client_vpn	event
aws_cloudtrail	data
aws_cloudtrail	management
aws_elemental_mediapackage	egress_access

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
aws_elemental_mediapackage	ingress_access
aws_elemental_mediatailor	ad_decision
aws_elemental_mediatailor	manifest
aws_elemental_mediatailor	transcode
aws_entity_resolution	id_mapping_workflow
aws_entity_resolution	matching_workflow
aws_iot_fleetwise	error
aws_mainframe_modernization	batch_job
aws_mainframe_modernization	config
aws_mainframe_modernization	console
aws_mainframe_modernization	dataset_import
aws_network_firewall	alert
aws_network_firewall	flow
aws_network_firewall	tls
aws_nlb	access
aws_pcs	job_completion
aws_pcs	scheduler
aws_security_hub	compliance_finding
aws_security_hub	data_security_finding
aws_security_hub	detection_finding

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
aws_security_hub	vulnerability_finding
aws_security_hub_cspm	asff_finding
aws_shield	protection_flow
aws_step_functions	express
aws_step_functions	standard
aws_transfer_family	server
aws_waf	access

Sumber pihak ketiga yang didukung untuk sumber data

Tabel berikut mencantumkan sumber pihak ketiga yang secara otomatis dikategorikan oleh CloudWatch Log sebagai sumber data saat dicerna melalui pipeline:

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
cisco_umbrella	data_security_finding
cisco_umbrella	dns_activity
cisco_umbrella	entity_management
cisco_umbrella	network_activity
crowdstrike_falcon	detection_finding
crowdstrike_falcon	process_activity
github_auditlogs	account_change
github_auditlogs	api_activity

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
github_auditlogs	entity_management
microsoft_entraid	account_change
microsoft_entraid	authentication
microsoft_entraid	entity_management
microsoft_entraid	user_access_management
microsoft_office365	account_change
microsoft_office365	application_lifecycle
microsoft_office365	authentication
microsoft_office365	compliance_finding
microsoft_office365	detection_finding
microsoft_office365	email_activity
microsoft_office365	file_hosting_activity
microsoft_office365	group_management
microsoft_office365	incident_finding
microsoft_office365	user_access_management
microsoft_office365	vulnerability_finding
microsoft_office365	web_resources_activity
microsoft_windows	account_change
microsoft_windows	authentication
microsoft_windows	entity_management

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
microsoft_windows	event_log_activity
microsoft_windows	file_system_activity
microsoft_windows	group_management
microsoft_windows	kernel_activity
okta_auth0	api_activity
okta_auth0	authentication
okta_sso	api_activity
okta_sso	authentication
okta_sso	detection_finding
okta_sso	entity_management
paloaltonetworks_nextgenerationfirewall	authentication
paloaltonetworks_nextgenerationfirewall	detection_finding
paloaltonetworks_nextgenerationfirewall	network_activity
paloaltonetworks_nextgenerationfirewall	process_activity
sentinelone_endpointsecurity	dns_activity
sentinelone_endpointsecurity	file_system_activity
sentinelone_endpointsecurity	http_activity

Nama Sumber Data (bidang @data_source_name)	Jenis Sumber Data (bidang @data_source_type)
sentinelone_endpointsecurity	process_activity
servicenow_cmdb	api_activity
servicenow_cmdb	datastore_activity
servicenow_cmdb	entity_management
wiz_cnapp	api_activity
wiz_cnapp	authentication
wiz_cnapp	compliance_finding
wiz_cnapp	detection_finding
wiz_cnapp	vulnerability_finding
zscaler_internetaccess	authentication
zscaler_internetaccess	dns_activity
zscaler_internetaccess	http_activity
zscaler_internetaccess	network_activity

Sumber pihak ketiga tambahan melalui AWS Security Hub CSPM

Temuan keamanan pihak ketiga tambahan tersedia melalui integrasi CSPM AWS Security Hub. Mitra berikut mengirimkan temuan ke Security Hub CSPM, yang kemudian tersedia sebagai sumber data di CloudWatch Log. Untuk detail lengkap tentang integrasi ini, lihat Integrasi [produk pihak ketiga dengan Security Hub CSPM](#) di Panduan Pengguna Security Hub AWS .

Mitra	Integrasi
3 CORESec — NTA	Mengirimkan temuan melalui Security Hub CSPM
Logika Peringatan - Manajemen SIEMless Ancaman	Mengirimkan temuan melalui Security Hub CSPM
Aqua Security - Platform Keamanan Asli Cloud	Mengirimkan temuan melalui Security Hub CSPM
Aqua Security — Bangku Kube	Mengirimkan temuan melalui Security Hub CSPM
Armor - Armor Dimana Saja	Mengirimkan temuan melalui Security Hub CSPM
AttackIQ	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Barracuda - Penjaga Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
BiGid — BiGid Perusahaan	Mengirimkan temuan melalui Security Hub CSPM
Hexagon Biru	Mengirimkan temuan melalui Security Hub CSPM
Titik Periksa - CloudGuard IaaS	Mengirimkan temuan melalui Security Hub CSPM
Check Point - Manajemen CloudGuard Postur	Mengirimkan temuan melalui Security Hub CSPM
Claroty — xDome	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
Keamanan Penyimpanan Cloud - Antivirus untuk Amazon S3	Mengirimkan temuan melalui Security Hub CSPM
Keamanan Kontras - Penilaian Kontras	Mengirimkan temuan melalui Security Hub CSPM
CrowdStrike — CrowdStrike Elang	Mengirimkan temuan melalui Security Hub CSPM
CyberArk — Analisis Ancaman Istimewa	Mengirimkan temuan melalui Security Hub CSPM
Teorema Data	Mengirimkan temuan melalui Security Hub CSPM
Drata	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - CASB	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - Gerbang Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - DLP	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - NGFW	Mengirimkan temuan melalui Security Hub CSPM
Fugue	Mengirimkan temuan melalui Security Hub CSPM
Guardicore — Pusat	Mengirimkan temuan melalui Security Hub CSPM
HackerOne — Kecerdasan Kerentanan	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
JFrog — Xray	Mengirimkan temuan melalui Security Hub CSPM
Juniper Networks — Firewall Generasi Berikutnya VsRx	Mengirimkan temuan melalui Security Hub CSPM
k9 Keamanan - Penganalisis Akses	Mengirimkan temuan melalui Security Hub CSPM
Lacework	Mengirimkan temuan melalui Security Hub CSPM
McAfee — MVISION CNAPP	Mengirimkan temuan melalui Security Hub CSPM
NETSCOUT — Penyelidik Cyber	Mengirimkan temuan melalui Security Hub CSPM
Orca - Platform Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Palo Alto - Komputasi Awan Prisma	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Palo Alto - Perusahaan Awan Prisma	Mengirimkan temuan melalui Security Hub CSPM
Plerion - Platform Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Prowler	Mengirimkan temuan melalui Security Hub CSPM
Qualys - Manajemen Kerentanan	Mengirimkan temuan melalui Security Hub CSPM
Rapid7 - InsightVM	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
SentinelOne	Mengirimkan temuan melalui Security Hub CSPM
Snyk	Mengirimkan temuan melalui Security Hub CSPM
Keamanan Sonrai - Sonrai Dig	Mengirimkan temuan melalui Security Hub CSPM
Sophos - Perlindungan Server	Mengirimkan temuan melalui Security Hub CSPM
StackRox — Keamanan Kubernetes	Mengirimkan temuan melalui Security Hub CSPM
Logika Sumo - Analisis Data Mesin	Mengirimkan temuan melalui Security Hub CSPM
Symantec - Perlindungan Beban Kerja Cloud	Mengirimkan temuan melalui Security Hub CSPM
Tenable.io	Mengirimkan temuan melalui Security Hub CSPM
Trend Micro - Cloud One	Mengirimkan temuan melalui Security Hub CSPM
Vectra - Deteksi Cognito	Mengirimkan temuan melalui Security Hub CSPM
Wiz	Mengirimkan temuan melalui Security Hub CSPM
Caveonix - Awan Caveonix	Mengirim dan menerima temuan melalui Security Hub CSPM

Mitra	Integrasi
Penjaga Awan	Mengirim dan menerima temuan melalui Security Hub CSPM
DisruptOps	Mengirim dan menerima temuan melalui Security Hub CSPM
Kion	Mengirim dan menerima temuan melalui Security Hub CSPM
Turbot	Mengirim dan menerima temuan melalui Security Hub CSPM

Note

Daftar ini mencerminkan integrasi mitra Security Hub yang mengirimkan temuan pada saat penulisan. Karena AWS Security Hub secara teratur menambahkan integrasi mitra baru, lihat [integrasi produk Pihak Ketiga dengan Security Hub CSPM](#) di Panduan Pengguna AWS Security Hub untuk up-to-date daftar mitra terbanyak yang tersedia.

Sumber kustom

Untuk log aplikasi kustom, Anda dapat menentukan kategorisasi sumber data Anda sendiri menggunakan salah satu metode berikut:

- Tag grup log - Tambahkan tag ke grup log Anda menggunakan kunci `cw:datasource:name` dan `cw:datasource:type` untuk menentukan nama dan jenis sumber data untuk semua log yang dicerna dalam grup log.
- Konfigurasi saluran pipa — Konfigurasikan informasi sumber data melalui pipeline pemrosesan log saat menelan log aplikasi Anda.

 Note

Nama sumber data khusus tidak dapat dimulai dengan “aws” atau “amazon” untuk menghindari konflik dengan log AWS layanan.

Menganalisis data log dengan Wawasan CloudWatch Log

Dengan Wawasan CloudWatch Log, Anda dapat mencari dan menganalisis data log Anda secara interaktif di Log Amazon CloudWatch. Anda dapat melakukan kueri untuk membantu Anda agar lebih efisien dan efektif dalam menanggapi masalah operasional. Selain kueri menggunakan grup log, Anda dapat melakukan kueri menggunakan aspek, sumber data, dan tipe data. Jika terjadi masalah, Anda dapat menggunakan Wawasan CloudWatch Log untuk mengidentifikasi penyebab potensial dan memvalidasi perbaikan yang diterapkan. Anda dibatasi hingga 100 Wawasan CloudWatch Log bersamaan QL per akun, termasuk kueri yang ditambahkan ke dasbor. Selain itu, Anda dapat menjalankan 15 kueri bersamaan untuk OpenSearch Service PPL atau OpenSearch Service SQL.

CloudWatch Logs Insights mendukung tiga bahasa kueri yang dapat Anda gunakan untuk kueri:

- Bahasa kueri Logs Insights yang dibuat khusus (Logs Insights QL) dengan beberapa perintah sederhana namun kuat.
- OpenSearch Service Piped Processing Language (PPL). OpenSearch PPL memungkinkan Anda menganalisis log Anda menggunakan serangkaian perintah yang dibatasi oleh pipa (|).

Dengan OpenSearch PPL, Anda dapat mengambil, menanyakan, dan menganalisis data dengan menggunakan perintah yang disatukan, sehingga lebih mudah untuk memahami dan menyusun kueri yang kompleks. Sintaks memungkinkan rantai perintah untuk mengubah dan memproses data. Dengan PPL, Anda dapat memfilter dan mengumpulkan data, dan menggunakan serangkaian matematika, string, tanggal, kondisional, dan fungsi lainnya yang kaya untuk analisis.

- OpenSearch Layanan Structured Query Language (SQL). Dengan query OpenSearch SQL, Anda dapat menganalisis log Anda dengan cara deklaratif. Anda dapat menggunakan perintah seperti SELECT, FROM, WHERE, GROUP BY, HAVING, dan berbagai perintah dan fungsi lain yang tersedia di SQL. Anda dapat mengeksekusi JOINS di seluruh grup log, mengkorelasikan data di seluruh log menggunakan sub-kueri, dan menggunakan kumpulan kaya fungsi JSON, Matematika, String, Bersyarat, dan SQL lainnya untuk melakukan analisis yang kuat pada log.

Saat Anda menggunakan perintah SQL atau PPL, pastikan untuk melampirkan bidang dengan karakter khusus (non-alfabet dan non-numerik) di backticks agar berhasil menyanyakannya.

Misalnya, lampirkan, `@messageOperation.Export`, dan `Test::Field` di backticks. Anda tidak perlu melampirkan bidang dengan nama abjad murni di backticks.

CloudWatch Logs Insights menawarkan fitur-fitur berikut yang tersedia untuk digunakan dengan salah satu bahasa kueri.

- [Penemuan otomatis bidang log](#) dalam log dari AWS layanan seperti Amazon Route 53,, AWS Lambda AWS CloudTrail, dan Amazon VPC, dan aplikasi atau log kustom apa pun yang memancarkan peristiwa log sebagai JSON.
- Membuat [indeks bidang](#) untuk mengurangi biaya dan mempercepat hasil, terutama untuk kueri sejumlah besar grup log atau peristiwa log. Setelah membuat indeks bidang bidang yang umum dalam peristiwa log Anda, Anda dapat menggunakannya dalam kueri. Kueri melewati pemrosesan peristiwa log yang diketahui tidak menyertakan bidang yang diindeks, dan memproses lebih sedikit data.

 Note

`filterIndexPerintah` ini hanya tersedia di Log Insights QL.

- [Deteksi dan analisis pola](#) dalam peristiwa log Anda. Pola adalah struktur teks bersama yang berulang di antara bidang log Anda. Saat melihat hasil kueri, Anda dapat memilih tab Pola untuk melihat pola yang ditemukan CloudWatch Log berdasarkan sampel hasil Anda.
- [Menyimpan kueri](#), melihat riwayat kueri, menjalankan kembali kueri tersimpan, dan [menggunakan kueri tersimpan dengan](#) parameter.
- [Menambahkan kueri ke dasbor](#).
- [Mengkripsi hasil kueri](#) dengan AWS Key Management Service
- [Pembuatan kueri menggunakan bahasa alami](#) memungkinkan Anda menggunakan bahasa alami untuk membuat kueri Wawasan CloudWatch Log. Anda dapat mengajukan pertanyaan tentang atau menjelaskan data yang Anda cari, kemudian AI menghasilkan kueri berdasarkan prompt Anda dan memberikan line-by-line penjelasan tentang cara kerja kueri.
- [Gunakan aspek untuk mengelompokkan, memfilter, dan mengeksplorasi log Anda secara interaktif](#).
- [Log sekitarnya](#) memungkinkan Anda melihat baris log sebelum dan sesudah catatan log tertentu dalam hasil kueri Anda untuk mendapatkan konteks instan seputar peristiwa log kritis. Anda dapat mengonfigurasi rentang untuk melihat 5, 10, 20, 50, atau 100 baris log sebelum dan sesudah catatan yang dipilih, dan mencari kata kunci tertentu dalam log sekitarnya.

Fitur Wawasan CloudWatch Log berikut hanya didukung jika Anda menggunakan QL Wawasan Log.

- [Kueri perbandingan](#) yang membandingkan peristiwa log dalam grup log dengan peristiwa log dari periode waktu sebelumnya.

 Important

CloudWatch Wawasan Log tidak dapat mengakses peristiwa log dengan stempel waktu yang mendahului waktu pembuatan grup log.

Jika Anda masuk ke akun yang disiapkan sebagai akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat menjalankan kueri Wawasan CloudWatch Log pada grup log di akun sumber yang ditautkan ke akun pemantauan ini. Anda dapat menjalankan kueri yang menanyakan beberapa grup log yang terletak di akun yang berbeda. Untuk informasi lebih lanjut, lihat [CloudWatch observabilitas lintas akun](#).

Saat membuat kueri menggunakan Log Insights QL, Anda juga dapat menggunakan bahasa alami untuk membuat kueri Wawasan CloudWatch Log. Untuk melakukan hal itu, ajukan pertanyaan atau jelaskan data yang Anda cari. Kemampuan berbantuan AI ini menghasilkan kueri berdasarkan prompt Anda dan memberikan line-by-line penjelasan tentang cara kerja kueri. Untuk informasi selengkapnya, lihat [Menggunakan bahasa alami untuk membuat dan memperbarui kueri Wawasan CloudWatch Log](#).

Kueri menggunakan salah satu bahasa kueri yang didukung habis setelah 60 menit, jika belum selesai. Hasil kueri tersedia selama tujuh hari.

CloudWatch Kueri Log Insights dikenakan biaya berdasarkan jumlah data yang ditanyakan, terlepas dari bahasa kueri. Untuk informasi selengkapnya, lihat [CloudWatch Harga Amazon](#).

Anda dapat menggunakan Wawasan CloudWatch Log untuk mencari data log yang dikirim ke CloudWatch Log pada 5 November 2018 atau lebih baru.

 Important

Jika tim keamanan jaringan Anda tidak mengizinkan penggunaan soket web, saat ini Anda tidak dapat mengakses bagian CloudWatch Logs Insights dari CloudWatch konsol. Anda dapat menggunakan kemampuan kueri Wawasan CloudWatch Log menggunakan APIs. Untuk informasi selengkapnya, lihat [StartQuery](#) di Referensi API Amazon CloudWatch Logs.

Daftar Isi

- [Bahasa kueri yang didukung](#)

- [Gunakan bahasa alami untuk menghasilkan dan memperbarui kueri Wawasan CloudWatch Log](#)
- [Log yang didukung dan bidang yang ditemukan](#)
- [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#)
- [Gunakan aspek untuk mengelompokkan dan menjelajahi log](#)
- [Melihat log sekitarnya di Wawasan CloudWatch Log](#)
- [Analisis pola](#)
- [Simpan dan jalankan kembali kueri CloudWatch Logs Insights](#)
- [Tambahkan kueri ke dasbor atau ekspor hasil kueri](#)
- [Lihat kueri atau riwayat kueri yang sedang berjalan](#)
- [Enkripsi hasil kueri dengan AWS Key Management Service](#)
- [Hasilkan ringkasan bahasa alami dari hasil kueri CloudWatch Log Insights](#)

Bahasa kueri yang didukung

Bagian berikut mencantumkan perintah yang didukung dalam setiap bahasa kueri. Mereka juga menggambarkan format sintaks dan memberikan contoh query.

Topik

- [CloudWatch Bahasa kueri Wawasan Log \(Wawasan Log QL\)](#)
- [OpenSearch Bahasa Pemrosesan Pipa \(PPL\)](#)
- [OpenSearch Bahasa Kueri Terstruktur \(SQL\)](#)

CloudWatch Bahasa kueri Wawasan Log (Wawasan Log QL)

Bagian ini mencakup dokumentasi lengkap perintah dan fungsi QL Wawasan Log. Ini juga mencakup kueri sampel untuk bahasa ini.

Untuk informasi tentang bahasa kueri lain yang dapat Anda gunakan, lihat [OpenSearch Layanan PPL](#), [SQL OpenSearch Layanan](#), dan [CloudWatch Metrics Insights](#)

Topik

- [CloudWatch Sintaks kueri bahasa Wawasan Log](#)
- [Memulai dengan Logs Insights QL: Tutorial kueri](#)
- [Kueri Sampel](#)

- [Bandingkan \(diff\) dengan rentang waktu sebelumnya](#)
- [Visualisasikan data log dalam grafik](#)

CloudWatch Sintaks kueri bahasa Wawasan Log

Bagian ini memberikan rincian tentang QL Wawasan Log. Sintaks kueri mendukung berbagai fungsi dan operasi yang menyertakan tetapi tidak terbatas pada fungsi umum, operasi aritmatika dan perbandingan, dan ekspresi reguler.

Important

Untuk menghindari biaya yang berlebihan dengan menjalankan kueri besar, ingatlah praktik terbaik berikut:

- Pilih hanya grup log yang diperlukan untuk setiap kueri.
- Selalu tentukan rentang waktu tersempit yang mungkin untuk kueri Anda.
- Saat Anda menggunakan konsol untuk menjalankan kueri, batalkan semua kueri sebelum menutup halaman konsol Wawasan CloudWatch Log. Jika tidak, kueri terus berjalan hingga selesai.
- Saat Anda menambahkan widget Wawasan CloudWatch Log ke dasbor, pastikan dasbor tidak menyegarkan pada frekuensi tinggi, karena setiap penyegaran memulai kueri baru.

Untuk membuat kueri yang berisi beberapa perintah, pisahkan perintah dengan karakter pipa (|).

Untuk membuat kueri yang berisi komentar, matikan komentar dengan karakter hash (#).

Note

CloudWatch Logs Insights secara otomatis menemukan bidang untuk jenis log yang berbeda dan menghasilkan bidang yang dimulai dengan karakter @. Untuk informasi selengkapnya tentang bidang ini, lihat [Log yang didukung dan bidang yang ditemukan](#) di Panduan CloudWatch Pengguna Amazon.

Tabel berikut menjelaskan secara singkat setiap perintah. Mengikuti tabel ini adalah deskripsi yang lebih komprehensif dari setiap perintah, dengan contoh.

Note

Semua perintah kueri QL Wawasan Log didukung pada grup log di kelas log Standar. Grup log di kelas log Akses Jarang mendukung semua perintah kueri QL Wawasan Log kecuali `pattern`, `diff` dan `unmask`.

<u>anomaly</u>	Mengidentifikasi pola yang tidak biasa dalam data log Anda menggunakan pembelajaran mesin.
<u>display</u>	Menampilkan bidang atau bidang tertentu dalam hasil kueri.
<u>fields</u>	Menampilkan bidang tertentu dalam hasil kueri dan mendukung fungsi dan operasi yang dapat Anda gunakan untuk memodifikasi nilai bidang dan membuat bidang baru untuk digunakan dalam kueri Anda.
<u>filter</u>	Memfilter kueri untuk mengembalikan hanya peristiwa log yang cocok dengan satu atau beberapa kondisi.
<u>filterIndex</u>	Memaksa kueri untuk mencoba memindai hanya grup log yang keduanya diindeks pada bidang yang disebutkan dalam indeks bidang dan juga berisi nilai untuk indeks bidang tersebut. Ini mengurangi volume yang dipindai dengan mencoba memindai hanya peristiwa log dari grup log ini yang berisi nilai yang ditentukan dalam kueri untuk indeks bidang ini. Perintah ini tidak didukung untuk grup log di kelas log Akses Jarang.
<u>pattern</u>	Secara otomatis mengelompokkan data log Anda ke dalam pola. Pola adalah struktur teks bersama yang berulang di antara bidang log Anda. CloudWatch Logs Insights menyediakan cara bagi Anda untuk menganalisis pola yang ditemukan dalam peristiwa log Anda. Untuk informasi selengkapnya, lihat Analisis pola .
<u>diff</u>	Membandingkan peristiwa log yang ditemukan dalam periode waktu yang Anda minta dengan peristiwa log dari periode waktu sebelumnya dengan panjang yang sama, sehingga Anda dapat mencari tren dan mencari tahu apakah peristiwa log tertentu baru.

<u>parse</u>	Mengekstrak data dari bidang log untuk membuat bidang yang diekstraksi yang dapat Anda proses dalam kueri Anda. parse mendukung mode glob menggunakan wildcard, dan ekspresi reguler.
<u>sort</u>	Menampilkan peristiwa log yang dikembalikan dalam urutan ascending (asc) atau descending (desc).
<u>SOURCE</u>	Termasuk SOURCE dalam kueri adalah cara yang berguna untuk menentukan sejumlah besar grup log berdasarkan awalan nama grup log, pengidentifikasi akun, dan kelas grup log untuk disertakan dalam kueri. Perintah ini didukung hanya ketika Anda membuat kueri di AWS CLI atau secara terprogram, bukan di CloudWatch konsol.
<u>stats</u>	Hitung statistik agregat menggunakan nilai di bidang log.
<u>limit</u>	Menentukan jumlah maksimum peristiwa log yang Anda ingin query Anda untuk kembali. sort Berguna dengan mengembalikan hasil “20 teratas” atau “20 terbaru”.
<u>dedup</u>	Menghapus hasil duplikat berdasarkan nilai tertentu di bidang yang Anda tentukan.
<u>unmask</u>	Menampilkan semua konten peristiwa log yang memiliki beberapa konten yang disembunyikan karena kebijakan perlindungan data. Untuk informasi selengkapnya tentang perlindungan data di grup log, lihat Membantu melindungi data log sensitif dengan masking .
<u>unnest</u>	Meratakan daftar yang diambil sebagai masukan untuk menghasilkan beberapa catatan dengan satu catatan untuk setiap elemen dalam daftar.
<u>lookup</u>	Memperkaya peristiwa log dengan data dari tabel pencarian dengan mencocokkan nilai bidang. Gunakan tabel pencarian untuk menambahkan data referensi seperti detail pengguna, nama aplikasi, atau informasi produk ke hasil kueri Anda.

<u>join</u>	Menggabungkan peristiwa log dari grup log sumber dengan peristiwa dari grup log lain atau hasil kueri berdasarkan bidang yang cocok. Gunakan perintah join untuk mengkorelasikan peristiwa log terkait di berbagai sumber menggunakan kunci yang umum di dalamnya seperti pencocokan pengidentifikasi permintaan atau transaksi. IDs
<u>subqueries</u>	Subquery adalah kueri Wawasan Log bersarang yang dapat digunakan sebagai masukan ke kueri lain. Subquery dapat digunakan untuk mendapatkan set hasil antara yang kemudian dikonsumsi oleh perintah berikutnya.
<u>Operasi dan fungsi lainnya</u>	CloudWatch Logs Insights juga mendukung banyak perbandingan, aritmatika, datetime, numerik, string, alamat IP, dan fungsi dan operasi umum.

Bagian berikut memberikan detail selengkapnya tentang perintah kueri Wawasan CloudWatch Log.

Topik

- [Log Insights Perintah QL didukung di kelas log](#)
- [anomali](#)
- [tampilan](#)
- [ladang](#)
- [filter](#)
- [FilterIndex](#)
- [SUMBER](#)
- [pola](#)
- [perbedaan](#)
- [mengurai](#)
- [menyortir](#)
- [statistik](#)
- [batasan](#)
- [dedup](#)
- [membuka kedok](#)

- [unnest](#)
- [pencarian](#)
- [join](#)
- [subkueri](#)
- [Boolean, perbandingan, numerik, datetime, dan fungsi lainnya](#)
- [Bidang yang berisi karakter khusus](#)
- [Gunakan alias dan komentar dalam kueri](#)

Log Insights Perintah QL didukung di kelas log

Semua perintah kueri QL Wawasan Log didukung pada grup log di kelas log Standar. Grup log di kelas log Akses Jarang mendukung semua perintah kueri `kecualipattern`, `difffilterIndex`, dan `unmask`.

anomaly

Gunakan `anomaly` untuk secara otomatis mengidentifikasi pola yang tidak biasa dan potensi masalah dalam data log Anda menggunakan pembelajaran mesin.

`anomalyPerintah` memperluas `pattern` fungsionalitas yang ada dan memanfaatkan analitik lanjutan untuk membantu mengidentifikasi potensi anomali dalam data log. Anda dapat menggunakannya `anomaly` untuk mengurangi waktu yang diperlukan untuk mengidentifikasi dan menyelesaikan masalah operasional dengan secara otomatis menampilkan pola atau perilaku yang tidak biasa di log Anda.

`anomalyPerintah` bekerja dengan [pattern](#) perintah untuk pertama mengidentifikasi pola log, kemudian mendeteksi anomali dalam pola tersebut. Anda juga dapat menggabungkan `anomaly` dengan [sort](#) perintah [filter](#) atau untuk memfokuskan deteksi anomali pada himpunan bagian tertentu dari data Anda.

Masukan Perintah Anomali

`anomalyPerintah` ini biasanya digunakan setelah [pattern](#) perintah untuk menganalisis pola yang diidentifikasi dalam data log Anda. Perintah tidak memerlukan parameter tambahan dan menganalisis output dari perintah sebelumnya dalam kueri Anda.

Jenis Anomali yang Diidentifikasi

`anomalyPerintah` mengidentifikasi lima jenis anomali yang berbeda:

- **Anomali Frekuensi Pola:** Frekuensi yang tidak biasa dari pola log tertentu, seperti ketika aplikasi mulai menghasilkan lebih banyak pesan kesalahan daripada biasanya.
- **Anomali Pola Baru:** Pola log yang sebelumnya tidak terlihat yang mungkin menunjukkan jenis kesalahan atau pesan baru yang muncul di log Anda.
- **Anomali Variasi Token:** Perubahan tak terduga dalam konten pesan log yang mungkin menunjukkan variasi yang tidak biasa dalam format log yang diharapkan.
- **Anomali Token Numerik:** Perubahan yang tidak biasa dalam nilai numerik dalam log yang dapat membantu mendeteksi potensi masalah kinerja atau variasi metrik yang tidak terduga.
- **Anomali Kode Kesalahan HTTP:** Pola yang terkait dengan respons kesalahan HTTP, sangat berguna saat memantau aplikasi web dan APIs

Output Perintah Anomali

`anomaly`Perintah mempertahankan semua bidang dari data input dan menambahkan hasil deteksi anomali untuk membantu mengidentifikasi pola yang tidak biasa dalam data log Anda.

Contoh

Perintah berikut mengidentifikasi pola dalam data log Anda dan kemudian mendeteksi anomali dalam pola tersebut:

```
fields @timestamp, @message
| pattern @message
| anomaly
```

`anomaly`Perintah ini dapat digunakan dengan pemfilteran untuk fokus pada jenis log tertentu:

```
fields @timestamp, @message
| filter @type = "REPORT"
| pattern @message
| anomaly
```

`anomaly`Perintah dapat dikombinasikan dengan penyortiran untuk mengatur hasil:

```
fields @timestamp, @message
| filter @type = "ERROR"
| pattern @message
| anomaly
```

```
| sort @timestamp desc
```

tampilan

Gunakan `display` untuk menampilkan bidang atau bidang tertentu dalam hasil kueri.

`displayPerintah` hanya menampilkan bidang yang Anda tentukan. Jika kueri Anda berisi beberapa `display` perintah, hasil kueri hanya menampilkan bidang atau bidang yang Anda tentukan dalam `display` perintah akhir.

Contoh: Menampilkan satu bidang

Cuplikan kode menunjukkan contoh kueri yang menggunakan perintah `parse` untuk mengekstrak data dari `@message` untuk membuat bidang yang diekstraksi dan. `loggingType` `loggingMessage` Query mengembalikan semua peristiwa log di mana nilai-nilai untuk `loggingType` adalah `ERROR`. `display` hanya menampilkan nilai untuk `loggingMessage` dalam hasil query.

```
fields @message
| parse @message "[*] *" as loggingType, loggingMessage
| filter loggingType = "ERROR"
| display loggingMessage
```

Tip

Gunakan `display` hanya sekali dalam kueri. Jika Anda menggunakan `display` lebih dari sekali dalam kueri, hasil kueri menunjukkan bidang yang ditentukan dalam kemunculan terakhir `display` perintah yang digunakan.

ladang

Gunakan `fields` untuk menampilkan bidang tertentu dalam hasil kueri.

Jika kueri Anda berisi beberapa `fields` perintah dan tidak menyertakan `display` perintah, hasilnya akan menampilkan semua bidang yang ditentukan dalam `fields` perintah.

Contoh: Menampilkan bidang tertentu

Contoh berikut menunjukkan query yang mengembalikan 20 peristiwa log dan menampilkannya dalam urutan menurun. Nilai untuk `@timestamp` dan `@message` ditampilkan dalam hasil query.

```
fields @timestamp, @message
| sort @timestamp desc
| limit 20
```

Gunakan `fields` alih-alih `display` saat Anda ingin menggunakan berbagai fungsi dan operasi yang didukung oleh `fields` untuk memodifikasi nilai bidang dan membuat bidang baru yang dapat digunakan dalam kueri.

Anda dapat menggunakan `fields` perintah dengan kata kunci untuk membuat bidang yang diekstraksi yang menggunakan bidang dan fungsi dalam peristiwa log Anda. Misalnya, `fields ispresent as isRes` membuat bidang yang diekstraksi bernama `isRes`, dan bidang yang diekstraksi dapat digunakan di sisa kueri Anda.

filter

Gunakan `filter` untuk mendapatkan peristiwa log yang cocok dengan satu atau beberapa kondisi.

Contoh: Filter peristiwa log menggunakan satu kondisi

Cuplikan kode menunjukkan contoh kueri yang mengembalikan semua peristiwa log di mana nilainya ***range*** lebih besar dari 3000. Kueri membatasi hasil hingga 20 peristiwa log dan mengurutkan peristiwa log berdasarkan `@timestamp` dan dalam urutan menurun.

```
fields @timestamp, @message
| filter (range>3000)
| sort @timestamp desc
| limit 20
```

Contoh: Filter peristiwa log menggunakan lebih dari satu kondisi

Anda dapat menggunakan kata kunci `and` dan `or` menggabungkan lebih dari satu kondisi.

Cuplikan kode menunjukkan contoh kueri yang mengembalikan peristiwa log di mana nilai untuk lebih besar dari 3000 dan nilai untuk ***range*** sama dengan ***accountId*** 123456789012. Kueri membatasi hasil hingga 20 peristiwa log dan mengurutkan peristiwa log berdasarkan `@timestamp` dan dalam urutan menurun.

```
fields @timestamp, @message
| filter (range>3000 and accountId=123456789012)
| sort @timestamp desc
```

```
| limit 20
```

Bidang yang diindeks dan perintah filter

Jika Anda telah membuat indeks bidang untuk grup log, Anda dapat memanfaatkan indeks bidang tersebut untuk membuat filter kueri Anda lebih efisien dan mengurangi volume yang dipindai. Misalnya, Anda telah membuat indeks bidang untuk `requestId`. Kemudian, kueri CloudWatch Logs Insights apa pun pada grup log yang menyertakan filter `requestId = value` atau filter `requestId IN [value, value, ...]` akan mencoba melewati proses peristiwa log yang diketahui tidak menyertakan bidang yang diindeks. Dengan mencoba memindai hanya peristiwa log yang diketahui berisi bidang yang diindeks itu, volume pemindaian dapat dikurangi dan kueri lebih cepat.

Untuk informasi selengkapnya tentang indeks bidang dan cara membuatnya, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Important

Hanya kueri dengan filter `fieldName = ...` dan filter `fieldName IN ...` akan mendapat manfaat dari peningkatan indeks lapangan. Kueri dengan filter `fieldName like` tidak menggunakan indeks dan selalu memindai semua peristiwa log di grup log yang dipilih.

Contoh: Temukan peristiwa log yang terkait dengan ID permintaan tertentu, menggunakan indeks

Contoh ini mengasumsikan bahwa Anda telah membuat indeks bidang pada `requestId`. Untuk grup log yang menggunakan indeks bidang ini, kueri akan memanfaatkan indeks bidang untuk mencoba memindai jumlah peristiwa log paling sedikit untuk menemukan peristiwa `requestId` dengan nilai 123456

```
fields @timestamp, @message
| filter requestId = "123456"
| limit 20
```

Kecocokan dan ekspresi reguler dalam perintah filter

Perintah filter mendukung penggunaan ekspresi reguler. Anda dapat menggunakan operator perbandingan berikut (`=`, `!=`, `<`, `<=`, `>`, `>=`) dan operator Boolean (`and`, `or`, `not`).

Anda dapat menggunakan kata kunci `in` untuk menguji keanggotaan set dan memeriksa elemen dalam array. Untuk memeriksa elemen dalam array, letakkan array setelahnya `in`. Anda dapat menggunakan operator Boolean `not` dengan `in`. Anda dapat membuat kueri yang digunakan `in` untuk mengembalikan peristiwa log di mana bidang adalah string yang cocok. Bidang harus string lengkap. Misalnya, cuplikan kode berikut menunjukkan kueri yang digunakan `in` untuk mengembalikan peristiwa log di mana bidang `logGroup` adalah string lengkap. `example_group`

```
fields @timestamp, @message
| filter logGroup in ["example_group"]
```

Anda dapat menggunakan frase kata kunci `like` dan `not like` untuk mencocokkan substring. Anda dapat menggunakan operator ekspresi reguler `=~` untuk mencocokkan substring. Untuk mencocokkan substring dengan `like` dan `not like`, lampirkan substring yang ingin Anda cocokkan dalam tanda kutip tunggal atau ganda. Anda dapat menggunakan pola ekspresi reguler dengan `like` dan `not like`. Untuk mencocokkan substring dengan operator ekspresi reguler, lampirkan substring yang ingin Anda cocokkan dalam garis miring maju. Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana Anda dapat mencocokkan substring menggunakan perintah `filter`

Contoh: Match substring

Contoh berikut mengembalikan peristiwa log yang `f1` berisi kata Pengecualian. Ketiga contoh tersebut peka terhadap huruf besar dan kecil.

Contoh pertama cocok dengan substring dengan `like`.

```
fields f1, f2, f3
| filter f1 like "Exception"
```

Contoh kedua cocok dengan substring dengan `like` dan pola ekspresi reguler.

```
fields f1, f2, f3
| filter f1 like /Exception/
```

Contoh ketiga cocok dengan substring dengan ekspresi reguler.

```
fields f1, f2, f3
| filter f1 =~ /Exception/
```

Contoh: Cocokkan substring dengan wildcard

Anda dapat menggunakan simbol periode (.) sebagai wildcard dalam ekspresi reguler untuk mencocokkan substring. Dalam contoh berikut, query mengembalikan kecocokan di mana nilai untuk f1 dimulai dengan stringServiceLog.

```
fields f1, f2, f3
| filter f1 like /ServiceLog./
```

Anda dapat menempatkan simbol tanda bintang setelah simbol periode (.*) untuk membuat kuantifier serakah yang mengembalikan kecocokan sebanyak mungkin. Misalnya, query berikut mengembalikan kecocokan di mana nilai untuk f1 tidak hanya dimulai dengan stringServiceLog, tetapi juga termasuk stringServiceLog.

```
fields f1, f2, f3
| filter f1 like /ServiceLog.*/
```

Kemungkinan kecocokan dapat diformat seperti berikut:

- ServiceLogSampleApiLogGroup
- SampleApiLogGroupServiceLog

Contoh: Kecualikan substring dari korek api

Contoh berikut menunjukkan query yang mengembalikan peristiwa log di mana f1 tidak mengandung kata Exception. Contohnya adalah case sensitive.

```
fields f1, f2, f3
| filter f1 not like "Exception"
```

Contoh: Cocokkan substring dengan pola case-insensitive

Anda dapat mencocokkan substring yang tidak peka huruf besar/kecil dengan ekspresi like reguler. Tempatkan parameter berikut (?i) sebelum substring yang ingin Anda cocokkan. Contoh berikut menunjukkan query yang mengembalikan peristiwa log yang f1 berisi kata Pengecualian atau pengecualian.

```
fields f1, f2, f3
| filter f1 like /(?!i)Exception/
```


FilterIndex

Gunakan `filterIndex` untuk mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri. Untuk grup log ini yang diindeks di bidang ini, selanjutnya mengoptimalkan kueri dengan melewati grup log yang tidak memiliki peristiwa log yang berisi bidang yang ditentukan dalam kueri untuk bidang yang diindeks. Selanjutnya mengurangi volume yang dipindai dengan mencoba memindai hanya peristiwa log dari grup log ini yang cocok dengan nilai yang ditentukan dalam kueri untuk indeks bidang ini. Untuk informasi selengkapnya tentang indeks bidang dan cara membuatnya, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Menggunakan `filterIndex` dengan bidang yang diindeks dapat membantu Anda menanyakan grup log yang menyertakan petabyte data log secara efisien dengan membatasi ruang pencarian aktual untuk grup log dan peristiwa log yang memiliki indeks bidang.

Misalnya, Anda telah membuat indeks bidang untuk `IPAddress` beberapa grup log di akun Anda. Anda kemudian dapat membuat kueri berikut dan memilih untuk menanyakan semua grup log di akun untuk menemukan peristiwa log yang menyertakan nilai `198.51.100.0` di `IPAddress` bidang.

```
fields @timestamp, @message
| filterIndex IPAddress = "198.51.100.0"
| limit 20
```

`filterIndex`Perintah menyebabkan kueri ini mencoba melewati semua grup log yang tidak diindeks. `IPAddress` Selain itu, dalam grup log yang diindeks, kueri melewati peristiwa log yang memiliki `IPAddress` bidang tetapi tidak diamati `198.51.100.0` sebagai nilai untuk bidang tersebut.

Gunakan `IN` operator untuk memperluas hasil ke salah satu dari beberapa nilai untuk bidang yang diindeks. Contoh berikut menemukan log peristiwa yang menyertakan nilai `198.51.100.0` atau `198.51.100.1` di `IPAddress` bidang.

```
fields @timestamp, @message
| filterIndex IPAddress in ["198.51.100.0", "198.51.100.1"]
| limit 20
```

CloudWatch Log menyediakan indeks bidang default untuk semua grup log di kelas log Standar. Indeks bidang default secara otomatis tersedia untuk bidang berikut:

- `@logStream`
- `@aws.region`

- `@aws.account`
- `@source.log`
- `@data_source_name`
- `@data_source_type`
- `@data_format`
- `traceId`
- `severityText`
- `attributes.session.id`

CloudWatch Log menyediakan indeks bidang default untuk nama sumber data tertentu dan kombinasi tipe juga. Indeks bidang default secara otomatis tersedia untuk kombinasi nama dan jenis sumber data berikut:

Nama dan Jenis Sumber Data	Indeks Bidang Default
<code>amazon_vpc.flow</code>	<code>action</code> <code>logStatus</code> <code>region</code> <code>flowDirection</code> <code>type</code>
<code>amazon_route53.resolver_query</code>	<code>query_type</code> <code>transport</code> <code>rcode</code>
<code>aws_waf.access</code>	<code>action</code> <code>httpRequest.country</code>
<code>aws_cloudtrail.data</code>	<code>eventSource</code>
<code>aws_cloudtrail.management</code>	<code>eventName</code>

Nama dan Jenis Sumber Data	Indeks Bidang Default
	awsRegion
	userAgent
	errorCode
	eventType
	managementEvent
	readOnly
	eventCategory
	requestId

Indeks bidang default merupakan tambahan dari indeks bidang kustom apa pun yang Anda tentukan dalam kebijakan Anda. Indeks bidang default tidak dihitung terhadap kuota [indeks bidang](#) Anda.

FilterIndex dibandingkan dengan filter

Untuk mengilustrasikan perbedaan antara `filterIndex` dan `filter`, pertimbangkan contoh kueri berikut. Misalnya Anda telah membuat indeks bidang untuk `IPAddress`, untuk empat grup log Anda, tetapi tidak untuk grup log kelima. Penggunaan kueri berikut `filterIndex` akan melewati pemindaian grup log yang tidak memiliki bidang yang diindeks. Untuk setiap grup log yang diindeks, ia mencoba memindai hanya peristiwa log yang memiliki bidang yang diindeks, dan juga hanya mengembalikan hasil setelah indeks bidang dibuat.

```
fields @timestamp, @message
| filterIndex IPAddress = "198.51.100.0"
| limit 20
```

Sebaliknya, jika Anda menggunakan `filter` bukan `filterIndex` untuk kueri dari lima grup log yang sama, kueri akan mencoba memindai tidak hanya peristiwa log yang berisi nilai dalam grup log yang diindeks, tetapi juga akan memindai grup log kelima yang tidak diindeks, dan itu akan memindai setiap peristiwa log di grup log kelima itu.

```
fields @timestamp, @message
```

```
| filter IPAddress = "198.51.100.0"  
| limit 20
```

SUMBER

Termasuk **SOURCE** dalam kueri adalah cara yang berguna untuk menentukan sumber and/or data grup log untuk disertakan dalam kueri saat Anda menggunakan API AWS CLI atau untuk membuat kueri. **SOURCE**Perintah hanya didukung di API AWS CLI dan, bukan di CloudWatch konsol. Saat Anda menggunakan CloudWatch konsol untuk memulai kueri, Anda menggunakan antarmuka konsol untuk menentukan grup log.

Grup log kueri

Untuk digunakan **SOURCE** untuk menentukan grup log untuk query, Anda dapat menggunakan kata kunci berikut:

- **namePrefix** menjalankan kueri terhadap grup log yang memiliki nama yang dimulai dengan string yang Anda tentukan. Jika Anda menghilangkan ini, semua grup log ditanyakan.

Anda dapat memasukkan sebanyak lima awalan dalam daftar.

- **accountId** menjalankan kueri terhadap grup log di AWS akun yang ditentukan. Ini hanya berfungsi ketika Anda menjalankan kueri di akun pemantauan. Jika Anda menghilangkan ini, defaultnya adalah menanyakan semua akun sumber tertaut dan akun pemantauan saat ini. Untuk informasi lebih lanjut tentang observabilitas lintas akun, lihat observabilitas [CloudWatch lintas](#) akun.

Anda dapat memasukkan sebanyak 20 pengenalan akun dalam daftar.

- **logGroupClass** menjalankan kueri terhadap grup log yang berada di kelas log tertentu, baik Standard atau Infrequent Access. Jika Anda menghilangkan ini, default kelas log Standar digunakan. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

Karena Anda dapat menentukan sejumlah besar grup log untuk kueri dengan cara ini, sebaiknya gunakan **SOURCE** hanya dalam kueri yang memanfaatkan indeks bidang yang telah Anda buat. Untuk informasi selengkapnya tentang bidang pengindeksan di grup log, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#)

Contoh berikut memilih semua grup log di akun. Jika ini adalah akun pemantauan maka grup log di seluruh pemantauan dan semua akun sumber akan dipilih. Jika jumlah grup log melebihi 10.000 maka Anda akan melihat kesalahan yang mendorong Anda untuk mengurangi jumlah grup log dengan menggunakan metode pemilihan grup log yang berbeda.

```
SOURCE logGroups()
```

Contoh berikut memilih grup log di akun 111122223333 sumber. Jika Anda memulai kueri di akun pemantauan dalam pengamatan CloudWatch lintas akun, grup log di semua akun sumber dan di akun pemantauan dipilih secara default.

```
SOURCE logGroups(accountIdentifiers:['111122223333'])
```

Contoh berikutnya memilih grup log berdasarkan awalan nama.

```
SOURCE logGroups(namePrefix: ['namePrefix1', 'namePrefix2'])
```

Contoh berikut memilih semua grup log di kelas log Akses Jarang. Jika Anda tidak menyertakan `class` pengenalan, kueri hanya berlaku untuk grup log di kelas log Standar, yang merupakan default.

```
SOURCE logGroups(class: ['INFREQUENT_ACCESS'])
```

Contoh berikutnya memilih grup log di akun 111122223333 yang dimulai dengan awalan nama tertentu dan berada di kelas log Standar. Kelas tidak disebutkan dalam perintah karena Standar adalah nilai kelas log default.

```
SOURCE logGroups(accountIdentifiers:['111122223333'], namePrefix: ['namePrefix1',  
'namePrefix2'])
```

Contoh terakhir menampilkan cara menggunakan SOURCE perintah dengan `start-query` AWS CLI perintah.

```
aws logs start-query  
--region us-east-1  
--start-time 1729728200  
--end-time 1729728215  
--query-string "SOURCE logGroups(namePrefix: ['Query']) | fields @message | limit 5"
```

Sumber data kueri

Untuk digunakan SOURCE untuk menentukan sumber data untuk query, Anda dapat menggunakan `dataSource` kata kunci. Anda dapat memasukkan sebanyak sepuluh sumber data dalam daftar.

Contoh berikut memilih sumber `amazon_vpc.flow` data.

```
SOURCE dataSource(['amazon_vpc.flow'])
```

Contoh berikut memilih sumber `amazon_vpc.flow` data dan membatasi grup log berdasarkan awalan nama grup log.

```
SOURCE dataSource(['amazon_vpc.flow']) logGroups(namePrefix: ['namePrefix1'])
```

pola

Gunakan `pattern` untuk secara otomatis mengelompokkan data log Anda ke dalam pola.

Pola adalah struktur teks bersama yang berulang di antara bidang log Anda. Anda dapat menggunakan `pattern` untuk memunculkan tren yang muncul, memantau kesalahan yang diketahui, dan mengidentifikasi jalur log yang sering terjadi atau berbiaya tinggi. CloudWatch Logs Insights juga menyediakan pengalaman konsol yang dapat Anda gunakan untuk menemukan dan menganalisis pola lebih lanjut dalam peristiwa log Anda. Untuk informasi selengkapnya, lihat [Analisis pola](#).

Karena `pattern` perintah secara otomatis mengidentifikasi pola umum, Anda dapat menggunakannya sebagai titik awal untuk mencari dan menganalisis log Anda. Anda juga dapat menggabungkan `pattern` dengan [filter](#), [parse](#), atau [sort](#) perintah untuk mengidentifikasi pola dalam kueri yang lebih disempurnakan.

Masukan Perintah Pola

`pattern` [Perintah mengharapkan salah satu input berikut: @message bidang, bidang yang diekstraksi yang dibuat menggunakan `parse` perintah, atau string yang dimanipulasi menggunakan satu atau beberapa fungsi String.](#)

Jika CloudWatch Log tidak dapat menyimpulkan jenis data yang diwakili oleh token dinamis, menampilkannya sebagai `<Token- number>`, dan *number* menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

Contoh umum token dinamis termasuk kode kesalahan, alamat IP, stempel waktu, dan permintaan. IDs

Output Perintah Pola

`pattern`Perintah menghasilkan output berikut:

- `@pattern`: Struktur teks bersama yang berulang di antara bidang peristiwa log Anda. Bidang yang bervariasi dalam suatu pola, seperti ID permintaan atau stempel waktu, diwakili oleh token. Jika CloudWatch Log dapat menentukan jenis data yang diwakili oleh token dinamis, itu akan menampilkan token sebagai `<string-number>`. `string`Ini adalah deskripsi dari jenis data yang diwakili oleh token. `number`Menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

CloudWatch Log menetapkan bagian string dari nama berdasarkan analisis konten peristiwa log yang berisi itu.

Jika CloudWatch Log tidak dapat menyimpulkan jenis data yang diwakili oleh token dinamis, menampilkannya sebagai `<Token- number >`, dan `number` menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

Misalnya, `[INFO] Request time: <Time-1> ms` adalah output potensial untuk pesan log `[INFO] Request time: 327 ms`.

- `@ratio`: Rasio peristiwa log dari periode waktu yang dipilih dan grup log tertentu yang cocok dengan pola yang diidentifikasi. Misalnya, jika setengah dari peristiwa log dalam grup log yang dipilih dan periode waktu cocok dengan pola, `@ratio` kembali `0.50`
- `@sampleCount`: Hitungan jumlah peristiwa log dari periode waktu yang dipilih dan grup log tertentu yang cocok dengan pola yang diidentifikasi.
- `@severityLabel`: Tingkat keparahan atau tingkat log, yang menunjukkan jenis informasi yang terkandung dalam log. Contohnya: `Error`, `Warning`, `Info`, atau `Debug`.

Contoh

Perintah berikut mengidentifikasi log dengan struktur serupa dalam grup log tertentu selama rentang waktu yang dipilih, mengelompokkannya berdasarkan pola dan hitungan

```
pattern @message
```

`pattern`Perintah dapat digunakan dalam kombinasi dengan [filter](#) perintah

```
filter @message like /ERROR/
| pattern @message
```

patternPerintah dapat digunakan dengan [sort](#) perintah [parse](#) dan

```
filter @message like /ERROR/
| parse @message 'Failed to do: *' as cause
| pattern cause
| sort @sampleCount asc
```

perbedaan

Membandingkan peristiwa log yang ditemukan dalam periode waktu yang Anda minta dengan peristiwa log dari periode waktu sebelumnya dengan panjang yang sama. Dengan cara ini, Anda dapat mencari tren dan menemukan apakah peristiwa log tertentu baru.

Tambahkan pengubah ke diff perintah untuk menentukan periode waktu yang ingin Anda bandingkan dengan:

- `diff` membandingkan peristiwa log dalam rentang waktu yang dipilih saat ini dengan peristiwa log dari rentang waktu sebelumnya.
- `diff previousDay` membandingkan peristiwa log dalam rentang waktu yang dipilih saat ini dengan peristiwa log dari waktu yang sama pada hari sebelumnya.
- `diff previousWeek` membandingkan peristiwa log dalam rentang waktu yang dipilih saat ini dengan peristiwa log dari waktu yang sama minggu sebelumnya.
- `diff previousMonth` membandingkan peristiwa log dalam rentang waktu yang dipilih saat ini dengan peristiwa log dari waktu yang sama pada bulan sebelumnya.

Untuk informasi selengkapnya, lihat [Bandingkan \(diff\) dengan rentang waktu sebelumnya](#).

mengurai

Gunakan `parse` untuk mengekstrak data dari bidang log dan membuat bidang yang diekstraksi yang dapat Anda proses dalam kueri Anda. Jika peristiwa log tidak cocok dengan pola yang ditentukan, Anda masih melihatnya di hasil, tetapi tanpa bidang yang diekstraksi. **parse** mendukung mode glob menggunakan wildcard, dan ekspresi reguler. Untuk informasi tentang sintaks ekspresi reguler, lihat [Sintaks ekspresi reguler \(regex\) yang didukung](#).

Anda dapat mengurai bidang JSON bersarang dengan ekspresi reguler.

Contoh: Mengurai bidang JSON bersarang

Cuplikan kode menunjukkan cara mengurai peristiwa log JSON yang telah diratakan selama konsumsi.

```
{'fieldsA': 'logs', 'fieldsB': [{'fA': 'a1'}, {'fA': 'a2'}]}
```

Cuplikan kode menunjukkan kueri dengan ekspresi reguler yang mengekstrak nilai `fieldsB` untuk `fieldsA` dan membuat bidang yang diekstraksi dan. `fld` array

```
parse @message "'fieldsA': '*', 'fieldsB': ['*']" as fld, array
```

Dinamakan menangkap kelompok

Bila Anda menggunakan **parse** dengan ekspresi reguler, Anda dapat menggunakan grup penangkap bernama untuk menangkap pola ke dalam bidang. Sintaksnya adalah `parse @message (? <Name>pattern)`

Contoh berikut menggunakan grup menangkap pada log aliran VPC untuk mengekstrak ENI ke dalam bidang bernama. `NetworkInterface`

```
parse @message /(?(<NetworkInterface>eni-.*?) / | display NetworkInterface, @message
```

Note

Peristiwa log JSON diratakan selama konsumsi. Saat ini, mengurai bidang JSON bersarang dengan ekspresi glob tidak didukung. Anda hanya dapat mengurai peristiwa log JSON yang menyertakan tidak lebih dari 200 bidang peristiwa log. Saat Anda mengurai bidang JSON bersarang, Anda harus memformat ekspresi reguler dalam kueri agar sesuai dengan format peristiwa log JSON Anda.

Contoh perintah parse

Gunakan ekspresi glob untuk mengekstrak bidang **@user**, **@method**, dan **@latency** dari bidang log **@message** dan kembalikan latensi rata-rata untuk setiap kombinasi unik dan. **@method @user**

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

Gunakan ekspresi reguler untuk mengekstrak bidang `@user2`, `@method2`, dan `@latency2` dari bidang log `@message` dan kembalikan latensi rata-rata untuk setiap kombinasi unik `@method2` dan `@user2`.

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

Mengekstrak bidang **loggingTime**, **loggingType** dan **loggingMessage**, memfilter ke log peristiwa yang berisi **ERROR** atau **INFO** string, dan kemudian hanya menampilkan **loggingMessage** dan **loggingType** bidang untuk peristiwa yang berisi **ERROR** string.

```

FIELDS @message
  | PARSE @message "*" [*] "*" as loggingTime, loggingType, loggingMessage
  | FILTER loggingType IN ["ERROR", "INFO"]
  | DISPLAY loggingMessage, loggingType = "ERROR" as isError

```

menyortir

Gunakan `sort` untuk menampilkan peristiwa log dalam urutan ascending (`asc`) atau descending (`desc`) dengan bidang tertentu. Anda dapat menggunakan ini dengan `limit` perintah untuk membuat kueri “N atas” atau “N bawah”.

Algoritma penyortiran adalah versi terbaru dari penyortiran alami. Jika Anda mengurutkan dalam urutan menaik, logika berikut digunakan.

- Semua nilai non-angka datang sebelum semua nilai angka. Nilai angka adalah nilai yang hanya mencakup angka, bukan campuran angka dan karakter lainnya.
- Untuk nilai non-angka, algoritme mengelompokkan karakter numerik berurutan dan karakter alfabet berurutan ke dalam potongan terpisah untuk perbandingan. Ini memisahkan bagian non-numerik dengan nilai Unicode mereka, dan mengurutkan bagian numerik berdasarkan panjangnya terlebih dahulu dan kemudian dengan nilai numeriknya.

Untuk informasi selengkapnya tentang urutan Unicode, lihat [Daftar karakter Unicode](#).

Misalnya, berikut ini adalah hasil dari semacam dalam urutan menaik.

[illegible]

Satuan waktu dan singkatan berikut didukung dengan `bin` fungsi tersebut. Untuk semua unit dan singkatan yang menyertakan lebih dari satu karakter, menambahkan `s` ke pluralisasi didukung. Jadi keduanya `hr` dan `hrs` bekerja untuk menentukan jam.

- `millisecond ms msec`
- `second s sec`
- `minute m min`
- `hour h hr`
- `day d`
- `week w`
- `month mo mon`
- `quarter q qtr`
- `year y yr`

Topik

- [Visualisasikan data deret waktu](#)
- [Visualisasikan data log yang dikelompokkan berdasarkan bidang](#)
- [Gunakan beberapa perintah statistik dalam satu kueri](#)
- [Fungsi untuk digunakan dengan statistik](#)

Visualisasikan data deret waktu

Visualisasi deret waktu dapat digunakan dengan kueri yang memiliki karakteristik berikut:

- Kueri berisi satu atau beberapa fungsi agregasi. Untuk informasi selengkapnya, lihat [Aggregation Functions in the Stats Command](#).
- Kueri menggunakan fungsi `bin()` untuk mengelompokkan data dengan satu bidang.

Kueri-kueri ini dapat menghasilkan diagram garis, diagram area bertumpuk, diagram batang, dan diagram lingkaran.

Contoh

Untuk tutorial lengkap, lihat [the section called “Tutorial: Jalankan kueri yang menghasilkan visualisasi deret waktu”](#).

Berikut adalah contoh kueri lain yang dapat digunakan untuk visualisasi deret waktu.

Kueri berikut menghasilkan visualisasi nilai rata-rata bidang `myfield1`, dengan titik data yang dibuat setiap lima menit. Setiap titik data adalah agregasi dari rata-rata nilai `myfield1` dari log lima menit sebelumnya.

```
stats avg(myfield1) by bin(5m)
```

Kueri berikut menghasilkan visualisasi dari tiga nilai berdasarkan pada bidang-bidang yang berbeda, dengan titik data yang dibuat setiap lima menit. Visualisasi dihasilkan karena kueri berisi fungsi agregat dan menggunakan `bin()` sebagai bidang pengelompokan.

```
stats avg(myfield1), min(myfield2), max(myfield3) by bin(5m)
```

Bagan garis dan batasan bagan area bertumpuk

Pertanyaan yang membuat agregat informasi entri log, tetapi tidak menggunakan fungsi `bin()` dapat menghasilkan diagram batang. Namun, kueri tidak dapat menghasilkan diagram garis atau diagram area bertumpuk. Untuk informasi selengkapnya tentang tipe kueri ini, lihat [the section called “Visualisasikan data log yang dikelompokkan berdasarkan bidang”](#).

Visualisasikan data log yang dikelompokkan berdasarkan bidang

Anda dapat menghasilkan diagram batang untuk kueri yang menggunakan fungsi `stats` dan satu atau beberapa fungsi agregasi. Untuk informasi selengkapnya, lihat [Aggregation Functions in the Stats Command](#).

Untuk melihat visualisasi, jalankan kueri Anda. Lalu pilih tab Visualization (Visualisasi), pilih panah di sebelah Line (Garis), dan pilih Bar (Batang). Visualisasi dibatasi hingga 100 batang dalam diagram batang.

Contoh

Untuk tutorial lengkap, lihat [the section called “Tutorial: Jalankan kueri yang menghasilkan visualisasi yang dikelompokkan berdasarkan bidang log”](#). Paragraf berikut mencakup lebih banyak contoh kueri untuk visualisasi berdasarkan bidang.

Kueri log alur VPC berikut menemukan jumlah rata-rata byte yang ditransfer per sesi untuk setiap alamat tujuan.

```
stats avg(bytes) by dstAddr
```

Anda juga dapat menghasilkan diagram yang mencakup lebih dari satu batang untuk setiap nilai yang dihasilkan. Misalnya, kueri log alur VPC berikut menemukan jumlah rata-rata dan maksimum byte yang ditransfer per sesi untuk setiap alamat tujuan.

```
stats avg(bytes), max(bytes) by dstAddr
```

Kueri berikut menemukan jumlah log kueri Amazon Route 53 untuk setiap jenis kueri.

```
stats count(*) by queryType
```

Gunakan beberapa perintah statistik dalam satu kueri

Anda dapat menggunakan sebanyak dua **stats** perintah dalam satu kueri. Ini memungkinkan Anda untuk melakukan agregasi tambahan pada output agregasi pertama.

Contoh: Query dengan dua **stats** perintah

Misalnya, kueri berikut pertama-tama menemukan total volume lalu lintas di tempat sampah 5 menit, kemudian menghitung volume lalu lintas tertinggi, terendah, dan rata-rata di antara tempat sampah 5 menit tersebut.

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length)/1024/1024 as logs_mb BY bin(5m)
| STATS max(logs_mb) AS peak_ingest_mb,
      min(logs_mb) AS min_ingest_mb,
      avg(logs_mb) AS avg_ingest_mb
```

Contoh: Menggabungkan beberapa perintah statistik dengan fungsi lain seperti **filter**,, **fields bin**

Anda dapat menggabungkan dua **stats** perintah dengan perintah lain seperti **filter** dan **fields** dalam satu kueri. Misalnya, kueri berikut menemukan jumlah alamat IP yang berbeda dalam sesi dan menemukan jumlah sesi berdasarkan platform klien, memfilter alamat IP tersebut, dan akhirnya menemukan rata-rata permintaan sesi per platform klien.

```
STATS count_distinct(client_ip) AS session_ips,
      count(*) AS requests BY session_id, client_platform
| FILTER session_ips > 1
```

```
| STATS count(*) AS multiple_ip_sessions,
      sum(requests) / count(*) AS avg_session_requests BY client_platform
```

Anda dapat menggunakan `bin` dan `dateceil` berfungsi dalam kueri dengan beberapa `stats` perintah. Misalnya, kueri berikut pertama-tama menggabungkan pesan menjadi blok 5 menit, kemudian menggabungkan blok 5 menit tersebut menjadi blok 10 menit dan menghitung volume lalu lintas tertinggi, terendah, dan rata-rata dalam setiap blok 10 menit.

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length) / 1024 / 1024 AS logs_mb BY BIN(5m) as @t
| STATS max(logs_mb) AS peak_ingest_mb,
      min(logs_mb) AS min_ingest_mb,
      avg(logs_mb) AS avg_ingest_mb BY dateceil(@t, 10m)
```

Catatan dan batasan

Kueri dapat memiliki maksimal dua `stats` perintah. Kuota ini tidak dapat diubah.

Jika Anda menggunakan `limit` perintah `sort` atau, itu harus muncul setelah `stats` perintah kedua. Jika sebelum `stats` perintah kedua, kueri tidak valid.

Ketika kueri memiliki dua `stats` perintah, sebagian hasil dari kueri tidak mulai ditampilkan sampai `stats` agregasi pertama selesai.

Dalam `stats` perintah kedua dalam satu kueri, Anda hanya dapat merujuk ke bidang yang didefinisikan dalam `stats` perintah pertama. Misalnya, kueri berikut tidak valid karena `@message` bidang tidak akan tersedia setelah `stats` agregasi pertama.

```
FIELDS @message
| STATS SUM(Fault) by Operation
# You can only reference `SUM(Fault)` or Operation at this point
| STATS MAX(strlen(@message)) AS MaxMessageSize # Invalid reference to @message
```

Bidang apa pun yang Anda referensikan setelah `stats` perintah pertama harus didefinisikan dalam `stats` perintah pertama itu.

```
STATS sum(x) as sum_x by y, z
| STATS max(sum_x) as max_x by z
# You can only reference `max(sum_x)`, max_x or z at this point
```

⚠ Important

`bin` Fungsi selalu secara implisit menggunakan bidang. `@timestamp` Ini berarti bahwa Anda tidak dapat menggunakan `bin` dalam `stats` perintah kedua tanpa menggunakan `stats` perintah pertama untuk menyebarkan `timestamp` bidang. Misalnya, kueri berikut tidak valid.

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length) AS ingested_bytes BY @logStream
| STATS avg(ingested_bytes) BY bin(5m) # Invalid reference to @timestamp field
```

Sebagai gantinya, tentukan `@timestamp` bidang di `stats` perintah pertama, dan kemudian Anda dapat menggunakannya dengan `dateceil` `stats` perintah kedua seperti pada contoh berikut.

```
FIELDS strlen(@message) AS message_length
| STATS sum(message_length) AS ingested_bytes, max(@timestamp) as @t BY @logStream
| STATS avg(ingested_bytes) BY dateceil(@t, 5m)
```

Fungsi untuk digunakan dengan statistik

CloudWatch Logs Insights mendukung fungsi agregasi statistik dan fungsi non-agregasi statistik.

Gunakan fungsi `statsaggregation` dalam `stats` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>avg(fieldName: NumericLogField)</code>	nomor	Rata-rata nilai di bidang yang ditentukan.
<code>count()</code> <code>count(fieldName: LogField)</code>	nomor	Menghitung log acara. <code>count()</code> (atau <code>count(*)</code>) menghitung semua kejadian yang dikembalikan oleh kueri, sementara <code>count(fieldName)</code> menghitung semua catatan yang menyertakan nama bidang yang ditentukan.

Fungsi	Tipe hasil	Deskripsi
<code>count_distinct(fieldName: LogField)</code>	nomor	Mengembalikan jumlah nilai unik untuk bidang. Jika bidang memiliki kardinalitas yang sangat tinggi (mengandung banyak nilai unik), nilai yang dikembalikan oleh <code>count_distinct</code> hanyalah sebuah perkiraan.
<code>max(fieldName: LogField)</code>	LogFieldValue	Nilai maksimum untuk bidang log ini dalam log yang dikueri.
<code>min(fieldName: LogField)</code>	LogFieldValue	Nilai minimum untuk bidang log ini dalam log yang dikueri.
<code>pct(fieldName: LogFieldValue, percent: number)</code>	LogFieldValue	Persentil menunjukkan posisi relatif dari nilai dalam rangkaian data. Misalnya, <code>pct(@duration, 95)</code> mengembalikan nilai <code>@duration</code> di mana 95 persen dari nilai <code>@duration</code> lebih rendah dari nilai ini, dan 5 persen lebih tinggi dari nilai ini.
<code>stddev(fieldName: NumericLogField)</code>	nomor	Standar deviasi di bidang yang ditentukan.
<code>sum(fieldName: NumericLogField)</code>	nomor	Jumlah nilai di bidang yang ditentukan.

Statistik fungsi non-agregasi

Gunakan fungsi non-agregasi dalam `stats` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>earliest(fieldName: LogField)</code>	LogField	Mengembalikan nilai <code>fieldName</code> dari log acara yang memiliki stempel waktu paling awal dalam log yang dikueri.

Fungsi	Tipe hasil	Deskripsi
<code>latest(fieldName: LogField)</code>	LogField	Mengembalikan nilai <code>fieldName</code> dari log acara yang memiliki stempel waktu paling akhir dalam log yang dikueri.
<code>sortsFirst(fieldName: LogField)</code>	LogField	Mengembalikan nilai <code>fieldName</code> yang ada di urutan pertama dalam log yang dikueri.
<code>sortsLast(fieldName: LogField)</code>	LogField	Mengembalikan nilai <code>fieldName</code> yang ada di urutan terakhir dalam log yang dikueri.

batasan

Gunakan `limit` untuk menentukan jumlah peristiwa log yang Anda ingin kueri Anda kembalikan. Jika Anda menghilangkan `limit`, kueri akan mengembalikan sebanyak 10.000 peristiwa log dalam hasil.

Misalnya, contoh berikut hanya mengembalikan 25 peristiwa log terbaru

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

dedup

Gunakan `dedup` untuk menghapus hasil duplikat berdasarkan nilai tertentu di bidang yang Anda tentukan. Anda dapat menggunakan `dedup` dengan satu atau lebih bidang. Jika Anda menentukan satu bidang dengan `dedup`, hanya satu peristiwa log yang dikembalikan untuk setiap nilai unik bidang itu. Jika Anda menentukan beberapa bidang, maka satu peristiwa log dikembalikan untuk setiap kombinasi nilai unik untuk bidang tersebut.

Duplikat dibuang berdasarkan urutan pengurutan, dengan hanya hasil pertama dalam urutan pengurutan yang disimpan. Kami menyarankan Anda mengurutkan hasil Anda sebelum memasukkannya melalui `dedup` perintah. Jika hasilnya tidak diurutkan sebelum dijalankan `dedup`, maka urutan urutan menurun default yang digunakan `@timestamp` digunakan.

Nilai nol tidak dianggap duplikat untuk evaluasi. Peristiwa log dengan nilai null untuk salah satu bidang tertentu dipertahankan. Untuk menghilangkan bidang dengan nilai nol, gunakan **filter** menggunakan `isPresent(field)` fungsi.

Satu-satunya perintah kueri yang dapat Anda gunakan dalam kueri setelah dedup perintah adalah `limit`.

Saat Anda menggunakan dedup kueri, konsol akan menampilkan pesan seperti Menampilkan X catatan Y, di mana X adalah jumlah hasil deduplikasi dan Y adalah jumlah total rekaman yang cocok sebelum deduplikasi. Ini menunjukkan bahwa catatan duplikat telah dihapus dan tidak berarti bahwa data hilang.

Contoh: Lihat hanya peristiwa log terbaru untuk setiap nilai unik bidang bernama **server**

Contoh berikut menampilkan `timestamp`, `server`, `severity`, dan `message` bidang hanya untuk acara terbaru untuk setiap nilai unik `server`.

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server
```

Untuk lebih banyak contoh kueri Wawasan CloudWatch Log, lihat [Kueri umum](#)

membuka kedok

Gunakan `unmask` untuk menampilkan semua konten peristiwa log yang memiliki beberapa konten yang disembunyikan karena kebijakan perlindungan data. Untuk menggunakan perintah ini, Anda harus memiliki `logs:Unmask` izin.

Untuk informasi selengkapnya tentang perlindungan data di grup log, lihat [Membantu melindungi data log sensitif dengan masking](#).

`unnest`

Gunakan `unnest` untuk meratakan daftar yang diambil sebagai masukan untuk menghasilkan beberapa catatan dengan satu catatan untuk setiap elemen dalam daftar. Berdasarkan jumlah item yang terkandung dalam bidang, perintah ini membuang catatan saat ini dan menghasilkan catatan baru. Setiap catatan mencakup `unnested_field`, yang mewakili item. Semua bidang lainnya berasal dari catatan asli.

Input untuk `unnest` adalah `LIST`, yang berasal dari `jsonParse` fungsi. Untuk informasi selengkapnya, lihat [Jenis struktur](#). Tipe lain, seperti `MAP`, `String` dan `numbers`, diperlakukan sebagai daftar dengan satu item di `unnest`.

Struktur perintah

Contoh berikut menjelaskan format perintah ini.

```
unnest field into unnested_field
```

Kueri contoh

Contoh berikut mem-parsing string objek JSON dan memperluas daftar peristiwa bidang.

```
fields jsonParse(@message) as json_message  
| unnest json_message.events into event  
| display event.name
```

Peristiwa log untuk kueri contoh ini bisa berupa string JSON sebagai berikut:

```
{  
  "events": [  
    {  
      "name": "exception"  
    },  
    {  
      "name": "user action"  
    }  
  ]  
}
```

Dalam hal ini, kueri sampel menghasilkan dua catatan dalam hasil kueri, satu dengan event.name as exception dan yang lain dengan event.name sebagai tindakan pengguna

Kueri contoh

Contoh berikut meratakan daftar dan kemudian menyaring item.

```
fields jsonParse(@message) as js  
| unnest js.accounts into account  
| filter account.type = "internal"
```

Kueri contoh

Contoh berikut meratakan daftar untuk agregasi.

```
fields jsonParse(trimmedData) as accounts
| unnest accounts into account
| stats sum(account.droppedSpans) as n by account.accountId
| sort n desc
| limit 10
```

pencarian

Gunakan lookup untuk memperkaya hasil kueri Anda dengan data referensi dari tabel pencarian. Tabel pencarian berisi data CSV yang Anda unggah ke Amazon CloudWatch Logs. Saat kueri berjalan, lookup perintah mencocokkan bidang dalam peristiwa log Anda terhadap bidang di tabel pencarian dan menambahkan bidang keluaran yang ditentukan ke hasil.

Gunakan tabel pencarian untuk skenario pengayaan data seperti memetakan pengguna IDs ke detail pengguna, kode produk ke informasi produk, atau kode kesalahan untuk deskripsi kesalahan.

Membuat dan mengelola tabel pencarian

Sebelum Anda dapat menggunakan lookup perintah dalam kueri, Anda harus membuat tabel pencarian. Anda dapat membuat dan mengelola tabel pencarian dari CloudWatch konsol atau dengan menggunakan Amazon CloudWatch Logs API.

Untuk membuat tabel pencarian (konsol)

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Pengaturan, lalu pilih tab Log.
3. Gulir ke tabel Pencarian dan pilih Kelola.
4. Pilih Buat tabel pencarian.
5. Masukkan nama untuk tabel pencarian. Nama hanya dapat berisi karakter alfanumerik, tanda hubung, dan garis bawah.
6. (Opsional) Masukkan deskripsi.
7. Unggah file CSV. File harus menyertakan baris header dengan nama kolom, gunakan pengkodean UTF-8, dan tidak melebihi 10 MB.
8. (Opsional) Tentukan AWS KMS kunci untuk mengenkripsi data tabel.
9. Pilih Buat.

Setelah membuat tabel pencarian, Anda dapat melihatnya di editor kueri Wawasan CloudWatch Log. Pilih tab Lookup tables untuk menelusuri tabel yang tersedia dan bidangnya.

Untuk memperbarui tabel pencarian, pilih tabel dan pilih Tindakan, Perbarui. Unggah file CSV baru untuk menggantikan semua konten yang ada. Untuk menghapus tabel pencarian, pilih Tindakan, Hapus.

 Note

Anda dapat membuat hingga 100 tabel pencarian per akun per Wilayah AWS akun. File CSV dapat mencapai 10 MB. Anda juga dapat mengelola tabel pencarian dengan menggunakan Amazon CloudWatch Logs API. Untuk informasi selengkapnya, lihat [CreateLookupTable](#) di Referensi API Amazon CloudWatch Logs.

 Note

Jika tabel pencarian dienkripsi dengan kunci KMS, penelepon harus memiliki kms:Decrypt izin pada kunci (kunci KMS yang digunakan untuk mengenkripsi tabel pencarian) untuk menggunakan StartQuery API dengan kueri yang mereferensikan tabel pencarian itu. Untuk informasi selengkapnya, lihat [Enkripsi tabel pencarian di CloudWatch Log menggunakan AWS Key Management Service](#).

Sintaks kueri untuk pencarian

Struktur perintah

Berikut ini menunjukkan format perintah ini.

```
lookup table lookup-field as log-field [,...] output-mode output-field [,...]
```

Perintah menggunakan argumen berikut:

- *table*— Nama tabel pencarian yang akan digunakan.
- *lookup-field*— Bidang di tabel pencarian untuk dicocokkan.
- *log-field*— Bidang di acara log Anda untuk dicocokkan. Pertandingannya tepat dan peka huruf besar/kecil.

- *lookup-field* as *log-field* [, ...]— Anda dapat menentukan beberapa pasangan bidang kecocokan yang dipisahkan dengan koma. Ketika beberapa pasangan ditentukan, baris dalam tabel pencarian harus cocok dengan semua bidang untuk menghasilkan hasil (DAN logika).
- *output-mode*- Menentukan bagaimana bidang output ditambahkan ke hasil. Gunakan salah satu langkah berikut:
 - OUTPUT— Menambahkan bidang output ke hasil. Jika bidang dengan nama yang sama sudah ada dalam peristiwa log, itu akan ditimpa dengan nilai tabel pencarian. Jika tidak ada kecocokan yang ditemukan, bidang diatur ke null.
 - OUTPUTNEW— Menambahkan bidang output ke hasil hanya jika bidang belum ada dalam peristiwa log. Jika bidang sudah memiliki nilai, nilai asli disimpan. Jika tidak ada kecocokan yang ditemukan, bidang dibiarkan tidak berubah.
- *output-field*— Satu atau lebih bidang dari tabel pencarian untuk ditambahkan ke hasil.

Contoh: Per kaya peristiwa log dengan detail pengguna

Misalkan Anda memiliki grup log dengan peristiwa yang berisi id bidang, dan tabel pencarian bernama `user_data` dengan kolom `id`, `name`, `email`, dan `department`. Kueri berikut memperkaya setiap peristiwa log dengan nama pengguna, email, dan departemen dari tabel pencarian.

```
fields action, status, name, email, department
| lookup user_data id OUTPUT name, email, department
```

Contoh: Gunakan pencarian dengan agregasi

Anda dapat menggunakan bidang output pencarian dengan fungsi agregasi. Kueri berikut memperkaya peristiwa log dengan detail pengguna dan kemudian menghitung peristiwa yang dikelompokkan berdasarkan alamat email.

```
fields user_id, action, username, email, department
| lookup user_data user_id OUTPUT username, email, department
| stats count(*) by email
```

Contoh: Gunakan pencarian dengan filter

Anda dapat memfilter hasil berdasarkan bidang yang dikembalikan oleh pencarian. Kueri berikut memperkaya peristiwa log dan kemudian memfilter untuk hanya menampilkan peristiwa dari departemen tertentu.

```
fields user_id, action
| lookup user_data user_id OUTPUT username, email, department
| filter department = "Engineering"
```

Contoh: Gunakan OUTPUTNEW untuk memperkaya tanpa menimpa

Jika peristiwa log Anda sudah berisi hostname bidang tetapi terkadang kosong, gunakan OUTPUTNEW untuk mengisi nilai yang hilang tanpa menimpa yang sudah ada.

```
fields srcAddr, hostname
| lookup known_hosts ip_address as srcAddr OUTPUTNEW hostname, region
```

Contoh: Gunakan pencarian dengan beberapa bidang kecocokan

Anda dapat mencocokkan lebih dari satu bidang. Kueri berikut cocok srcAddr dengan tabel pencarian untuk mengidentifikasi layanan jaringan yang dikenal. dstPort

```
fields @timestamp, srcAddr, dstAddr, dstPort
| lookup network_services ip_address as srcAddr, port as dstPort OUTPUT service_name,
  owner
| filter ispresent(service_name)
```

join

Menggabungkan peristiwa log dari grup log sumber dengan peristiwa dari grup log lain atau hasil kueri berdasarkan bidang yang cocok.

Gunakan join perintah untuk mengkorelasikan peristiwa log terkait di berbagai sumber seperti grup log menggunakan kunci yang umum di seluruh mereka seperti pencocokan pengidentifikasi permintaan atau transaksi. IDs

Sintaks

```
join type=<join_type> left=<left_alias> right=<right_alias>
  where <left_alias>.<field>=<right_alias>.<field>
  (SOURCE <right_log_group>)
```

Parameter

- `<right_log_group>`— Sumber data sekunder untuk bergabung dengan.
- `<left_alias>` dan `<right_alias>` — Alias untuk membedakan bidang dari sumber data kiri (primer) dan kanan (sekunder).
- `where <field>`- Menentukan bidang yang digunakan sebagai kunci bergabung. Bidang harus ada di kedua sumber data.
- `type=<join_type>(opsional)` - Menentukan jenis bergabung. Nilai yang valid adalah:
 - `inner(default)` - Mengembalikan hanya catatan yang cocok
 - `left`— Mengembalikan semua catatan dari sumber data primer dan catatan yang cocok dari sumber data sekunder

Contoh

Example Contoh 1: Korelasikan permintaan API Gateway dengan log eksekusi Lambda

Contoh ini menunjukkan cara menggabungkan log akses API Gateway dengan log fungsi Lambda untuk menghubungkan permintaan yang masuk dengan pemrosesan backend mereka. Ini berguna untuk memecahkan masalah alur end-to-end permintaan dan mengidentifikasi pemanggilan Lambda mana yang sesuai dengan permintaan API tertentu.

```
filter status >= 500
| join type=inner left=api right=lambda
  where api.requestId=lambda.requestId
  (SOURCE '/aws/lambda/my-function')
| fields api.requestId, api.status, api.latency, lambda.duration, lambda.memoryUsed
| sort api.latency desc
```

Kueri ini:

1. Kueri API Gateway akses log dan filter untuk kesalahan server (`status >= 500`)
2. Bergabung dengan log fungsi Lambda menggunakan bidang `requestId` yang muncul di kedua sumber log
3. Menggunakan alias (`api` dan `lambda`) untuk membedakan bidang dari setiap sumber
4. Mengembalikan informasi gabungan yang menunjukkan latensi API bersama durasi eksekusi Lambda dan penggunaan memori
5. Mengurutkan hasil berdasarkan latensi API untuk mengidentifikasi permintaan paling lambat

Example Contoh 2: Lacak transaksi terdistribusi di seluruh layanan mikro

Saat men-debug masalah dalam arsitektur layanan mikro, Anda sering perlu melacak transaksi di beberapa layanan. Contoh ini menunjukkan cara menggabungkan log dari dua layanan berbeda menggunakan ID transaksi umum.

```
filter eventType = "ORDER_CREATED"
| join type=left left=order right=payment
  where order.transactionId=payment.transactionId
  (SOURCE '/aws/lambda/payment-service')
| filter payment.eventType = "PAYMENT_PROCESSED" or !ispresent(payment.eventType)
| fields order.transactionId, order.orderId, order.customerId,
  payment.paymentStatus, payment.amount
| filter payment.paymentStatus != "SUCCESS" or !ispresent(payment.paymentStatus)
```

Kueri ini:

1. Dimulai dengan acara pembuatan pesanan dari layanan pesanan
2. Menggunakan a `left join` untuk memasukkan semua pesanan, bahkan yang tidak memiliki catatan pembayaran yang cocok
3. Bergabung dengan acara pemrosesan pembayaran menggunakan bidang bersama `transactionId`
4. Memfilter hasil akhir untuk hanya menampilkan pesanan dengan pembayaran gagal atau catatan pembayaran yang hilang

Gabungan kiri penting di sini karena memastikan Anda melihat pesanan yang dibuat tetapi tidak pernah memiliki acara pembayaran yang sesuai, yang dapat menunjukkan kegagalan sistem.

Perilaku

- Sumber data primer (sisi kiri) diproses terlebih dahulu.
- Sumber data sekunder dievaluasi dan dicocokkan menggunakan kunci gabungan yang ditentukan.
- Pencocokan dilakukan dengan menggunakan perbandingan kesetaraan di bidang gabungan.
- Untuk gabungan kiri, catatan tak tertandingi dari sumber data primer dipertahankan dengan nilai nol untuk bidang sekunder.

Catatan dan batasan

- Hanya kondisi kesetaraan (=) yang didukung.
- Hanya satu perintah join yang didukung per kueri.
- Kunci gabungan harus ada di kedua sumber data dan tipe yang kompatibel.
- Kueri yang menggunakan join dapat memindai lebih banyak data dan menimbulkan biaya lebih tinggi.
- Jumlah nilai kunci unik dalam sumber data sekunder dibatasi hingga 50.000 untuk memastikan kinerja kueri.
- Subquery di sisi kanan join tidak didukung.

Perintah-perintah terkait

- [bidang](#)
- [menyaring](#)
- [mengurai](#)
- [statistik](#)
- [mengurutkan](#)
- [batasi](#)

subkueri

Subquery adalah kueri Wawasan Log bersarang yang dapat digunakan sebagai masukan ke kueri lain. Subquery dapat digunakan untuk mendapatkan set hasil antara yang kemudian dikonsumsi oleh perintah berikutnya.

Sintaksis

Subquery dalam filter

```
filter <field> in (  
    <subquery>  
)
```

Parameter

- `<subquery>`— Kueri Wawasan Log valid yang mengembalikan kumpulan hasil. Subquery harus menghasilkan bidang yang direferensikan oleh kueri luar.

Contoh

Example Contoh 1: Temukan permintaan yang mengalami kesalahan dalam layanan hilir

Contoh ini menunjukkan cara menggunakan subquery untuk mengidentifikasi permintaan di layanan utama Anda yang mengakibatkan kesalahan dalam layanan hilir. Ini berguna untuk memecahkan masalah kegagalan cascading dalam sistem terdistribusi.

```
filter requestId in (  
  SOURCE '/aws/lambda/database-service'  
  | filter errorType = "DatabaseConnectionTimeout"  
  | fields requestId  
)  
| fields @timestamp, requestId, endpoint, userId, responseTime  
| sort @timestamp desc
```

Kueri ini:

1. Subquery menemukan semua requestId nilai dari layanan database yang mengalami batas waktu koneksi
2. Kueri luar memfilter log layanan utama Anda untuk hanya menampilkan permintaan yang cocok dengan permintaan rawan kesalahan tersebut IDs
3. Hasil menunjukkan konteks lengkap permintaan yang gagal di hilir, termasuk titik akhir dan pengguna mana yang terpengaruh

Pola ini membantu Anda memahami dampak hulu dari kegagalan hilir.

Example Contoh 2: Identifikasi permintaan yang sering gagal untuk investigasi yang ditargetkan

Contoh ini menunjukkan penggunaan subquery dengan agregasi untuk menemukan permintaan yang gagal berulang kali, yang sering menunjukkan masalah sistematis daripada kesalahan sementara.

```
filter requestId in (  

```

```
SOURCE '/aws/lambda/payment-processor'  
| filter status = "FAILED"  
| stats count(*) as failureCount by requestId  
| filter failureCount > 3  
| fields requestId  
)  
| fields @timestamp, requestId, customerId, amount, failureReason  
| sort @timestamp asc
```

Kueri ini:

1. Subquery menggabungkan upaya pembayaran gagal dan mengidentifikasi permintaan IDs yang gagal lebih dari 3 kali
2. Kueri luar mengambil semua peristiwa log untuk permintaan bermasalah tersebut IDs
3. Hasil diurutkan secara kronologis untuk menunjukkan perkembangan upaya coba lagi

Ini membantu membedakan antara kegagalan sementara (kejadian tunggal) dan masalah persisten (beberapa kegagalan) yang memerlukan penyelidikan lebih dalam.

Perilaku

- Subquery dijalankan secara independen dari query luar.
- Hasil diwujudkan sebelum dikonsumsi oleh kueri luar.
- Hanya bidang yang dipilih secara eksplisit dalam subquery yang tersedia untuk kueri luar.

Catatan dan batasan

- Subkueri harus mengembalikan bidang yang direferensikan oleh kueri luar.
- Subquery bersarang tidak didukung.
- Subquery dapat meningkatkan waktu eksekusi query dan biaya.
- Subquery berkorelasi tidak didukung.
- Eksekusi kueri batin dibatasi hingga 30 detik.

Perintah-perintah terkait

- [bidang](#)
- [menyaring](#)
- [mengurai](#)
- [statistik](#)
- [mengurutkan](#)
- [batasi](#)

Boolean, perbandingan, numerik, datetime, dan fungsi lainnya

CloudWatch Log Insights mendukung banyak operasi dan fungsi lain dalam kueri, seperti yang dijelaskan di bagian berikut.

Topik

- [Operator aritmatika](#)
- [Operator Boolean](#)
- [Operator perbandingan](#)
- [Operator numerik](#)
- [Jenis struktur](#)
- [Fungsi datetime](#)
- [Fungsi umum](#)
- [Fungsi JSON](#)
- [Fungsi string alamat IP](#)
- [Fungsi string](#)

Operator aritmatika

Operator aritmatika menerima tipe data numerik sebagai argumen dan mengembalikan hasil numerik. Gunakan operator aritmatika dalam `filter` dan `fields` perintah dan sebagai argumen untuk fungsi lainnya.

Operasi	Deskripsi
<code>a + b</code>	Penambahan

Operasi	Deskripsi
$a - b$	Pengurangan
$a * b$	Perkalian
a / b	Pembagian
$a ^ b$	Eksponensiasi (pengembalian) $2 ^ 3 8$
$a \% b$	Sisa atau modulus (pengembalian) $10 \% 3 1$

Operator Boolean

Gunakan operator Boolean **and**, **or**, dan **not**.

Note

Gunakan operator Boolean hanya dalam fungsi yang mengembalikan nilai TRUE atau FALSE.

Operator perbandingan

Operator perbandingan menerima semua tipe data sebagai argumen dan mengembalikan hasil Boolean. Gunakan operasi perbandingan dalam `filter` perintah dan sebagai argumen untuk fungsi lainnya.

Operator	Deskripsi
<code>=</code>	Sama
<code>!=</code>	Tidak sama
<code><</code>	Kurang dari
<code>></code>	Lebih besar dari
<code><=</code>	Kurang dari atau sama dengan

Operator	Deskripsi
<code>>=</code>	Lebih besar dari atau sama dengan

Operator numerik

Operasi numerik menerima tipe data numerik sebagai argumen dan mengembalikan hasil numerik. Gunakan operasi numerik dalam `filter` dan `fields` perintah dan sebagai argumen untuk fungsi lainnya.

Operasi	Tipe Hasil	Deskripsi
<code>abs(a: number)</code>	number	Nilai absolut
<code>ceil(a: number)</code>	number	Bulat ke langit-langit (bilangan bulat terkecil yang lebih besar dari nilai) a
<code>floor(a: number)</code>	number	Bulat ke lantai (bilangan bulat terbesar yang lebih kecil dari nilai) a
<code>greatest(a: number, ...numbers: number[])</code>	number	Mengembalikan nilai terbesar
<code>least(a: number, ...numbers: number[])</code>	number	Mengembalikan nilai terkecil
<code>log(a: number)</code>	number	Log alami
<code>sqrt(a: number)</code>	number	Akar kuadrat

Jenis struktur

Peta atau daftar adalah tipe struktur dalam Wawasan CloudWatch Log yang memungkinkan Anda mengakses dan menggunakan atribut untuk kueri.

Contoh: Untuk mendapatkan peta atau daftar

Gunakan `jsonParse` untuk mengurai bidang yang berupa string json ke dalam peta atau daftar.

```
fields jsonParse(@message) as json_message
```

Contoh: Untuk mengakses atribut

Gunakan operator akses titik (`map.attribute`) untuk mengakses item di peta.. Jika atribut dalam peta berisi karakter khusus, gunakan backticks untuk melampirkan nama atribut (`map.attributes.`special.char``).

```
fields jsonParse(@message) as json_message
| stats count() by json_message.status_code
```

Gunakan operator akses braket (`list [index]`) untuk mengambil item pada posisi tertentu dalam daftar.

```
fields jsonParse(@message) as json_message
| filter json_message.users[1].action = "PutData"
```

Bungkus karakter khusus dalam backticks (``) ketika karakter khusus hadir dalam nama kunci.

```
fields jsonParse(@message) as json_message
| filter json_message.`user.id` = "123"
```

Contoh: hasil kosong

Peta dan daftar diperlakukan sebagai null untuk fungsi string, nomor, dan datetime.

```
fields jsonParse(@message) as json_message
| display toupper(json_message)
```

Membandingkan peta dan daftar dengan bidang lain menghasilkan `false`.

Note

Menggunakan peta dan daftar di `deduppattern`, `sort`, dan `stats` tidak didukung.

Fungsi datetime

Fungsi Datetime

Gunakan fungsi `datetime` dalam `fields` dan `filter` perintah dan sebagai argumen untuk fungsi lainnya. Gunakan fungsi ini untuk membuat bucket waktu untuk kueri dengan fungsi agregat.

Gunakan periode waktu yang terdiri dari angka dan salah satu dari yang berikut:

- `ms` untuk milidetik
- `s` selama beberapa detik
- `m` selama beberapa menit
- `h` selama berjam-jam

Misalnya, `10m` adalah 10 menit, dan `1h` 1 jam.

Note

Gunakan unit waktu yang paling tepat untuk fungsi `datetime` Anda. CloudWatch Log membatasi permintaan Anda sesuai dengan satuan waktu yang Anda pilih. Misalnya, ini membatasi 60 sebagai nilai maksimum untuk setiap permintaan yang menggunakannya. Jadi, jika Anda menentukan `bin(300s)`, CloudWatch Log sebenarnya mengimplementasikan ini sebagai 60 detik, karena 60 adalah jumlah detik dalam satu menit sehingga CloudWatch Log tidak akan menggunakan angka yang lebih tinggi dari 60 dengans. Untuk membuat ember 5 menit, gunakan `bin(5m)` sebagai gantinya.

Tutup untuk `ms` adalah 1000, tutup untuk `s` dan `m` 60, dan tutupnya `h` adalah 24.

Tabel berikut berisi daftar fungsi `datetime` yang berbeda yang dapat Anda gunakan dalam perintah query. Tabel mencantumkan jenis hasil setiap fungsi dan berisi deskripsi dari setiap fungsi.

Tip

Saat Anda membuat perintah kueri, Anda dapat menggunakan pemilih interval waktu untuk memilih periode waktu yang ingin Anda kueri. Misalnya, Anda dapat mengatur periode waktu antara interval 5 dan 30 menit; interval 1, 3, dan 12 jam; atau kerangka waktu khusus. Anda juga dapat mengatur periode waktu antara tanggal tertentu.

Fungsi	Tipe hasil	Deskripsi
<code>bin(period: Period)</code>	Stempel waktu	Membulatkan nilai <code>@timestamp</code> ke periode waktu tertentu dan kemudian memotong. Misalnya, <code>bin(5m)</code> bulatkan nilai <code>@timestamp</code> ke 5 menit terdekat. Anda dapat menggunakan ini untuk mengelompokkan beberapa entri log bersama-sama dalam kueri. Contoh berikut mengembalikan jumlah pengecualian per jam: <pre>filter @message like /Exception/ stats count(*) as exceptionCount by bin(1h) sort exceptionCount desc</pre> Satuan waktu dan singkatan berikut didukung dengan <code>bin</code> fungsi tersebut. Untuk semua unit dan singkatan yang menyertakan lebih dari satu karakter, menambahkan <code>s</code> ke pluralisasi didukung. Jadi keduanya <code>hr</code> dan <code>hrs</code> bekerja untuk menentukan jam. • <code>millisecond ms msec</code> • <code>second s sec</code> • <code>minute m min</code> • <code>hour h hr</code> • <code>day d</code> • <code>week w</code> • <code>month mo mon</code> • <code>quarter q qtr</code> • <code>year y yr</code>

Fungsi	Tipe hasil	Deskripsi
<code>datefloor(timestamp: Timestamp, period: Period)</code>	Stempel waktu	Memotong stempel waktu ke periode tertentu. Misalnya, <code>datefloor(@timestamp, 1h)</code> memotong semua nilai <code>@timestamp</code> ke bagian bawah jam.
<code>dateceil(timestamp: Timestamp, period: Period)</code>	Stempel waktu	Membulatkan stempel waktu ke periode tertentu dan kemudian memotong. Misalnya, <code>dateceil(@timestamp, 1h)</code> memotong semua nilai <code>@timestamp</code> ke bagian atas jam.
<code>fromMillis(fieldName: number)</code>	Stempel waktu	Menafsirkan bidang input sebagai jumlah milidetik sejak jangka waktu Unix dan mengubahnya menjadi stempel waktu.
<code>toMillis(fieldName: Timestamp)</code>	nomor	Mengonversi stempel waktu yang ditemukan di bidang bernama menjadi angka yang mewakili milidetik sejak jangka waktu Unix. Misalnya, <code>toMillis(@timestamp)</code> mengubah stempel waktu <code>2022-01-14T13:18:031.000-08:00</code> menjadi <code>1642195111000</code> .

Fungsi	Tipe hasil	Deskripsi
now()	number	Mengembalikan waktu pemrosesan query dimulai, dalam detik epoch. Fungsi ini tidak membutuhkan argumen. Anda dapat menggunakan ini untuk memfilter hasil kueri Anda sesuai dengan waktu saat ini. Misalnya, query berikut mengembalikan semua kesalahan 4xx dari dua jam terakhir: <pre> parse @message "Status Code: *;" as statusCode\n filter statusCode >= 400 and statusCode <= 499 \n filter toMillis(@timestamp) >= (now() * 1000 - 7200000) </pre> Contoh berikut mengembalikan semua entri log dari lima jam terakhir yang berisi kata error atau failure <pre> fields @timestamp, @message filter @message like /(?(i)(error failure))/ filter toMillis(@timestamp) >= (now() * 1000 - 18000000) </pre>

Note

Saat ini, CloudWatch Logs Insights tidak mendukung pemfilteran log dengan stempel waktu yang dapat dibaca manusia.

Fungsi umum

Fungsi umum

Gunakan fungsi umum dalam `fields` dan `filter` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>ispresent(fieldName: LogField)</code>	Boolean	Mengembalikan <code>true</code> jika bidang ada
<code>coalesce(fieldName: LogField, ...fieldNames: LogField[])</code>	LogField	Mengembalikan nilai non-null pertama dari daftar

Fungsi JSON

Fungsi JSON

Gunakan fungsi JSON dalam `fields` dan `filter` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>jsonParse(fieldName: string)</code>	Peta Daftar Kosong	Mengembalikan peta atau daftar ketika input adalah representasi string dari objek JSON atau array JSON. Mengembalikan nilai kosong, jika masukan bukan salah satu representasi.
<code>jsonStringify(fieldName: Map List)</code>	String	Mengembalikan string JSON dari peta atau daftar data.

Fungsi string alamat IP

Fungsi string alamat IP

Gunakan fungsi string alamat IP dalam `filter` dan `fields` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>isValidIp(fieldName: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah valid IPv4 atau IPv6 alamat.
<code>isValidIPv4(fieldName: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah IPv4 alamat yang valid.
<code>isValidIPv6(fieldName: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah IPv6 alamat yang valid.
<code>isIpInSubnet(fieldName: string, subnet: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah valid IPv4 atau IPv6 alamat dalam v4 atau v6 subnet yang ditentukan. Saat Anda menentukan subnet, gunakan notasi CIDR seperti <code>192.0.2.0/24</code> atau <code>2001:db8::/32</code> , di mana <code>192.0.2.0</code> atau <code>2001:db8::</code> merupakan awal dari blok CIDR.
<code>isIPv4InSubnet(fieldName: string, subnet: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah IPv4 alamat yang valid dalam subnet v4 yang ditentukan. Saat Anda menentukan subnet, gunakan notasi CIDR seperti <code>192.0.2.0/24</code> di mana <code>192.0.2.0</code> adalah awal blok CIDR..
<code>isIPv6InSubnet(fieldName: string, subnet: string)</code>	boolean	Mengembalikan <code>true</code> jika bidang adalah IPv6 alamat yang valid dalam subnet v6 yang ditentukan. Saat Anda menentukan subnet, gunakan notasi CIDR seperti <code>2001:db8::/32</code> di mana <code>2001:db8::</code> adalah awal blok CIDR.

Fungsi string

Fungsi string

Gunakan fungsi string dalam `fields` dan `filter` perintah dan sebagai argumen untuk fungsi lainnya.

Fungsi	Tipe hasil	Deskripsi
<code>isEmpty(fieldName: string)</code>	Bilangan	Mengembalikan 1 jika bidang tidak ada atau string kosong.
<code>isBlank(fieldName: string)</code>	Bilangan	Mengembalikan 1 jika bidang tidak ada, string kosong, atau hanya berisi spasi.
<code>concat(str: string, ...strings: string[])</code>	string	Menyatukan string.
<code>ltrim(str: string)</code> <code>ltrim(str: string, trimChars: string)</code>	string	Jika fungsi tidak memiliki argumen kedua, ia menghapus spasi putih dari kiri string. Jika fungsi memiliki argumen string kedua, itu tidak menghapus spasi putih. Sebaliknya, ia menghapus karakter <code>trimChars</code> dari kiristr. Misalnya, <code>ltrim("xyZxyfooxyZ", "xyZ")</code> mengembalikan "fooxyZ".
<code>rtrim(str: string)</code> <code>rtrim(str: string, trimChars: string)</code>	string	Jika fungsi tidak memiliki argumen kedua, ia menghapus spasi putih dari kanan string. Jika fungsi memiliki argumen string kedua, itu tidak menghapus spasi putih. Sebaliknya, ia menghapus karakter <code>trimChars</code> dari dari kananstr. Misalnya, <code>rtrim("xyZfooxyxyZ", "xyZ")</code> mengembalikan "xyZfoo".

Fungsi	Tipe hasil	Deskripsi
<code>trim(str: string)</code> <code>trim(str: string, trimChars: string)</code>	string	Jika fungsi tidak memiliki argumen kedua, ia menghapus spasi putih dari kedua ujung string. Jika fungsi memiliki argumen string kedua, itu tidak menghapus spasi putih. Sebaliknya, ia menghapus karakter <code>trimChars</code> dari kedua sisi <code>str</code> . Misalnya, <code>trim("xyZxyfooxyxyZ", "xyZ")</code> mengembalikan "foo".
<code>strlen(str: string)</code>	nomor	Mengembalikan panjang string dalam poin kode Unicode.
<code>toupper(str: string)</code>	string	Mengonversi string menjadi huruf besar.
<code>tolower(str: string)</code>	string	Mengonversi string menjadi huruf kecil.
<code>substr(str: string, startIndex: number)</code> <code>substr(str: string, startIndex: number, length: number)</code>	string	Mengembalikan substring dari indeks yang ditentukan oleh argumen angka ke akhir string. Jika fungsi memiliki argumen angka kedua, itu berisi panjang substring yang akan diambil. Misalnya, <code>substr("xyZfooxyZ", 3, 3)</code> mengembalikan "foo".

Fungsi	Tipe hasil	Deskripsi
<code>replace(fieldName: string, searchValue: string, replaceValue: string)</code>	string	Mengganti semua <code>searchValue</code> dalam <code>fieldName: string</code> dengan <code>replaceValue</code> . Misalnya, fungsi <code>replace(logGroup, "smoke_test", "Smoke")</code> mencari peristiwa log di mana bidang <code>logGroup</code> berisi nilai string <code>smoke_test</code> dan mengganti nilai dengan string. <code>Smoke</code>
<code>strcontains(str: string, searchValue: string)</code>	number	Mengembalikan 1 jika <code>str</code> berisi <code>searchValue</code> dan 0 sebaliknya.

Bidang yang berisi karakter khusus

Jika bidang berisi karakter non-alfanumerik selain @ simbol atau periode (.), Anda harus mengelilingi bidang dengan karakter backtick (`). Misalnya, bidang log `foo-bar` harus diapit backticks (``foo-bar``) karena berisi karakter non-alfanumerik, tanda hubung (-).

Gunakan alias dan komentar dalam kueri

Buat kueri yang berisi alias. Gunakan alias untuk mengganti nama bidang log atau saat mengekstrak nilai ke dalam bidang. Gunakan kata kunci `as` untuk memberikan bidang log atau menghasilkan alias. Anda dapat menggunakan lebih dari satu alias dalam kueri. Anda dapat menggunakan alias dalam perintah berikut:

- `fields`
- `parse`
- `sort`
- `stats`

Contoh berikut menunjukkan cara membuat kueri yang berisi alias.

Contoh

Query berisi alias dalam `fields` perintah.

```
fields @timestamp, @message, accountId as ID
| sort @timestamp desc
| limit 20
```

Query mengembalikan nilai-nilai untuk bidang `@timestamp`, `@message`, dan `accountId`. Hasilnya diurutkan dalam urutan menurun dan dibatasi hingga 20. Nilai untuk `accountId` tercantum di bawah alias `ID`.

Contoh

Kueri berisi alias dalam `stats` perintah `sort` dan.

```
stats count(*) by duration as time
| sort time desc
```

Kueri menghitung berapa kali bidang `duration` terjadi di grup log dan mengurutkan hasil dalam urutan menurun. Nilai untuk `duration` tercantum di bawah alias `time`.

Gunakan komentar

CloudWatch Log Insights mendukung komentar dalam kueri. Gunakan karakter hash (`#`) untuk memicu komentar. Anda dapat menggunakan komentar untuk mengabaikan baris dalam kueri atau kueri dokumen.

Contoh: Query

Ketika query berikut dijalankan, baris kedua diabaikan.

```
fields @timestamp, @message, accountId
# | filter accountId not like "7983124201998"
| sort @timestamp desc
| limit 20
```

Memulai dengan Logs Insights QL: Tutorial kueri

Bagian berikut mencakup contoh tutorial kueri untuk membantu Anda memulai dengan Wawasan Log QL.

Topik

- [Tutorial: Jalankan dan modifikasi kueri sampel](#)
- [Tutorial: Jalankan kueri dengan fungsi agregasi](#)
- [Tutorial: Jalankan kueri yang menghasilkan visualisasi yang dikelompokkan berdasarkan bidang log](#)
- [Tutorial: Jalankan kueri yang menghasilkan visualisasi deret waktu](#)

Tutorial: Jalankan dan modifikasi kueri sampel

Tutorial berikut membantu Anda memulai dengan Wawasan CloudWatch Log. Anda menjalankan kueri sampel di Logs Insights QL, lalu melihat cara memodifikasi dan menjalankannya kembali.

Untuk menjalankan kueri, Anda harus sudah memiliki log yang disimpan di CloudWatch Log. Jika Anda sudah menggunakan CloudWatch Log dan memiliki grup log dan aliran log yang disiapkan, Anda siap untuk memulai. Anda mungkin juga sudah memiliki log jika Anda menggunakan layanan seperti AWS CloudTrail, Amazon Route 53, atau Amazon VPC dan Anda telah menyiapkan log dari layanan tersebut untuk masuk ke CloudWatch Log. Untuk informasi selengkapnya tentang mengirim CloudWatch log ke Log, lihat [Memulai dengan CloudWatch Log](#).

Kueri dalam Wawasan CloudWatch Log mengembalikan sekumpulan bidang dari peristiwa log atau hasil agregasi matematis atau operasi lain yang dilakukan pada peristiwa log. Tutorial ini menunjukkan kueri yang mengembalikan daftar log acara.

Jalankan kueri sampel

Untuk menjalankan kueri sampel Wawasan CloudWatch Log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.

Pada halaman Wawasan Log, editor kueri berisi kueri default di Logs Insights QL yang menampilkan 20 peristiwa log terbaru.

3. Dalam menu tarik-turun Pilih grup log, pilih satu atau beberapa grup log untuk kueri.

Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih grup log di akun sumber serta akun pemantauan. Satu kueri dapat menanyakan log dari akun yang berbeda sekaligus.

Anda dapat memfilter grup log berdasarkan nama grup log, ID akun, atau label akun.

Saat Anda memilih grup log di kelas log Standar, Wawasan CloudWatch Log secara otomatis mendeteksi bidang data dalam grup. Untuk melihat bidang yang ditemukan, pilih menu Fields di dekat kanan atas halaman.

 Note

Bidang yang ditemukan hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

4. (Opsional) Gunakan pemilih interval waktu untuk memilih periode waktu yang ingin Anda kueri.

Anda dapat memilih antara interval 5 dan 30 menit; interval 1, 3, dan 12 jam; atau kerangka waktu khusus.

5. Pilih Jalankan untuk melihat hasilnya.

Untuk tutorial ini, hasilnya mencakup 20 peristiwa log yang paling baru ditambahkan.

CloudWatch Log menampilkan grafik batang peristiwa log dalam grup log dari waktu ke waktu. Grafik batang tidak hanya menunjukkan peristiwa dalam tabel, tetapi juga distribusi peristiwa dalam grup log yang cocok dengan kueri dan jangka waktu.

6. Untuk melihat semua bidang untuk peristiwa log yang dikembalikan, pilih ikon tarik-turun segitiga di sebelah kiri acara bernomor.

Ubah kueri sampel

Dalam tutorial ini, Anda mengubah kueri sampel untuk menunjukkan 50 log acara terbaru.

Jika Anda belum menjalankan tutorial sebelumnya, lakukan sekarang. Tutorial ini dimulai di tempat tutorial sebelumnya berakhir.

 Note

Beberapa contoh kueri yang disediakan dengan penggunaan CloudWatch Log Insights head atau tail perintah sebagai gantinya. `limit` Perintah ini akan tidak digunakan lagi dan telah diganti dengan `limit`. Gunakan `limit`, dan bukan head atau tail dalam semua kueri yang Anda tulis.

Untuk memodifikasi CloudWatch kueri sampel Wawasan Log

1. Di editor kueri, ubah 20 menjadi 50, lalu pilih Run (Jalankan).

Hasil kueri baru akan muncul. Dengan asumsi ada cukup data dalam grup log selama rentang waktu default, sekarang ada 50 log acara yang tercantum.

2. (Opsional) Anda dapat menyimpan kueri yang telah Anda buat. Untuk menyimpan kueri ini, pilih Save (Simpan). Untuk informasi selengkapnya, lihat [Simpan dan jalankan kembali kueri CloudWatch Logs Insights](#).

Tambahkan perintah filter ke kueri sampel

Tutorial ini menunjukkan cara membuat perubahan yang lebih kuat pada kueri di editor kueri. Dalam tutorial ini, Anda memfilter hasil kueri sebelumnya berdasarkan bidang dalam log acara yang diambil.

Jika Anda belum menjalankan tutorial sebelumnya, lakukan sekarang. Tutorial ini dimulai di tempat tutorial sebelumnya berakhir.

Untuk menambahkan perintah filter ke kueri sebelumnya

1. Tentukan bidang untuk memfilter. Untuk melihat bidang paling umum yang CloudWatch terdeteksi Log dalam peristiwa log yang terdapat dalam grup log yang dipilih dalam 15 menit terakhir, dan persentase peristiwa log di mana setiap bidang muncul, pilih Bidang di sisi kanan halaman.

Untuk melihat bidang yang terdapat dalam log acara tertentu, pilih ikon di sebelah kiri baris tersebut.

Bidang `awsRegion` mungkin muncul dalam log acara Anda, tergantung pada kejadian yang ada di log Anda. Untuk bagian selanjutnya dalam tutorial ini, kita menggunakan `awsRegion` sebagai

bidang filter, tetapi Anda dapat menggunakan bidang yang berbeda jika bidang tersebut tidak tersedia.

2. Di kotak editor kueri, tempatkan kursor Anda setelah 50, lalu tekan Enter.
3. Di baris baru, pertama masukkan | (karakter pipa) dan spasi. Perintah dalam kueri Wawasan CloudWatch Log harus dipisahkan oleh karakter pipa.
4. Masukkan **filter** `awsRegion="us-east-1"`.
5. Pilih Jalankan.

Kueri berjalan lagi, dan sekarang menampilkan 50 hasil terbaru yang cocok dengan filter baru.

Jika Anda memfilter dengan bidang yang berbeda dan mendapat hasil kesalahan, Anda mungkin perlu melakukan escape pada nama bidang. Jika nama bidang mengandung karakter non-alfanumerik, Anda harus menempatkan karakter backtick (`) sebelum dan sesudah nama bidang (misalnya, ``error-code`="102"`).

Anda harus menggunakan karakter backtick untuk nama bidang yang berisi karakter non-alfanumerik, tetapi tidak untuk nilai. Nilai selalu ada dalam tanda kutip (").

Log Insights QL mencakup kemampuan kueri yang kuat, termasuk beberapa perintah dan dukungan untuk ekspresi reguler, matematika, dan operasi statistik. Untuk informasi selengkapnya, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

Tutorial: Jalankan kueri dengan fungsi agregasi

Anda dapat menggunakan fungsi agregasi dengan `stats` perintah dan sebagai argumen untuk fungsi lainnya. Dalam tutorial ini, Anda menjalankan perintah query yang menghitung jumlah peristiwa log yang berisi bidang tertentu. Perintah query mengembalikan jumlah total yang dikelompokkan berdasarkan nilai bidang tertentu atau nilai. Untuk informasi selengkapnya tentang fungsi agregasi, lihat [Operasi dan fungsi yang didukung](#) di Panduan Pengguna CloudWatch Log Amazon.

Untuk menjalankan kueri dengan fungsi agregasi

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Konfirmasikan bahwa tab QL Wawasan Log dipilih.
4. Dalam menu tarik-turun Pilih grup log, pilih satu atau beberapa grup log untuk kueri.

Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih grup log di akun sumber serta akun pemantauan. Satu kueri dapat menanyakan log dari akun yang berbeda sekaligus.

Anda dapat memfilter grup log berdasarkan nama grup log, ID akun, atau label akun.

Saat Anda memilih grup CloudWatch log, Wawasan Log secara otomatis mendeteksi bidang data dalam grup log jika itu adalah grup log kelas Standar. Untuk melihat bidang yang ditemukan, pilih menu Fields di dekat kanan atas halaman.

5. Hapus kueri default di editor kueri, dan masukkan perintah berikut:

```
stats count(*) by fieldName
```

6. Ganti *fieldName* dengan bidang yang ditemukan dari menu Fields.

Menu Fields terletak di kanan atas halaman dan menampilkan semua bidang yang ditemukan yang mendeteksi Wawasan CloudWatch Log di grup log Anda.

7. Pilih Jalankan untuk melihat hasil kueri.

Hasil kueri menunjukkan jumlah catatan dalam grup log Anda yang cocok dengan perintah kueri dan jumlah total yang dikelompokkan berdasarkan nilai atau nilai bidang yang ditentukan.

Tutorial: Jalankan kueri yang menghasilkan visualisasi yang dikelompokkan berdasarkan bidang log

Ketika menjalankan kueri yang menggunakan fungsi `stats` untuk mengelompokkan hasil yang dikembalikan oleh nilai dari satu atau beberapa bidang dalam entri log, Anda dapat melihat hasilnya sebagai diagram batang, diagram lingkaran, grafik garis, atau grafik area bertumpuk. Hal ini membantu memvisualisasikan tren dalam log Anda dengan lebih efisien.

Untuk menjalankan kueri untuk visualisasi

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Dalam menu tarik-turun Pilih grup log, pilih satu atau beberapa grup log untuk kueri.

Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih grup log di akun sumber serta akun pemantauan. Satu kueri dapat menanyakan log dari akun yang berbeda sekaligus.

Anda dapat memfilter grup log berdasarkan nama grup log, ID akun, atau label akun.

4. Di editor kueri, hapus konten saat ini, masukkan fungsi `stats` berikut ini, lalu pilih Run query (Jalankan kueri).

```
stats count(*) by @logStream
| limit 100
```

Hasilnya menunjukkan jumlah log acara dalam grup log untuk setiap pengaliran log. Hasilnya terbatas hanya 100 baris.

5. Pilih tab Visualization (Visualisasi).
6. Pilih panah di sebelah Line (Garis), lalu pilih Bar (Batang).

Akan muncul diagram batang yang menampilkan balok untuk setiap pengaliran log.

Tutorial: Jalankan kueri yang menghasilkan visualisasi deret waktu

Ketika menjalankan kueri yang menggunakan fungsi `bin()` untuk mengelompokkan hasil yang dikembalikan menurut jangka waktu, Anda dapat melihat hasilnya sebagai grafik garis, grafik area bertumpuk, diagram lingkaran, atau diagram batang. Hal ini membantu memvisualisasikan tren dalam log acara dengan lebih efisien dari waktu ke waktu.

Untuk menjalankan kueri untuk visualisasi

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Konfirmasikan bahwa tab QL Wawasan Log dipilih.
4. Dalam menu tarik-turun Pilih grup log, pilih satu atau beberapa grup log untuk kueri.

Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih grup log di akun sumber serta akun pemantauan. Satu kueri dapat menanyakan log dari akun yang berbeda sekaligus.

Anda dapat memfilter grup log berdasarkan nama grup log, ID akun, atau label akun.

5. Di editor kueri, hapus konten saat ini, masukkan fungsi `stats` berikut ini, lalu pilih Run query (Jalankan kueri).

```
stats count(*) by bin(30s)
```

Hasilnya menunjukkan jumlah peristiwa log dalam grup log yang diterima oleh CloudWatch Log untuk setiap periode 30 detik.

6. Pilih tab Visualization (Visualisasi).

Hasilnya ditampilkan sebagai grafik garis. Untuk beralih ke diagram batang, diagram lingkaran, atau diagram area bertumpuk, pilih panah di samping Line (Garis) di kiri atas grafik.

Kueri Sampel

Bagian ini berisi daftar perintah kueri umum dan berguna yang dapat Anda jalankan di [CloudWatchkonsol](#). Untuk informasi tentang cara menjalankan perintah kueri, lihat [Tutorial: Menjalankan dan memodifikasi contoh kueri](#) di Panduan Pengguna Amazon CloudWatch Logs.

Untuk informasi selengkapnya tentang sintaks kueri, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

Topik

- [Kueri umum](#)
- [Kueri untuk log Lambda](#)
- [Kueri untuk log aliran VPC Amazon](#)
- [Kueri untuk log Route 53](#)
- [Kueri untuk log CloudTrail](#)
- [Pertanyaan untuk Amazon API Gateway](#)
- [Pertanyaan untuk gateway NAT](#)
- [Kueri untuk log server Apache](#)
- [Kueri untuk Amazon EventBridge](#)
- [Contoh perintah parse](#)

Kueri umum

Temukan 25 peristiwa log yang paling baru ditambahkan.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

Dapatkan daftar jumlah pengecualian per jam.

```
filter @message like /Exception/  
  | stats count(*) as exceptionCount by bin(1h)  
  | sort exceptionCount desc
```

Dapatkan daftar peristiwa log yang bukan pengecualian.

```
fields @message | filter @message not like /Exception/
```

Dapatkan peristiwa log terbaru untuk setiap nilai unik **server** bidang.

```
fields @timestamp, server, severity, message  
  | sort @timestamp asc  
  | dedup server
```

Dapatkan peristiwa log terbaru untuk setiap nilai unik **server** bidang untuk setiap **severity** jenis.

```
fields @timestamp, server, severity, message  
  | sort @timestamp desc  
  | dedup server, severity
```

Kueri untuk log Lambda

Tentukan jumlah memori yang dilebih-lebihkan.

```
filter @type = "REPORT"  
  | stats max(@memorySize / 1000 / 1000) as provisionedMemoryMB,  
    min(@maxMemoryUsed / 1000 / 1000) as smallestMemoryRequestMB,  
    avg(@maxMemoryUsed / 1000 / 1000) as avgMemoryUsedMB,  
    max(@maxMemoryUsed / 1000 / 1000) as maxMemoryUsedMB,  
    provisionedMemoryMB - maxMemoryUsedMB as overProvisionedMB
```

Buat laporan latensi.

```
filter @type = "REPORT" |
```

```
stats avg(@duration), max(@duration), min(@duration) by bin(5m)
```

Cari pemanggilan fungsi lambat, dan hilangkan permintaan duplikat yang dapat muncul dari percobaan ulang atau kode sisi klien. Dalam query ini, **@duration** adalah dalam milidetik.

```
fields @timestamp, @requestId, @message, @logStream
| filter @type = "REPORT" and @duration > 1000
| sort @timestamp desc
| dedup @requestId
| limit 20
```

Kueri untuk log aliran VPC Amazon

Temukan 15 transfer paket teratas di seluruh host:

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
| sort packetsTransferred desc
| limit 15
```

Temukan transfer 15 byte teratas untuk host pada subnet tertentu.

```
filter isIpv4InSubnet(srcAddr, "192.0.2.0/24")
| stats sum(bytes) as bytesTransferred by dstAddr
| sort bytesTransferred desc
| limit 15
```

Temukan alamat IP yang menggunakan UDP sebagai protokol transfer data.

```
filter protocol=17 | stats count(*) by srcAddr
```

Temukan alamat IP tempat catatan aliran dilewati selama jendela pengambilan.

```
filter logStatus="SKIPDATA"
| stats count(*) by bin(1h) as t
| sort t
```

Temukan satu catatan untuk setiap koneksi, untuk membantu memecahkan masalah konektivitas jaringan.

```
fields @timestamp, srcAddr, dstAddr, srcPort, dstPort, protocol, bytes
| filter logStream = 'vpc-flow-logs' and interfaceId = 'eni-0123456789abcdef0'
| sort @timestamp desc
| dedup srcAddr, dstAddr, srcPort, dstPort, protocol
| limit 20
```

Kueri untuk log Route 53

Temukan distribusi catatan per jam berdasarkan jenis kueri.

```
stats count(*) by queryType, bin(1h)
```

Temukan 10 DNS resolver dengan jumlah permintaan tertinggi.

```
stats count(*) as numRequests by resolverIp
| sort numRequests desc
| limit 10
```

Temukan jumlah catatan berdasarkan domain dan subdomain di mana server gagal menyelesaikan permintaan DNS.

```
filter responseCode="SERVFAIL" | stats count(*) by queryName
```

Kueri untuk log CloudTrail

Temukan jumlah entri log untuk setiap layanan, jenis acara, dan AWS Wilayah.

```
stats count(*) by eventSource, eventName, awsRegion
```

Temukan host Amazon EC2 yang dimulai atau dihentikan di Wilayah tertentu AWS .

```
filter (eventName="StartInstances" or eventName="StopInstances") and awsRegion="us-east-2"
```

Temukan AWS Wilayah, nama pengguna, dan ARNs pengguna IAM yang baru dibuat.

```
filter eventName="CreateUser"
| fields awsRegion, requestParameters.userName, responseElements.user.arn
```

Temukan jumlah catatan di mana pengecualian terjadi saat menjalankan API **UpdateTrail**.

```
filter eventName="UpdateTrail" and ispresent(errorCode)
| stats count(*) by errorCode, errorMessage
```

Temukan entri log di mana TLS 1.0 atau 1.1 digunakan

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
| stats count(*) as numOutdatedTlsCalls by userIdentity.accountId, recipientAccountId,
eventSource, eventName, awsRegion, tlsDetails.tlsVersion, tlsDetails.cipherSuite,
userAgent
| sort eventSource, eventName, awsRegion, tlsDetails.tlsVersion
```

Temukan jumlah panggilan per layanan yang menggunakan TLS versi 1.0 atau 1.1

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
| stats count(*) as numOutdatedTlsCalls by eventSource
| sort numOutdatedTlsCalls desc
```

Pertanyaan untuk Amazon API Gateway

Temukan 10 kesalahan 4XX terakhir

```
fields @timestamp, status, ip, path, httpMethod
| filter status>=400 and status<=499
| sort @timestamp desc
| limit 10
```

Identifikasi 10 Amazon API Gateway permintaan yang paling lama berjalan di grup log akses Amazon API Gateway

```
fields @timestamp, status, ip, path, httpMethod, responseLatency
| sort responseLatency desc
| limit 10
```

Kembalikan daftar jalur API paling populer di grup log Amazon API Gateway akses Anda

```
stats count(*) as requestCount by path
```

```
| sort requestCount desc
| limit 10
```

Membuat laporan latensi integrasi untuk grup log Amazon API Gateway akses Anda

```
filter status=200
| stats avg(integrationLatency), max(integrationLatency),
min(integrationLatency) by bin(1m)
```

Pertanyaan untuk gateway NAT

Jika Anda melihat biaya yang lebih tinggi dari biasanya dalam AWS tagihan Anda, Anda dapat menggunakan Wawasan CloudWatch Log untuk menemukan kontributor teratas. Untuk informasi selengkapnya tentang perintah kueri berikut, [lihat Bagaimana cara menemukan kontributor teratas untuk lalu lintas melalui gateway NAT di VPC saya?](#) di halaman dukungan AWS premium.

 Note

Dalam perintah kueri berikut, ganti “x.x.x.x” dengan IP pribadi gateway NAT Anda, dan ganti “y.y” dengan dua oktet pertama dari rentang CIDR VPC Anda.

Temukan contoh yang mengirimkan lalu lintas terbanyak melalui gateway NAT Anda.

```
filter (dstAddr like 'x.x.x.x' and srcAddr like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

Tentukan lalu lintas yang menuju dan dari instance di gateway NAT Anda.

```
filter (dstAddr like 'x.x.x.x' and srcAddr like 'y.y.') or (srcAddr like 'xxx.xx.xx.xx'
and dstAddr like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

Tentukan tujuan internet yang paling sering berkomunikasi dengan instance di VPC Anda untuk upload dan download.

Untuk upload

```
filter (srcAddr like 'x.x.x.x' and dstAddr not like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

Untuk unduhan

```
filter (dstAddr like 'x.x.x.x' and srcAddr not like 'y.y.')
| stats sum(bytes) as bytesTransferred by srcAddr, dstAddr
| sort bytesTransferred desc
| limit 10
```

Kueri untuk log server Apache

Anda dapat menggunakan Wawasan CloudWatch Log untuk menanyakan log server Apache. Untuk informasi selengkapnya tentang kueri berikut, lihat [Menyederhanakan log server Apache dengan Wawasan CloudWatch Log](#) di Blog Operasi & Migrasi AWS Cloud.

Temukan bidang yang paling relevan, sehingga Anda dapat meninjau log akses Anda dan memeriksa lalu lintas di jalur /admin aplikasi Anda.

```
fields @timestamp, remoteIP, request, status, filename| sort @timestamp desc
| filter filename="/var/www/html/admin"
| limit 20
```

Temukan nomor permintaan GET unik yang mengakses halaman utama Anda dengan kode status "200" (sukses).

```
fields @timestamp, remoteIP, method, status
| filter status="200" and referrer= http://34.250.27.141/ and method= "GET"
| stats count_distinct(remoteIP) as UniqueVisits
| limit 10
```

Temukan berapa kali layanan Apache Anda dimulai ulang.

```
fields @timestamp, function, process, message
| filter message like "resuming normal operations"
| sort @timestamp desc
| limit 20
```


Kueri untuk Amazon EventBridge

Dapatkan jumlah EventBridge acara yang dikelompokkan berdasarkan jenis detail acara

```
fields @timestamp, @message
| stats count(*) as numberOfEvents by `detail-type`
| sort numberOfEvents desc
```

Contoh perintah parse

Gunakan ekspresi glob untuk mengekstrak bidang **@user**, **@method**, dan **@latency** dari bidang log **@message** dan kembalikan latensi rata-rata untuk setiap kombinasi unik dan. **@method @user**

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

Gunakan ekspresi reguler untuk mengekstrak bidang **@user2**, **@method2**, dan **@latency2** dari bidang log **@message** dan kembalikan latensi rata-rata untuk setiap kombinasi unik **@method2** dan **@user2**.

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

Mengekstrak bidang **loggingTime**, **loggingType** dan **loggingMessage**, memfilter ke log peristiwa yang berisi **ERROR** atau **INFO** string, dan kemudian hanya menampilkan **loggingMessage** dan **loggingType** bidang untuk peristiwa yang berisi **ERROR** string.

```
FIELDS @message
| PARSE @message "*" [*] "*" as loggingTime, loggingType, loggingMessage
| FILTER loggingType IN ["ERROR", "INFO"]
| DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

Bandingkan (diff) dengan rentang waktu sebelumnya

Anda dapat menggunakan Wawasan CloudWatch Log dengan QL Wawasan Log untuk membandingkan perubahan dalam peristiwa log Anda dari waktu ke waktu. Anda dapat membandingkan peristiwa log yang dicerna selama rentang waktu terakhir dengan log dari periode waktu sebelumnya. Atau, Anda dapat membandingkan dengan periode waktu sebelumnya yang

serupa. Ini dapat membantu Anda menemukan apakah kesalahan dalam log Anda baru-baru ini diperkenalkan atau sudah terjadi, dan dapat membantu Anda menemukan tren lainnya.

Kueri perbandingan hanya mengembalikan pola dalam hasil, bukan peristiwa log mentah. Pola yang dikembalikan akan membantu Anda dengan cepat melihat tren dan perubahan dalam peristiwa log dari waktu ke waktu. Setelah Anda menjalankan kueri perbandingan dan mendapatkan hasil pola, Anda dapat melihat contoh peristiwa log mentah untuk pola yang Anda minati. Untuk informasi selengkapnya tentang pola log, lihat [Analisis pola](#).

Saat Anda menjalankan kueri perbandingan, kueri Anda dianalisis terhadap dua periode waktu yang berbeda: periode kueri asli yang Anda pilih, dan periode perbandingan. Periode perbandingan selalu sama panjangnya dengan periode kueri asli Anda. Interval waktu default untuk perbandingan adalah sebagai berikut.

- Periode sebelumnya — Membandingkan dengan periode waktu sebelum periode waktu kueri Anda.
- Hari sebelumnya — Membandingkan dengan periode waktu satu hari sebelum periode waktu kueri Anda.
- Minggu sebelumnya — Bandingkan dengan periode waktu satu minggu sebelum periode waktu kueri Anda.
- Bulan sebelumnya — Membandingkan dengan periode waktu satu bulan sebelum periode waktu kueri Anda.

 Note

Kueri yang menggunakan perbandingan dikenakan biaya yang mirip dengan menjalankan satu kueri Wawasan CloudWatch Log selama rentang waktu gabungan. Untuk informasi lebih lanjut, lihat [Amazon CloudWatch Harga](#).

Untuk menjalankan kueri perbandingan

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Wawasan Log.

Kueri default muncul di kotak kueri.

3. Konfirmasikan bahwa tab QL Wawasan Log dipilih.

4. Simpan kueri default atau masukkan kueri yang berbeda.
5. Di menu drop-down Pilih grup log, pilih satu atau beberapa grup log untuk kueri.
6. (Opsional) Gunakan pemilih interval waktu untuk memilih periode waktu yang ingin Anda kueri. Kueri default adalah untuk jam data log sebelumnya.
7. Dengan pemilih rentang waktu, pilih Bandingkan. Kemudian pilih periode waktu sebelumnya yang ingin Anda bandingkan dengan log asli, dan pilih Terapkan.
8. Pilih Run query (Jalankan kueri).

Untuk menyebabkan kueri mengambil data dari periode perbandingan, `diff` perintah ditambahkan ke kueri Anda.

9. Pilih tab Patterns untuk melihat hasilnya.

Tabel menampilkan informasi berikut:

- Setiap Pola, dengan bagian variabel dari pola digantikan oleh simbol token dinamis `<string-number>`. `string` Ini adalah deskripsi dari jenis data yang diwakili oleh token. `number` Menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya. Untuk informasi selengkapnya, lihat [Analisis pola](#).
 - Jumlah peristiwa adalah jumlah peristiwa log dengan pola itu dalam periode waktu asli yang lebih terkini.
 - Perbedaan jumlah peristiwa adalah perbedaan antara jumlah peristiwa log yang cocok dalam periode waktu saat ini versus periode waktu perbandingan. Berbeda positif berarti ada lebih banyak peristiwa seperti itu dalam periode waktu saat ini.
 - Deskripsi perbedaan secara singkat merangkum perubahan pola itu antara periode waktu saat ini dan periode perbandingan.
 - Jenis keparahan adalah kemungkinan tingkat keparahan peristiwa log dengan pola ini, berdasarkan kata-kata yang ditemukan dalam peristiwa log seperti FATAL, ERROR, dan. WARN
10. Untuk memeriksa lebih lanjut salah satu pola dalam daftar, pilih ikon di kolom Inspect untuk salah satu pola.

Panel pemeriksaan Pola muncul dan menampilkan yang berikut:

- Pola. Pilih token dalam pola untuk menganalisis nilai token tersebut.
- Histogram yang menunjukkan jumlah kemunculan pola selama rentang waktu yang ditanyakan. Ini dapat membantu Anda mengidentifikasi tren menarik seperti peningkatan tiba-tiba dalam terjadinya suatu pola.

- Tab Log samples menampilkan beberapa peristiwa log yang cocok dengan pola yang dipilih.
- Tab Nilai Token menampilkan nilai token dinamis yang dipilih, jika Anda telah memilihnya.

 Note

Maksimal 10 nilai token ditangkap untuk setiap token. Jumlah token mungkin tidak tepat. CloudWatch Log menggunakan penghitung probabilistik untuk menghasilkan jumlah token, bukan nilai absolut.

- Tab pola Terkait menampilkan pola lain yang sering terjadi mendekati waktu yang sama dengan pola yang Anda periksa. Misalnya, jika pola untuk ERROR pesan biasanya disertai dengan peristiwa log lain yang ditandai INFO dengan detail tambahan, pola itu ditampilkan di sini.

Visualisasikan data log dalam grafik

Anda dapat menggunakan visualisasi seperti diagram batang, diagram garis, dan bagan area bertumpuk untuk mengidentifikasi pola dalam data log Anda dengan lebih efisien. CloudWatch Log Insights menghasilkan visualisasi untuk kueri yang menggunakan `stats` fungsi dan satu atau beberapa fungsi agregasi. Untuk informasi lebih lanjut, lihat [statistik](#).

OpenSearch Bahasa Pemrosesan Pipa (PPL)

Bagian ini berisi pengantar dasar untuk menanyakan CloudWatch Log menggunakan OpenSearch PPL. Dengan PPL, Anda dapat mengambil, menanyakan, dan menganalisis data menggunakan perintah piped-together, sehingga lebih mudah untuk memahami dan menyusun kueri yang kompleks. Sintaksnya didasarkan pada pipa Unix, dan memungkinkan rantai perintah untuk mengubah dan memproses data. Dengan PPL, Anda dapat memfilter dan mengumpulkan data, dan menggunakan serangkaian matematika, string, tanggal, kondisional, dan fungsi lainnya yang kaya untuk analisis.

Termasuk `SOURCE` dalam kueri PPL adalah cara yang berguna untuk menentukan indeks bidang grup log, dan sumber data untuk disertakan dalam kueri saat Anda menggunakan AWS CLI atau API untuk membuat kueri. `SOURCE` Perintah hanya didukung di API AWS CLI dan, bukan di CloudWatch konsol. Saat Anda menggunakan CloudWatch konsol untuk memulai kueri, Anda menggunakan antarmuka konsol untuk menentukan grup log dan nama dan jenis sumber data.

Gunakan `aws:fieldIndex` untuk mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri. Grup log yang relevan dipilih secara otomatis, berdasarkan bidang yang ditentukan dalam `filterIndex` perintah. Ini mengurangi volume yang dipindai dengan melewati grup log yang tidak memiliki peristiwa log yang berisi bidang yang ditentukan dalam kueri, dan hanya memindai grup log yang cocok dengan nilai yang ditentukan dalam kueri untuk indeks bidang ini. Gunakan `aws:fieldIndex` untuk menentukan nama bidang, bersama dengan nama bidang dan nilai dalam perintah sumber untuk menanyakan hanya data yang diindeks yang berisi bidang dan nilai yang ditentukan. Untuk informasi selengkapnya, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#)

Anda dapat menggunakan OpenSearch PPL untuk kueri grup log di Kelas Log Standar.

Note

Untuk informasi tentang semua perintah kueri OpenSearch PPL yang didukung di CloudWatch Log dan informasi terperinci tentang sintaks dan pembatasan, lihat [perintah PPL yang didukung di Panduan](#) Pengembang OpenSearch Layanan.

Untuk informasi tentang bahasa kueri lain yang dapat Anda gunakan lihat, [Wawasan CloudWatch Log](#), [SQL OpenSearch Layanan](#), dan [CloudWatch Metrics Insights](#)

Perintah atau fungsi	Kueri contoh	Deskripsi
<code>fields</code>	<code>fields field1, field2</code>	Menampilkan satu set bidang yang membutuhkan proyeksi.
<code>join</code>	<code>LEFT JOIN left=l, right=r on l.id = r.id 'join_right_lg' fields l.field_1, r.field_2</code>	Bergabung dengan dua kumpulan data bersama-sama.
<code>where</code>	<code>where field1="success" where field2 != "i-023fe0a90929d8822" fields field3, field4, field5,field6 head 1000</code>	Memfilter data berdasarkan

Perintah atau fungsi	Kueri contoh	Deskripsi
		kondisi yang Anda tentukan.
<code>aws:fieldIndex</code>	<code>source = [`aws:fieldIndex`="region", `region` = "us-west-2"] where status = 200 head 10</code>	Mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri.
<code>stats</code>	<code>stats count(), count(field1), min(field1), max(field1), avg(field1) by field2 head 1000</code>	Melakukan agregasi dan perhitungan
<code>parse</code>	<code>parse field1 ".*/(?<field2>[^\r\n]+)\$" where field2 = "requestId" fields field1, field2 head 1000</code>	Mengekstrak pola ekspresi reguler (regex) dari string dan menampilkan pola yang diekstraksi. Pola yang diekstraksi dapat digunakan lebih lanjut untuk membuat bidang baru atau menyaring data.

Perintah atau fungsi	Kueri contoh	Deskripsi
sort	<pre>stats count(), count(field1), min(field1) as field1Alias, max(`field1`), avg(`field1`) by field2 sort -field1Alias head 1000</pre>	Urutkan hasil yang ditampilkan dengan nama bidang. Gunakan sort - fieldName untuk mengurutkan dalam urutan menurun.
eval	<pre>eval field2 = field1 * 2 fields field1, field2 head 20</pre>	Memodifikasi atau memproses nilai bidang dan menyimpannya di bidang yang berbeda. Ini berguna untuk memodifikasi kolom secara matematis, menerapkan fungsi string ke kolom, atau menerapkan fungsi tanggal ke kolom.
rename	<pre>rename field2 as field1 fields field1;</pre>	Mengganti nama satu atau beberapa bidang di hasil pencarian.

Perintah atau fungsi	Kueri contoh	Deskripsi
head	<code>fields `@message` head 20</code>	Membatasi hasil kueri yang ditampilkan ke baris N pertama.
top	<code>top 2 field1 by field2</code>	Menemukan nilai yang paling sering untuk bidang.
dedup	<code>dedup field1 fields field1, field2, field3</code>	Menghapus entri duplikat berdasarkan bidang yang Anda tentukan.
rare	<code>rare field1 by field2</code>	Menemukan nilai yang paling tidak sering dari semua bidang dalam daftar bidang.
subquery	<code>where field_1 IN [search source=`subquery_lg` fields field_2] fields id, field_1</code>	Melakukan kueri yang kompleks dan bersarang dalam pernyataan PPL Anda.
trendline	<code>trendline sma(2, field1) as field1Alias</code>	Menghitung rata-rata bergerak bidang.

Perintah atau fungsi	Kueri contoh	Deskripsi
eventStats	<code>eventstats sum(field1) by field2</code>	Memperkaya data acara Anda dengan statistik ringkasan yang dihitung. Ini menganalisis bidang tertentu dalam acara Anda, menghitung berbagai ukuran statistik, dan kemudian menambahkan hasil ini ke setiap peristiwa asli sebagai bidang baru.
expand	<code>eval tags_array_string = json_extract(`@message`, '\$.tags') eval tags_array = json_array(json_extract(tags_string, '\$[0]'), json_extract(tags_string, '\$[1]')) expand tags_array as color_tags</code>	Memecah bidang yang berisi beberapa nilai menjadi baris terpisah, membuat baris baru untuk setiap nilai di bidang yang ditentukan.

Perintah atau fungsi	Kueri contoh	Deskripsi
<code>fillnull</code>	<code>fields `@timestamp`, error_code, status_code fillnull using status_code = "UNKNOWN", error_code = "UNKNOWN"</code>	Mengisi bidang null dengan nilai yang Anda berikan. Ini dapat digunakan dalam satu atau lebih bidang.
<code>flatten</code>	<code>eval metadata_struct = json_object('size', json_extract(metadata_string, '\$.size'), 'color', json_extract(metadata_string, '\$.color')) flatten metadata_struct as (meta_size, meta_color)</code>	Meratakan bidang. Bidang harus dari jenis ini: <code>struct<?,?></code> atau <code>array<struct<?,?>></code>
<code>cidrmatch</code>	<code>where cidrmatch(ip, '2003:db8::/32') fields ip</code>	Memeriksa apakah alamat IP yang ditentukan berada dalam kisaran CIDR yang diberikan.
<code>fieldsummary</code>	<code>where field1 != 200 fieldsummary includefields= field1 nulls=true</code>	Menghitung statistik dasar untuk setiap bidang (hitungan, hitungan berbeda, min, maks, rata-rata, stddev, dan rata-rata).

Perintah atau fungsi	Kueri contoh	Deskripsi
grok	<code>grok email '.*@%{HOSTNAME:host}' fields email, host</code>	Mem-parsing bidang teks dengan pola grok dan menambahkan hasilnya ke hasil pencarian.
Fungsi string	<code>eval field1Len = LENGTH(field1) fields field1Len</code>	Fungsi bawaan di PPL yang dapat memanipulasi dan mengubah data string dan teks dalam kueri PPL. Misalnya, mengonversi kasus, menggabungkan string, mengekstraksi bagian, dan membersihkan teks.

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi Tanggal-Waktu	<code>eval newDate = ADDDATE(DATE('2020-08-26'), 1) fields newDate</code>	Fungsi bawaan untuk menangani dan mengubah data tanggal dan stempel waktu dalam kueri PPL. Misalnya, <code>date_add</code> , <code>date_format</code> , <code>datediff</code> , <code>date-sub</code> , <code>timestamp add</code> , <code>timestamp diff</code> , <code>current_timezone</code> , <code>utc_timestamp</code> , dan <code>current_date</code> .
Fungsi syarat	<code>eval field2 = isnull(field1) fields field2, field1, field3</code>	Fungsi bawaan yang memeriksa kondisi bidang tertentu, dan mengevaluasi ekspresi secara kondisional. Misalnya, jika <code>field1</code> adalah null, kembalikan <code>field2</code> .

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi matematika	<code>eval field2 = ACOS(field1) fields field1</code>	Fungsi bawaan untuk melakukan perhitungan dan transformasi matematis dalam kueri PPL. Misalnya, <code>abs</code> (nilai absolut), <code>bulat</code> (angka bulat), <code>sqrt</code> (akar kuadrat), <code>pow</code> (perhitungan daya), dan <code>ceil</code> (membulatkan ke bilangan bulat terdekat).
CryptoGraphic fungsi	<code>eval crypto = MD5(field) head 1000</code>	Untuk menghitung hash dari bidang yang diberikan

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi JSON	<pre>eval valid_json = json('[1,2,3,{"f1":1,"f2":[5,6]},4]') fields valid_json</pre>	Fungsi bawaan untuk menangani JSON termasuk array, ekstraksi, dan validasi. Misalnya, <code>json_object</code> , <code>json_array</code> , <code>to_json_string</code> , <code>json_array_length</code> , <code>json_extract</code> , <code>json_keys</code> , dan <code>json_valid</code> .

Lingkup kueri

Menyertakan SUMBER dalam kueri adalah cara yang berguna untuk menentukan grup log yang akan disertakan dalam kueri saat Anda menggunakan API AWS CLI atau untuk membuat kueri. Perintah `SOURCE` hanya didukung di API AWS CLI dan, bukan di CloudWatch konsol. Saat Anda menggunakan CloudWatch konsol untuk memulai kueri, Anda menggunakan antarmuka konsol untuk menentukan grup log dan nama dan jenis sumber data.

Perintah sumber PPL sekarang mendukung beberapa cara untuk menentukannya:

1. Grup log
2. Indeks bidang - Baru
3. Sumber dan tipe data - Baru

Grup Log

Pemilihan sumber Grup Log dapat digunakan ketika pelanggan mengetahui grup log yang tepat yang perlu dicari

```
source = [lg:`/aws/lambda/my-function`] | where status = 200 | head 10
```

Indeks Lapangan

Pemilihan sumber berbasis indeks bidang mengurangi jumlah data yang ditanyakan dengan membatasi hasil hanya pada data yang diindeks saat filter Anda menargetkan bidang yang telah diindeks. Grup log yang relevan dipilih secara otomatis, berdasarkan bidang yang ditentukan dalam `filterIndex` perintah. Untuk informasi selengkapnya tentang indeks bidang dan cara membuatnya, lihat [Membuat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Gunakan `aws:fieldIndex` untuk mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri. Untuk grup log ini yang diindeks di bidang ini, selanjutnya mengoptimalkan kueri dengan melewati grup log yang tidak memiliki peristiwa log yang berisi bidang yang ditentukan dalam kueri untuk bidang yang diindeks. Selanjutnya mengurangi volume yang dipindai dengan mencoba memindai hanya peristiwa log dari grup log ini yang cocok dengan nilai yang ditentukan dalam kueri untuk indeks bidang ini. Untuk informasi selengkapnya tentang indeks bidang dan cara membuatnya, lihat [Membuat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Dalam PPL, `aws:fieldIndex` digunakan untuk menentukan pasangan nilai kunci mana yang harus diperlakukan sebagai indeks. Sintaksnya adalah sebagai berikut

```
source = [aws:fieldIndex="region", `region` = "us-west-2"] | where status = 200 | head 10
```

dimana,

1. ``aws:fieldIndex`="region"` mengidentifikasi wilayah sebagai Indeks bidang.
 - a. Catatan: Alih-alih = pelanggan dapat menggunakan IN untuk menentukan beberapa indeks (contoh di bawah)
2. ``region`="us-west-2"` mengidentifikasi kondisi filter yang akan diterapkan
 - a. Catatan: Alih-alih = pelanggan dapat menggunakan IN untuk menentukan beberapa nilai (contoh di bawah)

Pelanggan dapat menentukan beberapa `FieldIndex` sebagai berikut

```
source = [`aws:fieldIndex` IN ("status", "region"), `status` = 200, `region` IN ("us-west-2", "us-east-1")] | head 10
```

Sumber dan Jenis Data

Sumber data dan pemilihan sumber berbasis tipe dapat digunakan ketika pelanggan mengetahui sumber data pasti mana yang perlu ditanyakan. Kueri ini dijalankan melalui satu atau lebih grup log yang berisi sumber dan jenis data yang ditentukan.

```
source = [ds:`data_source.type`] | where status = 200 | head 10
```

PPL yang didukung untuk kueri sumber data

Untuk mendukung kasus penggunaan kueri sumber data di PPL, Anda dapat menggunakan klausa pemilih sumber dinamis. Dengan menggunakan sintaks ini, Anda dapat meminta sumber data dengan menentukannya dalam perintah pencarian. Anda dapat menentukan hingga 10 sumber data.

Sintaksis

```
source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`, ...ds:`DataSourceN.TypeN`]
```

Contoh kueri

```
search source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`] | fields field1, field2
```

Contoh gabungan

Pelanggan dapat menentukan semua operator pemilihan sumber dalam urutan apa pun & hasilnya akan menjadi persimpangan dari semua kondisi yang diterapkan.

Misalnya, `/aws/lambda/my-function-1` mungkin berisi beberapa sumber & tipe data termasuk berbagai indeks, ketika kueri berikut dijalankan, hasil yang dikembalikan hanya akan memiliki peristiwa sumber dan tipe `DataSource 1.Type1` dan cocok dengan kriteria `'status' = 200`.

```
search source=[
  ds:`DataSource1.Type1`,
  lg:`/aws/lambda/my-function-1`,
  `aws:fieldIndex` IN ("status"), `status` = 200
```


]

Pembatasan

Pembatasan berikut berlaku saat Anda menggunakan OpenSearch PPL untuk melakukan kueri di Wawasan CloudWatch Log.

- Anda tidak dapat menggunakan perintah bergabung atau subquery dengan kueri sumber data.

OpenSearch Bahasa Kueri Terstruktur (SQL)

Bagian ini berisi pengantar dasar untuk query CloudWatch Log menggunakan OpenSearch SQL. Ini memberikan opsi yang akrab jika Anda terbiasa bekerja dengan database relasional. OpenSearch SQL menawarkan subset fungsionalitas SQL, menjadikannya pilihan yang baik untuk melakukan kueri ad-hoc dan tugas analisis data. Dengan OpenSearch SQL, Anda dapat menggunakan perintah seperti SELECT, FROM, WHERE, GROUP BY, HAVING, dan berbagai perintah dan fungsi SQL lainnya. Anda dapat mengeksekusi JOINS di seluruh grup log, mengkorelasikan data di seluruh grup log menggunakan sub-kueri, dan menggunakan kumpulan kaya fungsi JSON, matematika, string, kondisional, dan SQL lainnya untuk melakukan analisis yang kuat pada data log dan keamanan.

Gunakan `filterIndex` untuk mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri. Kurangi volume yang dipindai dengan melewati grup log yang tidak memiliki peristiwa log yang berisi bidang yang ditentukan dalam kueri, dan hanya memindai grup log yang cocok dengan nilai yang ditentukan dalam kueri untuk indeks bidang ini. Gunakan `filterIndex` untuk menentukan nama bidang, bersama dengan nama bidang dan nilai untuk kueri hanya data yang diindeks yang berisi bidang dan nilai yang ditentukan.

Anda dapat menggunakan OpenSearch SQL untuk kueri grup log di Kelas Log Standar. SQL juga mendukung query menggunakan nama sumber data dan tipe sumber data.

Note

Tabel berikut mencantumkan perintah dan fungsi SQL yang didukung di CloudWatch Log. Untuk informasi tentang semua perintah OpenSearch SQL termasuk sintaks, lihat perintah [SQL yang didukung dalam Panduan Pengembang](#) OpenSearch Layanan.

Untuk informasi tentang bahasa kueri lain yang dapat Anda gunakan, lihat [Wawasan CloudWatch Log](#), [PPL OpenSearch Layanan](#), dan [CloudWatch Metrics Insights](#).

Perintah SQL yang didukung

Note

Di kolom query contoh, ganti sesuai *<logGroup>* kebutuhan tergantung pada sumber data yang Anda kueri.

Perintah atau fungsi	Kueri contoh	Deskripsi
SELECT	SELECT `@message`, Operation FROM `LogGroupA`	Menampilkan nilai yang diproyeksikan.
FROM	SELECT `@message`, Operation FROM `LogGroupA`	Klausula bawaan yang menentukan tabel sumber atau tampilan untuk mengambil data, mendukung berbagai jenis gabungan dan subkueri.
WHERE	SELECT * FROM `LogGroupA` WHERE Operation = 'x'	Memfilter peristiwa log berdasarkan kriteria bidang yang disediakan.
FilterIndex	SELECT * FROM `filterIndex('region' = 'us-east-1')` WHERE status = 200 LIMIT 10;	Mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai

Perintah atau fungsi	Kueri contoh	Deskripsi
		hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri.
GROUP BY	<code>SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream`</code>	Grup mencatat peristiwa berdasarkan kategori dan menemukan rata-rata berdasarkan statistik.
MEMILIKI	<code>SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream` HAVING log_count > 100</code>	Memfilter hasil berdasarkan kondisi pengelompokan.
ORDER BY	<code>SELECT * FROM `LogGroupA` ORDER BY `@timestamp` DESC</code>	Memesan hasil berdasarkan bidang dalam klausa ORDER BY. Anda dapat mengurutkan dalam urutan menurun atau naik.

Perintah atau fungsi	Kueri contoh	Deskripsi
BERGABUNG	<code>SELECT A.`@message`, B.`@timestamp` FROM `LogGroupA` as A INNER JOIN `LogGroupB` as B ON A.`requestId` = B.`requestId`</code>	Bergabung dengan hasil untuk dua tabel berdasarkan bidang umum. Inner JOIN atau Left Outer Join harus ditentukan
LIMIT	<code>Select * from `LogGroupA` limit 10</code>	Membatasi hasil kueri yang ditampilkan ke baris N pertama.
Fungsi string	<code>SELECT upper(Operation) , lower(Operation), Operation FROM `LogGroupA`</code>	Fungsi bawaan di SQL yang dapat memanipulasi dan mengubah string dan data teks dalam kueri SQL. Misalnya, mengonversi kasus, menggabungkan string, mengekstraksi bagian, dan membersihkan teks.

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi tanggal	<pre>SELECT current_date() as today, date_add(current_date(), 30) as thirty_days_later, last_day(current_date()) as month_end FROM `LogGroupA`</pre>	Fungsi bawaan untuk menangani dan mengubah data tanggal dan stempel waktu dalam kueri SQL. Misalnya, <code>date_add</code> , <code>date_format</code> , <code>datediff</code> , dan <code>current_date</code> .
Fungsi kondisional	<pre>SELECT Operation, IF(Error > 0, 'High', 'Low') as error_category FROM `LogGroupA`;</pre>	Fungsi bawaan yang melakukan tindakan berdasarkan kondisi tertentu, atau yang mengevaluasi ekspresi secara kondisional. Misalnya, <code>CASE</code> dan <code>IF</code> .

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi agregat	<code>SELECT AVG(bytes) as bytesWritten FROM `LogGroupA`</code>	Fungsi bawaan yang melakukan perhitungan pada beberapa baris untuk menghasilkan nilai ringkasan tunggal. Misalnya, SUM, COUNT, AVG, MAX, dan MIN.
Fungsi JSON	<code>SELECT get_json_object(json_column, '\$.name') as name FROM `LogGroupA`</code>	Fungsi bawaan untuk mengurai, mengekstrak, memodifikasi, dan menanyakan data berformat JSON dalam kueri SQL (misalnya <code>from_json</code> , <code>to_json</code> , <code>get_json_object</code> , <code>json_tuple</code>) yang memungkinkan manipulasi struktur JSON dalam kumpulan data.

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi array	<pre>SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;</pre>	Fungsi bawaan untuk bekerja dengan kolom tipe array dalam kueri SQL, memungkinkan operasi seperti mengakses, memodifikasi, dan menganalisis data array (misalnya, size, explode, array_contains).
Fungsi jendela	<pre>SELECT field1, field2, RANK() OVER (ORDER BY field2 DESC) as field2Rank FROM `LogGroupA`;</pre>	Fungsi bawaan yang melakukan perhitungan di serangkaian baris tertentu yang terkait dengan baris (jendela) saat ini, memungkinkan operasi seperti peringkat, total berjalan, dan rata-rata bergerak. Misalnya, ROW_NUMBER, RANK, LAG, dan LEAD

Perintah atau fungsi	Kueri contoh	Deskripsi
Fungsi konversi	<pre>SELECT CAST('123' AS INT) as converted_number, CAST(123 AS STRING) as converted_string FROM `LogGroupA`</pre>	Fungsi bawaan untuk mengonversi data dari satu jenis ke jenis lainnya dalam kueri SQL, memungkinkan transformasi tipe data dan konversi format. Misalnya, CAST, TO_DATE, TO_TIMESTAMP, dan BINARY.
Fungsi predikat	<pre>SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;</pre>	Fungsi bawaan yang mengevaluasi kondisi dan mengembalikan nilai boolean (benar/salah) berdasarkan kriteria atau pola yang ditentukan. Misalnya, IN, LIKE, BETWEEN, IS NULL, dan EXISTS.

Perintah atau fungsi	Kueri contoh	Deskripsi
Pilih beberapa grup log	<pre>SELECT lg1.field1, lg1.field2 from `logGroups( logGroupIdentifier: ['LogGroup1', 'LogGroup2'])` as lg1 where lg1.field3= "Success"</pre>	Memungkinkan Anda menentukan beberapa grup log dalam pernyataan SELECT
Pilih beberapa sumber data	<pre>SELECT ds1.field1, ds1.field2 from `dataSource(['DataSource1', 'DataSour ce2'])` as ds1 where ds1.field3= "Success"</pre>	Memungkinkan Anda menentukan beberapa sumber data dalam pernyataan SELECT

SQL yang didukung untuk multi-log-group kueri

Untuk mendukung kasus penggunaan untuk menanyakan beberapa grup log di SQL, Anda dapat menggunakan perintah. `logGroups` Dengan menggunakan sintaks ini, Anda dapat menanyakan beberapa grup log dengan menentukannya dalam perintah FROM.

Sintaksis:

```
`logGroups(
  logGroupIdentifier: ['LogGroup1', 'LogGroup2', ... 'LogGroupn']
)
```

Dalam sintaks ini, Anda dapat menentukan hingga 50 grup log dalam `logGroupIdentifier` parameter. Untuk mereferensikan grup log di akun pemantauan, gunakan ARNs alih-alih LogGroup nama.

Contoh kueri:

```
SELECT LG1.Column1, LG1.Column2 from `logGroups(
  logGroupIdentifier: ['LogGroup1', 'LogGroup2']
)` as LG1 WHERE LG1.Column1 = 'ABC'
```

Sintaks berikut yang melibatkan beberapa grup log setelah FROM pernyataan TIDAK didukung saat menanyakan CloudWatch Log.

```
SELECT Column1, Column2 FROM 'LogGroup1', 'LogGroup2', ... 'LogGroupn'  
WHERE Column1 = 'ABC'
```

SQL yang didukung untuk kueri sumber data

Untuk mendukung kasus penggunaan kueri sumber data di SQL, Anda dapat menggunakan perintah DataSource. Dengan menggunakan sintaks ini, Anda dapat meminta sumber data dengan menentukannya dalam perintah. FROM Anda dapat menentukan hingga 10 sumber data.

Sintaksis

```
`dataSource(  
    ['DataSource1', 'DataSource2', ... 'DataSourcen']  
)`
```

Contoh kueri

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(  
    ['DataSource1', 'DataSource2']  
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

Lingkup kueri

Di API AWS CLI dan, Anda dapat menentukan log mana yang akan dikueri dengan menggunakan grup log, sumber dan jenis data, serta indeks bidang.

Grup Log

Pemilihan sumber Grup Log dapat digunakan ketika pelanggan mengetahui grup log yang tepat yang perlu dicari

```
SELECT * FROM `logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])`;
```

Sumber dan Jenis Data

Pelanggan dapat menanyakan log mereka menggunakan nama sumber data dan tipe sumber data.

Sumber data dan pemilihan sumber berbasis tipe dapat digunakan ketika pelanggan mengetahui sumber data pasti mana yang perlu ditanyakan. Kueri ini dijalankan melalui satu atau lebih grup log yang berisi sumber dan jenis data yang ditentukan.

Untuk mendukung kasus penggunaan kueri sumber data di SQL, Anda dapat menggunakan perintah DataSource. Dengan menggunakan sintaks ini, Anda dapat meminta sumber data dengan menentukannya dalam perintah FROM. Anda dapat menentukan hingga 10 sumber data.

Sintaksis:

```
`dataSource(  
    ['DataSource1.Type1', 'DataSource2.Type2', ...'DataSourceN.TypeN']  
)`
```

Contoh kueri:

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(  
    ['DataSource1.Type1', 'DataSource2.Type2']  
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

Untuk informasi selengkapnya tentang kueri berdasarkan sumber data, lihat [Gunakan aspek untuk mengelompokkan dan menjelajahi log](#).

Contoh gabungan

Pelanggan dapat menentukan semua operator pemilihan sumber dalam backticks dalam urutan apa pun dan hasilnya akan didasarkan pada persimpangan semua kondisi yang diterapkan.

Misalnya, `/aws/lambda/my-function-1` mungkin berisi beberapa sumber & tipe data termasuk berbagai indeks, ketika kueri berikut dijalankan, hasil yang dikembalikan hanya akan memiliki peristiwa sumber dan tipe DataSource 1.Type1 dan cocok dengan kriteria `'status' = 200`.

```
SELECT * FROM `  
    logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])  
    filterIndex('status' = 200)  
    dataSource(['DataSource1.Type1'])  
`;  
`;
```

Indeks Lapangan

Pemilihan sumber berbasis Indeks Bidang secara otomatis mengidentifikasi grup log yang relevan saat filter Anda menargetkan bidang yang diindeks, mengurangi volume pemindaian dan runtime kueri.

Gunakan `filterIndex` untuk mengembalikan data yang diindeks saja, dengan memaksa kueri untuk memindai hanya grup log yang diindeks pada bidang yang Anda tentukan dalam kueri. Untuk grup log ini yang diindeks di bidang ini, selanjutnya mengoptimalkan kueri dengan melewati grup log yang tidak memiliki peristiwa log yang berisi bidang yang ditentukan dalam kueri untuk bidang yang diindeks. Selanjutnya mengurangi volume yang dipindai dengan mencoba memindai hanya peristiwa log dari grup log ini yang cocok dengan nilai yang ditentukan dalam kueri untuk indeks bidang ini. Untuk informasi selengkapnya tentang indeks bidang dan cara membuatnya, lihat [Membuat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Dalam SQL, `FilterIndex` digunakan untuk menentukan pasangan nilai kunci mana yang harus diperlakukan sebagai indeks. Sintaksnya adalah sebagai berikut

```
SELECT * FROM `filterIndex('region' = 'us-east-1')`;
```

dimana,

1. `filterIndex (...)` menentukan, memperlakukan nilai-nilai kunci di dalamnya sebagai indeks bidang. Setiap pasangan nilai kunci dipisahkan dengan koma (contoh di bawah)
2. `'region' = 'us-east-1'` menentukan kondisi aktual yang akan diterapkan
 - a. Catatan: Alih-alih = pelanggan dapat menggunakan `IN` untuk menentukan beberapa nilai (contoh di bawah)

Menggunakan beberapa `filterIndex` akan menggabungkan kondisi menggunakan “AND”. Dalam contoh, log yang cocok dengan `status=200` dan `region` di `us-east-1` atau `us-west-2` akan ditanyakan.

```
SELECT * FROM `filterIndex('status' = 200, 'region' IN ['us-east-1', 'us-west-2'])`;
```

Pembatasan

Pembatasan berikut berlaku saat Anda menggunakan OpenSearch SQL untuk melakukan kueri di Wawasan CloudWatch Log.

- Anda hanya dapat menyertakan satu `JOIN` dalam pernyataan `SELECT`.

- Anda tidak dapat menggunakan JOIN atau subquery dengan kueri sumber data.
- Hanya satu tingkat subkueri bersarang yang didukung.
- Beberapa kueri pernyataan yang dipisahkan oleh titik dua (;) tidak didukung.
- Kueri yang berisi nama bidang yang identik tetapi berbeda hanya dalam kasus (seperti field1 dan FIELD1) tidak didukung.

Misalnya, kueri berikut tidak didukung:

```
Select AWSAccountId, AwsAccountId from LogGroup
```

Namun, kueri berikut didukung karena nama bidang (@logStream) identik di kedua grup log:

```
Select a.`@logStream`, b.`@logStream` from Table A INNER Join Table B on a.id = b.id
```

- Fungsi dan ekspresi harus beroperasi pada nama bidang dan menjadi bagian dari pernyataan SELECT dengan grup log yang ditentukan dalam klausa FROM.

Misalnya, kueri ini tidak didukung:

```
SELECT cos(10) FROM LogGroup
```

Kueri ini didukung:

```
SELECT cos(field1) FROM LogGroup
```

- Saat menggunakan perintah SQL atau PPL, lampirkan bidang tertentu di backticks agar berhasil menanyakannya. Backticks diperlukan untuk bidang dengan karakter khusus (non-alfabet dan non-numerik). Misalnya, lampirkan, @messageOperation.Export, dan Test::Field di backticks. Anda tidak perlu melampirkan bidang dengan nama abjad murni di backticks.

Contoh kueri dengan bidang sederhana:

```
SELECT SessionToken, Operation, StartTime FROM `LogGroup-A`  
LIMIT 1000;
```

Kueri serupa dengan backticks ditambahkan:

```
SELECT `@SessionToken`, `@Operation`, `@StartTime` FROM `LogGroup-A` LIMIT 1000;
```

Gunakan bahasa alami untuk menghasilkan dan memperbarui kueri Wawasan CloudWatch Log

CloudWatch Log mendukung kemampuan kueri bahasa alami untuk membantu Anda menghasilkan dan memperbarui kueri untuk [Wawasan CloudWatch Log](#), [PPL OpenSearch Layanan](#), [SQL OpenSearch Layanan](#), dan [CloudWatch Metrics Insights](#).

Dengan kemampuan ini, Anda dapat mengajukan pertanyaan tentang atau menjelaskan data CloudWatch Log yang Anda cari dalam bahasa Inggris biasa. Kemampuan bahasa alami menghasilkan kueri berdasarkan prompt yang Anda masukkan dan memberikan line-by-line penjelasan tentang cara kerja kueri. Anda juga dapat memperbarui kueri Anda untuk menyelidiki lebih lanjut data Anda.

Bergantung pada lingkungan Anda, Anda dapat memasukkan petunjuk seperti “Apa 100 alamat IP sumber teratas berdasarkan byte yang ditransfer?” dan “Temukan 10 permintaan fungsi Lambda paling lambat.”

Note

Fitur kueri bahasa alami adalah layanan Regional. Untuk beberapa Wilayah, fitur ini membuat panggilan lintas wilayah ke Wilayah di Amerika Serikat untuk memproses permintaan kueri. Untuk informasi selengkapnya, lihat [Amazon CloudWatch memperluas dukungan wilayah untuk ringkasan hasil kueri bahasa alami dan pembuatan kueri](#).

Untuk menghasilkan kueri Wawasan CloudWatch Log dengan kemampuan ini, buka editor kueri Wawasan CloudWatch Log, pilih grup log yang ingin Anda kueri, dan pilih Buat kueri.

Important

Untuk menggunakan kemampuan kueri bahasa alami, Anda harus masuk dengan kebijakan [CloudWatchLogsFullAccess](#), [CloudWatchLogsReadOnlyAccess](#), [AdministratorAccess](#), atau [ReadOnlyAccess](#) IAM, atau memiliki `cloudwatch:GenerateQuery` izin.

Kueri contoh

Contoh di bagian ini menjelaskan cara menghasilkan dan memperbarui kueri menggunakan kemampuan bahasa alami.

Note

Untuk informasi selengkapnya tentang editor dan sintaks kueri Wawasan CloudWatch Log, lihat sintaks kueri [Wawasan CloudWatch Log](#).

Contoh: Menghasilkan kueri bahasa alami

Untuk menghasilkan kueri menggunakan bahasa alami, masukkan prompt dan pilih Hasilkan kueri baru. Contoh ini menunjukkan kueri yang melakukan pencarian dasar.

Prompt

Berikut ini adalah contoh prompt yang mengarahkan kemampuan untuk mencari 10 pemanggilan fungsi Lambda paling lambat.

```
Find the 10 slowest requests
```

Kueri

Berikut ini adalah kueri menggunakan bahasa kueri Wawasan CloudWatch Log yang dihasilkan oleh kemampuan bahasa alami berdasarkan prompt. Perhatikan bagaimana prompt muncul di komentar sebelum kueri. Setelah kueri, Anda dapat membaca penjelasan yang menggambarkan cara kerja kueri.

```
# Find the 10 slowest requests
fields @timestamp, @message, @duration
| sort @duration desc
| limit 10
# This query retrieves the timestamp, message and duration fields from the logs and
sorts them in descending order by duration to find the 10 slowest requests.
```

 Note

Untuk mematikan tampilan prompt Anda dan penjelasan tentang cara kerja kueri, gunakan ikon roda gigi di editor Anda.

Prompt

Untuk menghasilkan kueri OpenSearch SQL, pilih tab OpenSearch SQL, lalu buka kotak prompt generator kueri untuk memasukkan prompt bahasa alami Anda. Berikut ini adalah contoh prompt yang menggunakan kemampuan bahasa alami untuk menghasilkan query OpenSearch SQL.

```
Give me the number of errors and exceptions per hour
```

Kueri

Berikut ini adalah kueri SQL yang dihasilkan oleh prompt yang dapat Anda gunakan untuk menemukan jumlah kesalahan dan pengecualian yang dikumpulkan per jam:

```
SELECT DATE_FORMAT(`@timestamp`, 'yyyy-MM-dd HH') AS hour,  
       COUNT(*) AS error_count  
FROM `aws/lambda/CloudWatchOdysseyQueryGen`  
WHERE `@message` LIKE '%error%'  
      OR `@message` LIKE '%exception%'  
GROUP BY DATE_FORMAT(`@timestamp`, 'yyyy-MM-dd HH')  
ORDER BY hour
```

Prompt

Untuk menghasilkan kueri OpenSearch PPL, pilih tab OpenSearch PPL, lalu buka kotak prompt generator kueri untuk memasukkan prompt bahasa alami Anda. Berikut ini adalah contoh prompt yang menggunakan kemampuan bahasa alami untuk menghasilkan kueri OpenSearch PPL.

```
Give me all unique exception messages
```

Kueri

Berikut ini adalah kueri PPL yang dihasilkan oleh prompt yang dapat Anda gunakan untuk menemukan pesan pengecualian unik di log Anda:


```
dedup @message  
| fields @message
```

Contoh: Memperbarui kueri bahasa alami

Anda dapat memperbarui kueri dengan mengedit prompt awal dan kemudian memilih Perbarui kueri.

Prompt yang diperbarui

Contoh berikut menunjukkan versi yang diperbarui dari prompt sebelumnya. Alih-alih prompt yang mencari 10 pemanggilan fungsi Lambda paling lambat, prompt ini sekarang mengarahkan kemampuan untuk mencari 20 pemanggilan fungsi Lambda paling lambat dan menyertakan kolom lain untuk peristiwa log tambahan.

```
Show top 20 slowest requests instead and display requestId as a column
```

Kueri yang diperbarui

Berikut ini adalah contoh kueri yang diperbarui menggunakan bahasa kueri Wawasan CloudWatch Log. Perhatikan bagaimana prompt muncul di komentar sebelum kueri yang diperbarui. Setelah kueri, Anda dapat membaca penjelasan yang menggambarkan bagaimana kueri asli diperbarui.

```
# Show top 20 slowest requests instead and display requestId as a column  
fields @timestamp, @message, @requestId, @duration  
| sort @duration desc  
| limit 20  
# This query modifies the original query by replacing the @message field with the  
  @requestId field and changing the limit from 10 to 20 to return the top 20 log events  
  by duration instead of the top 10.
```

Memilih untuk tidak menggunakan data Anda untuk perbaikan layanan

Data prompt bahasa alami yang Anda berikan untuk melatih model AI dan menghasilkan kueri yang relevan digunakan semata-mata untuk menyediakan dan memelihara layanan Anda. Data ini dapat digunakan untuk meningkatkan kualitas Wawasan CloudWatch Log. Kepercayaan dan privasi Anda, serta keamanan konten Anda, menjadi prioritas utama kami. Untuk informasi selengkapnya, silakan lihat [AWS Ketentuan Layanan](#) dan [AWS kebijakan AI yang bertanggung jawab](#).

Anda dapat memilih untuk tidak menggunakan konten Anda untuk mengembangkan atau memperbaiki mutu kueri bahasa alami dengan membuat kebijakan penolakan layanan AI. Untuk memilih keluar dari pengumpulan data untuk semua fitur AI CloudWatch Log, termasuk kemampuan pembuatan kueri, Anda harus membuat kebijakan opt-out untuk Log. CloudWatch Untuk informasi selengkapnya, silakan lihat [kebijakan penolakan layanan AI](#) di AWS Organizations Panduan Pengguna.

Log yang didukung dan bidang yang ditemukan

CloudWatch Log Insights mendukung berbagai jenis log. Untuk setiap log yang dikirim ke grup log kelas Standar di Log Amazon CloudWatch, Wawasan CloudWatch Log secara otomatis menghasilkan lima bidang sistem:

- `@message` berisi log acara mentah yang belum diurai. Ini setara dengan message bidang di [InputLogevent](#).
- `@timestamp` berisi stempel waktu acara di bidang peristiwa log. `timestamp` Ini setara dengan `timestamp` bidang di [InputLogevent](#).
- `@ingestionTime` berisi waktu ketika CloudWatch Log menerima peristiwa log.
- `@logStream` berisi nama pengaliran log yang ditambahi log acara. Log mengalirkan log grup melalui proses yang sama yang menghasilkannya.
- `@log` adalah pengidentifikasi grup log dalam bentuk *account-id:log-group-name*. Saat menanyakan beberapa grup log, ini dapat berguna untuk mengidentifikasi grup log mana yang termasuk dalam acara tertentu.
- `@entity` berisi JSON pipih yang terkait dengan entitas untuk fitur telemetri [terkait Jelajahi](#).

Misalnya, JSON ini dapat mewakili entitas.

```
{
  "Entity": {
    "KeyAttributes": {
      "Type": "Service",
      "Name": "PetClinic"
    },
    "Attributes": {
      "PlatformType": "AWS::EC2",
      "EC2.InstanceId": "i-1234567890123"
    }
  }
}
```

```
}
```

Untuk entitas ini, bidang sistem yang diekstraksi adalah sebagai berikut:

```
@entity.KeyAttributes.Type = Service
@entity.KeyAttributes.Name = PetClinic
@entity.Attributes.PlatformType = AWS::EC2
@entity.Attributes.EC2.InstanceId = i-1234567890123
```

Note

Penemuan bidang hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

CloudWatch Logs Insights menyisipkan simbol @ di awal bidang yang dihasilkannya.

Untuk banyak jenis CloudWatch log, Log juga secara otomatis menemukan bidang log yang terdapat dalam log. Bidang penemuan otomatis ini ditunjukkan dalam tabel berikut.

Untuk jenis log lain dengan bidang yang tidak ditemukan secara otomatis oleh Wawasan CloudWatch Log, Anda dapat menggunakan parse perintah untuk mengekstrak dan membuat bidang yang diekstrak untuk digunakan dalam kueri tersebut. Untuk informasi selengkapnya, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

Jika nama bidang log yang ditemukan dimulai dengan @ karakter, Wawasan CloudWatch Log akan menampilkannya dengan tambahan yang @ ditambahkan ke awal. Sebagai contoh, jika nama bidang log adalah @example.com, nama bidang ini ditampilkan sebagai @@example.com.

Note

Kecuali untuk @message@timestamp,, atau@log, Anda dapat membuat indeks bidang untuk bidang yang ditemukan. Untuk informasi selengkapnya tentang indeks bidang, lihat [Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

Jenis log	Bidang log yang ditemukan
Log alur Amazon VPC	@timestamp , @logStream , @message, accountId , endTime, interfaceId , logStatus , startTime , version, action, bytes, dstAddr, dstPort, packets, protocol, srcAddr, srcPort
Log Route 53	@timestamp , @logStream , @message, edgeLocation , ednsClientSubnet , hostZoneId , protocol, queryName , queryTimestamp , queryType , resolverIp , responseCode , version
Log Lambda	@timestamp , @logStream , @message, @requestId , @duration, @billedDuration , @type, @maxMemoryUsed , @memorySize Jika baris log Lambda berisi ID jejak X-Ray, itu juga mencakup bidang berikut: @xrayTraceId dan @xraySegmentId . CloudWatch Logs Insights secara otomatis menemukan bidang log di log Lambda, tetapi hanya untuk fragmen JSON pertama yang disematkan di setiap peristiwa log. Jika log acara Lambda berisi beberapa fragmen JSON, Anda dapat mengurai dan mengekstraksi bidang log menggunakan perintah parse. Untuk informasi selengkapnya, lihat Bidang di log JSON.
CloudTrail log	Untuk informasi selengkapnya, lihat Bidang di log JSON .
Log dalam format JSON	
Jenis log lainnya	@timestamp , @ingestionTime , @logStream , @message, @log.

Bidang di log JSON

Dengan Wawasan CloudWatch Log, Anda menggunakan notasi titik untuk mewakili bidang JSON. Bagian ini berisi contoh peristiwa JSON dan cuplikan kode yang menunjukkan bagaimana Anda dapat mengakses bidang JSON menggunakan notasi titik.

Contoh: acara JSON

```
{
```

```

    "eventVersion": "1.0",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EX_PRINCIPAL_ID",
      "arn": "arn: aws: iam: : 123456789012: user/Alice",
      "accessKeyId": "EXAMPLE_KEY_ID",
      "accountId": "123456789012",
      "userName": "Alice"
    },
    "eventTime": "2014-03-06T21: 22: 54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "ec2-api-tools1.6.12.2",
    "requestParameters": {
      "instancesSet": {
        "items": [
          {
            "instanceId": "i-abcde123"
          }
        ]
      }
    },
    "responseElements": {
      "instancesSet": {
        "items": [
          {
            "instanceId": "i-abcde123",
            "currentState": {
              "code": 0,
              "name": "pending"
            },
            "previousState": {
              "code": 80,
              "name": "stopped"
            }
          }
        ]
      }
    }
  }
}

```

Contoh acara JSON berisi objek yang bernama `userIdentity`. `userIdentity` berisi bidang yang diberi nama `type`. Untuk mewakili nilai `type` menggunakan notasi titik, Anda menggunakan `userIdentity.type`.

Contoh acara JSON berisi array yang diratakan ke daftar nama bidang bersarang dan nilai. Untuk mewakili nilai `instanceId` untuk item pertama di `requestParameters.instancesSet`, Anda menggunakan `requestParameters.instancesSet.items.0.instanceId`. Angka `0` yang ditempatkan sebelum bidang `instanceId` mengacu pada posisi nilai untuk bidang `items`. Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana Anda dapat mengakses bidang JSON bersarang dalam peristiwa log JSON.

Contoh: Query

```
fields @timestamp, @message
| filter requestParameters.instancesSet.items.0.instanceId="i-abcde123"
| sort @timestamp desc
```

Cuplikan kode menunjukkan kueri yang menggunakan notasi titik dengan `filter` perintah untuk mengakses nilai bidang JSON bersarang. `instanceId` Kueri menyaring pesan di mana nilai `instanceId` sama `"i-abcde123"` dan mengembalikan semua peristiwa log yang berisi nilai yang ditentukan.

Note

CloudWatch Wawasan Log dapat mengekstrak maksimal 200 bidang peristiwa log dari log JSON. Untuk bidang tambahan yang tidak diekstrak, Anda dapat menggunakan `parse` perintah untuk mengekstrak bidang dari peristiwa log mentah yang tidak diurai di bidang pesan. Untuk informasi selengkapnya tentang `parse` perintah, lihat [Sintaks kueri](#) di Panduan CloudWatch Pengguna Amazon.

Buat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian

Anda dapat membuat indeks bidang bidang dalam peristiwa log Anda untuk pencarian berbasis kesetaraan yang efisien. Saat Anda kemudian menggunakan indeks bidang dalam kueri Wawasan CloudWatch Log, kueri akan mencoba melewati pemrosesan peristiwa log yang diketahui

tidak menyertakan bidang yang diindeks. Ini mengurangi volume pemindaian kueri Anda yang menggunakan indeks bidang, sehingga memungkinkan untuk mengembalikan hasil lebih cepat. Ini dapat membantu Anda dengan cepat mencari petabyte dari total log di ribuan grup log, dan mengasah log yang relevan lebih cepat. Bidang yang baik untuk diindeks adalah bidang yang sering perlu Anda kueri. Bidang yang memiliki nilai kardinalitas tinggi juga merupakan kandidat yang baik untuk indeks bidang karena kueri yang menggunakan indeks bidang ini akan selesai lebih cepat karena membatasi peristiwa log yang dicocokkan dengan nilai target.

Misalnya, Anda telah membuat indeks bidang untuk `requestId`. Kemudian, setiap kueri CloudWatch Logs Insights pada grup log yang menyertakan `requestId = value` atau `requestId IN [value, value, ...]` akan mencoba memproses hanya peristiwa log yang diketahui berisi bidang yang diindeks dan nilai kueri, dan bahwa CloudWatch Log telah mendeteksi nilai untuk bidang tersebut di masa lalu.

Anda juga dapat memanfaatkan indeks bidang Anda untuk membuat kueri efisien dari jumlah grup log yang lebih besar. Saat Anda menggunakan `filterIndex` perintah dalam kueri alih-alih `filter` perintah, kueri akan berjalan terhadap grup log yang dipilih pada peristiwa log yang memiliki indeks bidang. Kueri ini dapat memindai sebanyak 10.000 grup log yang Anda pilih dengan menentukan sebanyak lima awalan nama grup log. Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih semua akun sumber atau menentukan akun sumber individu untuk memilih grup log”.

Bidang yang diindeks peka huruf besar/kecil. Misalnya, indeks bidang tidak `RequestId` akan cocok dengan peristiwa log yang berisi `requestId`.

Indeks bidang didukung hanya untuk format log terstruktur JSON dan log layanan.

CloudWatch Log menyediakan indeks bidang default untuk semua grup log di kelas log Standar. Indeks bidang default secara otomatis tersedia untuk bidang berikut:

- `@logStream`
- `@aws.region`
- `@aws.account`
- `@source.log`
- `@data_source_name`
- `@data_source_type`
- `@data_format`

- `traceId`
- `severityText`
- `attributes.session.id`

CloudWatch Log menyediakan indeks bidang default untuk nama sumber data tertentu dan kombinasi tipe juga. Indeks bidang default secara otomatis tersedia untuk kombinasi nama dan jenis sumber data berikut:

Nama dan Jenis Sumber Data	Indeks Bidang Default
<code>amazon_vpc.flow</code>	<code>action</code> <code>logStatus</code> <code>region</code> <code>flowDirection</code> <code>type</code>
<code>amazon_route53.resolver_query</code>	<code>query_type</code> <code>transport</code> <code>rcode</code>
<code>aws_waf.access</code>	<code>action</code> <code>httpRequest.country</code>
<code>aws_cloudtrail.data</code>	<code>eventSource</code>
<code>aws_cloudtrail.management</code>	<code>eventName</code> <code>awsRegion</code> <code>userAgent</code> <code>errorCode</code> <code>eventType</code>

Nama dan Jenis Sumber Data	Indeks Bidang Default
	<code>managementEvent</code>
	<code>readOnly</code>
	<code>eventCategory</code>
	<code>requestId</code>

Indeks bidang default merupakan tambahan dari indeks bidang kustom apa pun yang Anda tentukan dalam kebijakan Anda. Indeks bidang default tidak dihitung terhadap kuota [indeks bidang](#) Anda.

CloudWatch Log hanya mengindeks peristiwa log yang dicerna setelah kebijakan indeks dibuat. Itu tidak mengindeks peristiwa log yang dicerna sebelum kebijakan dibuat. Setelah Anda membuat indeks bidang, setiap peristiwa log yang cocok tetap diindeks selama 30 hari dari waktu konsumsi peristiwa log.

Note

Jika Anda membuat kebijakan indeks bidang di akun pemantauan, kebijakan tersebut tidak akan digunakan untuk grup log di akun sumber tertaut. Kebijakan indeks bidang hanya berlaku di akun tempat kebijakan itu dibuat.

Topik lainnya di bagian ini menjelaskan cara membuat indeks bidang. Untuk informasi tentang merujuk ke indeks bidang dalam kueri Anda, lihat [FilterIndex](#) dan [filter](#)

Topik

- [Sintaks indeks bidang dan kuota](#)
- [Membuat kebijakan indeks bidang tingkat akun](#)
- [Membuat kebijakan indeks bidang log-group](#)
- [Opsi pemilihan grup log saat membuat kueri](#)
- [Efek menghapus kebijakan indeks bidang](#)

Sintaks indeks bidang dan kuota

Anda membuat indeks bidang dengan membuat kebijakan indeks bidang. Anda dapat membuat kebijakan indeks tingkat akun yang berlaku untuk seluruh akun Anda, dan Anda juga dapat membuat kebijakan yang hanya berlaku untuk satu grup log. Untuk kebijakan indeks seluruh akun, Anda dapat memiliki kebijakan yang berlaku untuk semua grup log di akun. Anda juga dapat membuat kebijakan indeks tingkat akun yang berlaku untuk subset grup log di akun, yang dipilih berdasarkan awalan nama grup log mereka. Jika Anda memiliki beberapa kebijakan tingkat akun di akun yang sama, awalan nama grup log untuk kebijakan ini tidak dapat tumpang tindih. Demikian pula, Anda dapat membuat kebijakan indeks tingkat akun yang berlaku untuk nama sumber data tertentu dan kombinasi tipe. Hanya satu kebijakan akun yang dapat dibuat per nama sumber data dan kombinasi jenis.

Kebijakan indeks bidang tingkat grup log mengganti kebijakan indeks bidang tingkat akun: yang berlaku untuk grup log secara keseluruhan (seperti, kebijakan tingkat akun tanpa kriteria pemilihan atau dengan kriteria pemilihan berdasarkan awalan nama grup log). Kebijakan tingkat akun yang cocok di tingkat peristiwa log (seperti, untuk nama sumber data tertentu dan kombinasi jenis) akan berlaku selain kebijakan yang cocok dengan grup log secara keseluruhan. Jika Anda membuat kebijakan indeks tingkat log-grup, grup log tersebut tidak menggunakan kebijakan tingkat akun yang cocok di tingkat grup log.

Pencocokan peristiwa log dengan nama indeks bidang peka huruf besar/kecil. Misalnya, indeks bidang tidak `RequestId` akan cocok dengan peristiwa log yang berisirequestId.

Anda dapat memiliki sebanyak 40 kebijakan indeks tingkat akun, dari kebijakan ini 20 dapat menggunakan kriteria pemilihan awalan nama grup log dan 20 dapat menggunakan kriteria pemilihan berbasis sumber data. Jika Anda memiliki beberapa kebijakan indeks tingkat akun yang difilter untuk mencatat awalan nama grup, tidak ada dua dari mereka yang dapat menggunakan awalan nama grup log yang sama atau tumpang tindih. Misalnya, jika Anda memiliki satu kebijakan yang difilter untuk mencatat grup yang dimulainya-log, Anda tidak dapat memfilter kebijakan indeks bidang lain ke `my-logpprod` atau `my-logging`. Demikian pula, jika Anda memiliki beberapa kebijakan indeks tingkat akun yang difilter ke nama sumber data dan kombinasi tipe, tidak ada dua dari mereka yang dapat menggunakan nama dan jenis sumber data yang sama. Misalnya, jika Anda memiliki satu kebijakan yang difilter ke nama sumber data `amazon_vpc` dan tipe sumber data, `flow` Anda tidak dapat membuat kebijakan lain dengan kombinasi ini.

Jika Anda memiliki kebijakan indeks tingkat akun yang tidak memiliki awalan nama dan berlaku untuk semua grup log, maka tidak ada kebijakan indeks tingkat akun lain dengan filter awalan nama grup

log yang dapat dibuat; Anda dapat membuat kebijakan indeks tingkat akun yang menggunakan nama sumber data dan jenis filter.

Setiap kebijakan indeks memiliki kuota dan batasan berikut:

- Sebanyak 20 bidang dapat dimasukkan dalam kebijakan.
- Setiap nama bidang dapat mencakup sebanyak 100 karakter.
- Untuk membuat indeks bidang kustom di grup log Anda yang dimulai dengan@, Anda harus menentukan bidang dengan tambahan @ di awal nama bidang. Misalnya, jika peristiwa log Anda menyertakan bidang bernamauserId, Anda harus menentukan @@userId untuk membuat indeks untuk bidang ini.

Untuk kebijakan indeks tingkat akun dengan nama sumber data dan kriteria pemilihan berdasarkan tipe, pembatasan tambahan berlaku: semua bidang harus berupa tipe data primitif, primitif bersarang hanya didukung untuk struct.

Bidang yang dihasilkan dan bidang cadangan

CloudWatch Wawasan Log secara otomatis menghasilkan bidang sistem di setiap peristiwa log. Bidang yang dihasilkan ini diawali dengan @ Untuk informasi selengkapnya tentang bidang yang dihasilkan, lihat [Log yang didukung dan bidang yang ditemukan](#).

Dari bidang yang dihasilkan ini, berikut ini didukung untuk digunakan sebagai indeks bidang:

- @logStream
- @ingestionTime
- @requestId
- @type
- @initDuration
- @duration
- @billedDuration
- @memorySize
- @maxMemoryUsed
- @xrayTraceId
- @xraySegmentId

Untuk mengindeks bidang yang dihasilkan ini, Anda tidak perlu menambahkan tambahan `@` saat menentukannya, seperti yang harus Anda lakukan untuk bidang khusus yang dimulai `@`. Misalnya, untuk membuat indeks bidang untuk `@logStream`, cukup tentukan `@logStream` sebagai indeks bidang.

CloudWatch Log menyediakan indeks bidang default untuk semua grup log di kelas log Standar. Indeks bidang default secara otomatis tersedia untuk bidang berikut:

- `@logStream`
- `@aws.region`
- `@aws.account`
- `@source.log`
- `@data_source_name`
- `@data_source_type`
- `@data_format`
- `traceId`
- `severityText`
- `attributes.session.id`

CloudWatch Log menyediakan indeks bidang default untuk nama sumber data tertentu dan kombinasi tipe juga. Indeks bidang default secara otomatis tersedia untuk kombinasi nama dan jenis sumber data berikut:

Nama dan Jenis Sumber Data	Indeks Bidang Default
<code>amazon_vpc.flow</code>	<code>action</code> <code>logStatus</code> <code>region</code> <code>flowDirection</code> <code>type</code>
<code>amazon_route53_resolver_query</code>	<code>query_type</code>

Nama dan Jenis Sumber Data	Indeks Bidang Default
	transport rcode
aws_waf.access	action httpRequest.country
aws_cloudtrail.data	eventSource
aws_cloudtrail.management	eventName awsRegion userAgent errorCode eventType managementEvent readOnly eventCategory requestId

Indeks bidang default merupakan tambahan dari indeks bidang kustom apa pun yang Anda tentukan dalam kebijakan Anda. Indeks bidang default tidak dihitung terhadap kuota [indeks bidang](#) Anda.

Bidang anak dan bidang array di log JSON

Anda dapat mengindeks bidang yang merupakan bidang turunan bersarang atau bidang array di log JSON.

Misalnya, Anda dapat membuat indeks bidang `accessKeyId` anak dalam `userIdentity` bidang dalam log ini:

```
{
```

```

    "eventVersion": "1.0",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "EXAMPLE_PRINCIPAL_ID",
      "arn": "arn: aws: iam: : 123456789012: user/Alice",
      "accessKeyId": "11112222",
      "accountId": "123456789012",
      "userName": "Alice"
    },
    "eventTime": "2014-03-06T21: 22: 54Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.255",
    "userAgent": "ec2-api-tools1.6.12.2",
    "requestParameters": {
      "instancesSet": {
        "items": [{
          "instanceId": "i-abcde123",
          "currentState": {
            "code": 0,
            "name": "pending"
          },
          "previousState": {
            "code": 80,
            "name": "stopped"
          }
        }]
      }
    }
  }
}

```

Untuk membuat bidang ini, Anda merujuknya menggunakan notasi titik (`userIdentity.accessKeyId`) baik saat membuat indeks bidang dan saat menentukannya dalam kueri. Kueri bisa terlihat seperti ini:

```

fields @timestamp, @message
| filterIndex userIdentity.accessKeyId = "11112222"

```

Dalam contoh peristiwa sebelumnya, `instanceId` bidang berada dalam array dalam `requestParameters.instancesSet.items` Untuk mewakili bidang ini baik saat membuat indeks bidang dan saat query, lihat itu sebagai

`requestParameters.instancesSet.items.0.instanceId` 0 mengacu pada tempat bidang itu dalam array.

Maka kueri untuk bidang ini bisa menjadi sebagai berikut:

```
fields @timestamp, @message
| filterIndex requestParameters.instancesSet.items.0.instanceId="i-abcde123"
```

Membuat kebijakan indeks bidang tingkat akun

Gunakan langkah-langkah di bagian ini untuk membuat kebijakan indeks bidang yang berlaku untuk semua grup log di akun, atau ke beberapa grup log yang memiliki nama grup log yang dimulai dengan string yang sama.

Untuk membuat kebijakan indeks bidang tingkat akun

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Pengaturan dan kemudian pilih tab Log.
3. Di bagian Kebijakan indeks tingkat akun, pilih Kelola.
4. Pilih Buat kebijakan indeks.
5. Untuk nama Kebijakan, masukkan nama untuk kebijakan baru Anda.
6. Untuk Pilih lingkup kebijakan, lakukan salah satu hal berikut:
 - Pilih Semua grup log standar agar kebijakan indeks berlaku untuk semua grup log Kelas Standar di akun.
 - Pilih Pencocokan grup log menurut awalan untuk menerapkan kebijakan ke subset grup log yang semuanya memiliki nama yang dimulai dengan string yang sama. Kemudian, masukkan awalan untuk grup log ini di Masukkan nama awalan.

Setelah memasukkan awalan, Anda dapat memilih Pratinjau awalan grup log yang cocok untuk mengonfirmasi bahwa awalan Anda cocok dengan grup log yang Anda harapkan.

Pilih Log Data menurut Sumber data untuk menerapkan kebijakan ke kombinasi nama dan tipe sumber data tertentu. Anda kemudian dapat memilih Sumber data dan Jenis data dari menu drop-down.

Setelah Anda memilih nama dan jenis sumber data, Anda dapat memilih Dapatkan bidang untuk mengisi bagian Konfigurasi indeks bidang dan aspek dengan informasi yang

relevan seperti bidang yang tersedia, termasuk grup log, serta indeks bidang default dan kustom.

7. Untuk konfigurasi bidang indeks kustom, pilih Tambahkan jalur bidang untuk memasukkan bidang pertama yang akan diindeks.

Kemudian masukkan string untuk digunakan sebagai nilai nama bidang atau pilih bidang dari menu drop-down. Ini harus menjadi kasus yang sama persis dengan apa yang muncul dalam peristiwa log. Misalnya, jika peristiwa log Anda termasuk `requestId`, Anda harus masuk `requestId` di sini. `RequestId`, `requestID`, dan `request Id` tidak akan cocok.

Jika Anda ingin mengindeks bidang log kustom yang dimulai dengan `@` karakter, Anda harus menyertakan `@` karakter tambahan saat memasukkan string indeks. Misalnya, jika Anda memiliki bidang log kustom `@emailname`, masukkan `@@emailname` di kotak Tambah jalur bidang.

Anda juga dapat membuat indeks untuk `@ingestionTime` dan `@logStream` bidang yang dihasilkan CloudWatch Log secara otomatis. Jika ya, Anda tidak perlu menambahkan tambahan `@` saat menentukannya.

8. (Opsional) Selain menentukan jalur bidang, Anda dapat memilih Set sebagai faset untuk membuat bidang sebagai faset.
9. Ulangi langkah sebelumnya untuk menambahkan sebanyak 20 indeks bidang.
10. Setelah selesai, pilih Buat.

Membuat kebijakan indeks bidang log-group

Gunakan langkah-langkah di bagian ini untuk membuat kebijakan indeks bidang yang berlaku untuk satu grup log.

Untuk membuat kebijakan indeks bidang tingkat log-group

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Logs, Log groups (Grup log).
3. Pilih nama grup log.
4. Pilih tab Indeks bidang.
5. Pilih Kelola indeks bidang untuk grup log ini
6. Untuk Mengelola indeks bidang tingkat grup log, pilih Tambahkan jalur bidang untuk memasukkan bidang pertama yang akan diindeks.

Kemudian masukkan string untuk digunakan sebagai nilai nama bidang. Ini harus menjadi kasus yang sama persis dengan apa yang muncul dalam peristiwa log. Misalnya, jika peristiwa log Anda termasuk `requestId`, Anda harus masuk `requestId` di sini. `RequestId,requestID`, dan tidak `request Id` akan cocok.

Jika Anda ingin mengindeks bidang log kustom yang dimulai dengan `@` karakter, Anda harus menyertakan `@` karakter tambahan saat memasukkan string indeks. Misalnya, jika Anda memiliki bidang log `kustom@emailname`, masukkan `@emailname` di kotak Tambah jalur bidang.

Anda juga dapat membuat indeks untuk `@ingestionTime` dan `@logStream` bidang yang dihasilkan CloudWatch Log secara otomatis. Jika Anda melakukannya, tidak perlu menambahkan tambahan `@` saat menentukannya.

7. (Opsional) Selain menentukan jalur bidang, Anda dapat memilih Set sebagai faset untuk membuat bidang sebagai faset.
8. Ulangi langkah sebelumnya untuk menambahkan sebanyak 20 indeks bidang.
9. Setelah Anda selesai, pilih Simpan.

Opsi pemilihan grup log saat membuat kueri

Bagian ini menjelaskan berbagai cara Anda dapat memilih grup log untuk disertakan dalam kueri.

Untuk memilih grup log untuk kueri di konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Wawasan Log.
3. Pilih bahasa kueri yang ingin Anda gunakan untuk kueri ini. Anda dapat memilih: Log Wawasan QL, OpenSearch PPL, atau SQL. OpenSearch
4. Ada tiga cara untuk memilih grup log untuk kueri:
 - Gunakan kotak nama grup Log. Ini adalah metode pemilihan default. Anda dapat memasukkan sebanyak 50 nama grup log dengan metode ini. Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih grup log di akun sumber serta akun pemantauan. Satu kueri dapat menanyakan log dari akun yang berbeda sekaligus.
 - Gunakan bagian Kriteria grup Log. Di bagian ini, Anda dapat memilih grup log berdasarkan awalan nama grup log. Anda dapat menyertakan sebanyak lima awalan dalam satu kueri.

Grup log yang memiliki awalan ini dalam nama mereka akan dipilih. Atau, opsi Semua grup log memilih semua grup log dari akun.

- Jika ini adalah akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat memilih Semua akun di menu tarik-turun akun untuk memilih grup log dari semua akun yang ditautkan. Atau, Anda dapat secara individual memilih akun mana yang harus disertakan untuk kueri ini.

Jika pilihan Anda cocok dengan lebih dari 10.000 grup log, Anda akan melihat kesalahan yang meminta Anda untuk mempersempit pilihan Anda.

5. Kelas log default untuk kueri adalah Standar. Anda dapat menggunakan kelas Log untuk mengubahnya menjadi akses yang jarang.

Menggunakan AWS CLI

Untuk membuat jenis pilihan ini ketika Anda memulai kueri dari baris perintah, Anda dapat menggunakan `source` perintah dalam kueri Anda. Untuk informasi selengkapnya dan contoh tambahan, lihat [SUMBER](#).

Efek menghapus kebijakan indeks bidang

Jika Anda menghapus kebijakan indeks bidang yang telah berlaku selama beberapa waktu, hal berikut akan terjadi:

- Hingga 30 hari setelah kebijakan dihapus, kueri masih dapat memperoleh manfaat dari peristiwa log yang diindeks.
- Jika Anda menghapus kebijakan indeks tingkat log-grup, dan sudah ada kebijakan tingkat akun yang akan berlaku untuk grup log tersebut, kebijakan tingkat akun pada akhirnya akan berlaku untuk grup log tersebut.

Gunakan aspek untuk mengelompokkan dan menjelajahi log

Aspek berguna untuk menganalisis log karena memungkinkan Anda memfilter dan mengelompokkan data secara interaktif tanpa menjalankan kueri. Aspek adalah bidang di log Anda (seperti `ServiceName` atau `StatusCode`) yang memungkinkan penyaringan, agregasi, dan analisis di seluruh grup log. Anda dapat melihat daftar bidang faset di konsol Wawasan CloudWatch Log, bersama dengan jumlah peristiwa log untuk setiap nilai aspek berdasarkan rentang waktu yang Anda

pilih. Saat Anda memilih berbagai aspek dan nilai, nilai dan jumlah faset diperbarui secara real-time, memungkinkan Anda menjelajahi log secara interaktif.

Setiap aspek menunjukkan nilai dan jumlah yang tersedia secara otomatis diekstrak dari bidang di log Anda berdasarkan rentang waktu dan cakupan kueri yang dipilih, dan dipertahankan selama 30 hari. Jumlah faset yang ditampilkan adalah perkiraan. Anda dapat menggunakan aspek default seperti nama sumber data atau tipe sumber data untuk menjelajahi log Anda, atau membuat aspek khusus pada salah satu bidang di log Anda. Nama Sumber Data adalah layanan AWS atau aplikasi yang menghasilkan log (misalnya, Route 53, Amazon VPC, atau CloudTrail) dan tipe Sumber Data adalah jenis log spesifik yang dihasilkan oleh layanan tersebut. Aspek default dibuat oleh CloudWatch dan termasuk `@aws.region`, `@data_source_name@data_source_type`, dan `@data_format`. Untuk informasi selengkapnya, lihat [Manajemen log](#). Aspek hanya tersedia untuk log yang dicerna di akun. Jika Anda telah menyiapkan observabilitas lintas akun, akun pemantauan tidak dapat melihat aspek berdasarkan log dari akun sumber.

Untuk membuat aspek tambahan, pilih bidang di log Anda yang relevan dengan pemecahan masalah Anda dan konfigurasi menggunakan kebijakan indeks. Untuk aspek khusus, kami sarankan untuk membuatnya di bidang kardinalitas rendah (bidang yang memiliki kurang dari 100 nilai unik per hari seperti Status dan). ApplicationName Aspek dengan lebih dari 100 nilai unik per hari diklasifikasikan sebagai kardinalitas tinggi dan nilai untuk aspek-aspek ini tidak ditampilkan. Pilih satu atau beberapa aspek dan klik untuk menjalankan kueri di seluruh log Anda.

Untuk memulai dengan aspek di Wawasan CloudWatch Log:

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Wawasan Log.
3. (Opsional) Gunakan pemilih rentang waktu untuk memilih periode waktu yang ingin Anda analisis. Untuk rentang waktu yang dipilih, aspek dan nilai yang tersedia ditampilkan di panel.
4. Pilih aspek untuk menjelajahi data Anda dan melihat pembaruan real-time dari distribusi nilai di seluruh aspek.

Aspek dengan lebih dari 100 nilai unik tidak ditampilkan. Untuk menanyakan nilai tertentu, gunakan filter dalam kueri Anda sebagai gantinya.

Untuk menjalankan kueri berbasis facet

1. Pilih satu atau lebih nilai di seluruh aspek.

2. Jumlah acara akan diperbarui berdasarkan aspek dan nilai yang dipilih.
3. Saat nilai faset dipilih, cakupan kueri diperbarui untuk mencerminkan pemilihan.
4. Setelah memilih nilai faset, klik jalankan untuk menjalankan kueri Anda.
5. Jumlah maksimum nilai unik yang didukung per segi adalah 100. Misalnya, jika ada lebih dari 100 nilai untuk suatu segi, maka semua hitungan ditampilkan sebagai "-", yang menunjukkan bahwa nilainya tidak diketahui.

Untuk menyimpan kueri berbasis aspek

1. Buat kueri Anda menggunakan satu atau beberapa nilai facet.
2. Langkah-langkah lainnya sama dengan menyimpan kueri Wawasan Log. Lihat [kueri Menyimpan CloudWatch Log Wawasan](#).
3. Kueri tersimpan tersedia di bagian Kueri Tersimpan. Saat Anda mengambil kueri yang disimpan, itu akan secara otomatis menyertakan aspek dan nilai yang digunakan untuk kueri, sehingga mudah untuk menganalisis log Anda.

Untuk membuat aspek tingkat akun

1. Untuk membuat faset, Anda harus terlebih dahulu membuat bidang sebagai indeks dan mengkonfigurasinya sebagai faset. Di panel navigasi, pilih Pengaturan, Log, Kebijakan indeks tingkat akun. Atau, Anda dapat memilih Kelola aspek pada panel faset.
2. Pilih Buat kebijakan indeks baru. Untuk detail tentang membuat kebijakan indeks, lihat [Membuat kebijakan indeks bidang tingkat akun](#).
3. Untuk membuat facet, centang Set as facet untuk bidang yang dipilih di halaman pembuatan kebijakan indeks.

Manajemen Facet menggunakan APIs

Pengelolaan facet dapat dilakukan dengan menggunakan kebijakan indeks bidang. Lihat [field index](#) APIs untuk detail.

Indeks Lapangan APIs

Tidak.	Nama	Deskripsi
1	PutIndexPolicy	Membuat atau memperbarui kebijakan indeks bidang untuk grup log tertentu
2	PutAccountPolicy	Membuat kebijakan perlindungan data tingkat akun, kebijakan filter langganan, kebijakan indeks bidang, kebijakan transformator, atau kebijakan ekstraksi metrik yang berlaku untuk semua grup log atau subset grup log di akun
3	DeleteIndexPolicy	Menghapus kebijakan indeks bidang log-group yang diterapkan ke grup log tunggal
4	DeleteAccountPolicy	Menghapus kebijakan akun CloudWatch Log

Melihat log sekitarnya di Wawasan CloudWatch Log

Anda dapat melihat peristiwa log yang terjadi sebelum dan sesudah catatan log tertentu yang ditampilkan oleh kueri Wawasan CloudWatch Log Anda. Log sekitarnya membantu Anda memahami konteks seputar kesalahan, peringatan, atau peristiwa penting lainnya.

Cara kerja log di sekitarnya

Saat Anda menjalankan kueri Wawasan CloudWatch Log, setiap catatan log dalam hasil menyertakan opsi Log Sekitar. Saat Anda memilih opsi ini, Wawasan CloudWatch Log menampilkan peristiwa log tambahan dari aliran log yang sama yang terjadi segera sebelum dan sesudah rekaman yang dipilih. Peristiwa ini muncul meskipun tidak cocok dengan filter kueri asli Anda.

Anda dapat mengonfigurasi jumlah baris log di sekitarnya untuk ditampilkan. Pilih dari 5, 10, 20, 50, atau 100 baris log di atas dan di bawah catatan yang dipilih.

Log sekitarnya berguna dalam skenario berikut:

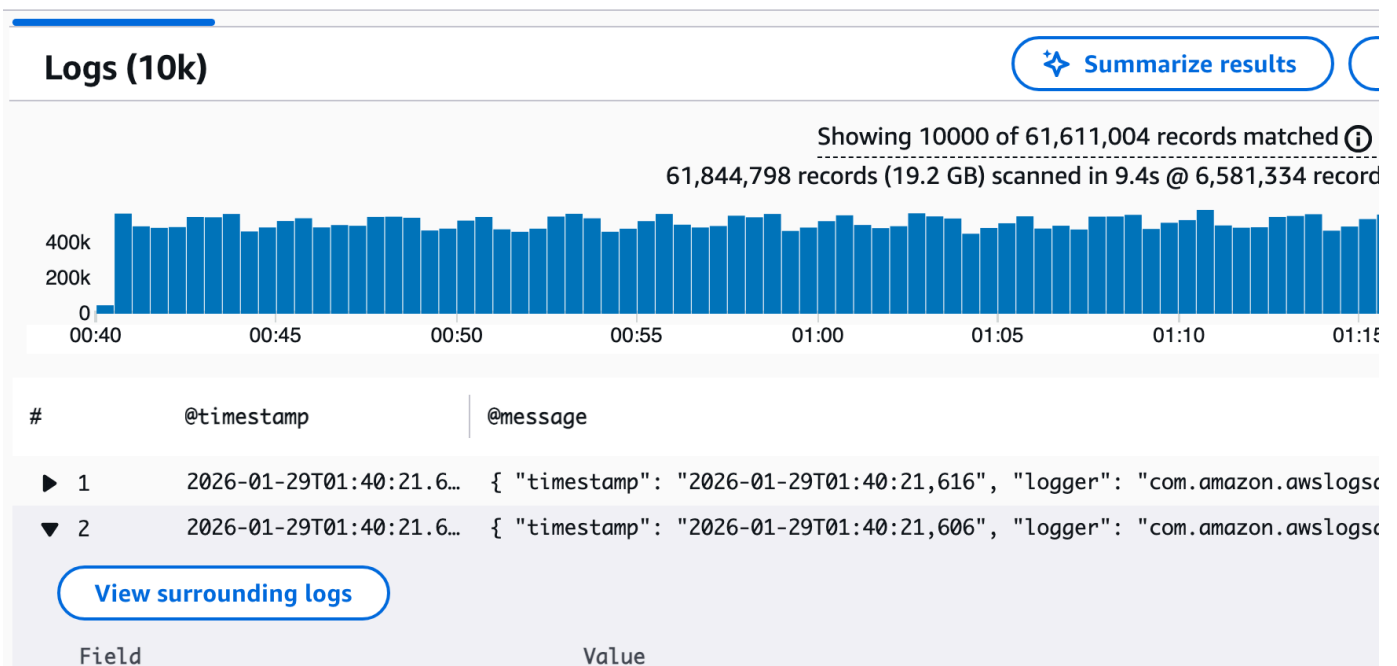
- Kesalahan debugging dengan melihat peristiwa yang menyebabkan kegagalan
- Menyelidiki peringatan atau batas waktu dalam sistem terdistribusi

- Memahami urutan peristiwa di seluruh aplikasi Anda
- Menganalisis masalah di lingkungan log volume tinggi

Lihat log di sekitarnya

Untuk melihat log di sekitarnya

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Wawasan Log.
3. Jalankan kueri Anda.
4. Dalam hasil kueri, cari catatan log yang ingin Anda selidiki.
5. Pilih log Sekitar.



6. Dalam menu tarik-turun Jumlah peristiwa, pilih jumlah peristiwa log yang akan ditampilkan di atas dan di bawah catatan yang dipilih. Anda dapat memilih 5, 10, 20, 50, atau 100.

Surrounding Logs [ApplicationLogs-ip-10-0-42-156.ec2.internalApplicationLogs](#) ✕

Q Search log messages... Log event count +/- 5 ▲

Timestamp	Message
▶ 2026-01-29T01:40:21.246Z	{ "timestamp": "2026-01-29T01:40:21,246", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.246Z	{ "timestamp": "2026-01-29T01:40:21,246", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.577Z	{ "timestamp": "2026-01-29T01:40:21,577", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.587Z	{ "timestamp": "2026-01-29T01:40:21,587", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.597Z	{ "timestamp": "2026-01-29T01:40:21,597", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ » 2026-01-29T01:40:21.607Z	{ "timestamp": "2026-01-29T01:40:21,606", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.616Z	{ "timestamp": "2026-01-29T01:40:21,616", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.625Z	{ "timestamp": "2026-01-29T01:40:21,625", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.635Z	{ "timestamp": "2026-01-29T01:40:21,635", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.644Z	{ "timestamp": "2026-01-29T01:40:21,644", "logger": "com.amazon.awslogsanalysisdataplane.util..." }
▶ 2026-01-29T01:40:21.653Z	{ "timestamp": "2026-01-29T01:40:21,653", "logger": "com.amazon.awslogsanalysisdataplane.util..." }

There are newer logs to load. [Load more](#)



Cari di dalam log sekitarnya

Setelah Anda membuka panel log di sekitarnya, Anda dapat mencari kata kunci tertentu untuk menemukan konteks yang relevan.

Untuk mencari di dalam log sekitarnya

1. Di panel log sekitarnya, masukkan istilah pencarian Anda di kotak pencarian.
2. Tinjau garis log pencocokan yang disorot.
3. Gunakan kontrol navigasi untuk berpindah antar pertandingan.

Surrounding Logs [ApplicationLogs-ip-10-0-42-156.ec2.internalApplicationLogs](#) ✕

Q 33750 ✕ Log event count +/- 5 ▼

Timestamp	Message
▶ 2026-01-29T01:40:21.246Z	{ "timestamp": "2026-01-29T01:40:21,246", "logger": "com.amazon.awslogsanalysisdataplane.patt...
▼ 2026-01-29T01:40:21.246Z	{ "timestamp": "2026-01-29T01:40:21,246", "logger": "com.amazon.awslogsanalysisdataplane.patt...

```
{
  "timestamp": "2026-01-29T01:40:21,246",
  "logger": "com.amazon.awslogsanalysisdataplane.patternlibrary.PatternLibraryStoreImpl",
  "level": "INFO",
  "threadID": "33750",
  "threadName": "sdk-async-response-4-18426",
  "message": "Calling putPattern with anomalyDetectorId '80a6b9d7-daa0-44ac-bef7-3ef8662e4347'"
}
```

Analisis pola

CloudWatch Logs Insights menggunakan algoritme pembelajaran mesin untuk menemukan pola saat Anda menanyakan log Anda. Pola adalah struktur teks bersama yang berulang di antara bidang log Anda. Saat melihat hasil kueri, Anda dapat memilih tab Pola untuk melihat pola yang ditemukan CloudWatch Log berdasarkan sampel hasil Anda. Atau, Anda dapat menambahkan pattern perintah ke kueri Anda untuk menganalisis pola di seluruh rangkaian peristiwa log yang cocok.

Pola berguna untuk menganalisis kumpulan log besar karena sejumlah besar peristiwa log sering dapat dikompresi menjadi beberapa pola.

Pertimbangkan contoh berikut dari tiga peristiwa log.

```
2023-01-01 19:00:01 [INFO] Calling DynamoDB to store for resource id 12342342k124-12345
2023-01-01 19:00:02 [INFO] Calling DynamoDB to store for resource id 324892398123-12345
2023-01-01 19:00:03 [INFO] Calling DynamoDB to store for resource id 3ff231242342-12345
```

Dalam sampel sebelumnya, ketiga peristiwa log mengikuti satu pola:

```
<Time-1> [INFO] Calling DynamoDB to store for resource id <ID-2>
```

Bidang dalam pola disebut token. Bidang yang bervariasi dalam suatu pola, seperti ID permintaan atau stempel waktu, adalah token dinamis. Setiap token dinamis diwakili oleh `<string-number>`. `string` Ini adalah deskripsi dari jenis data yang diwakili oleh token. `number` Menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

Contoh umum token dinamis termasuk kode kesalahan, stempel waktu, dan permintaan. IDs Nilai token mewakili nilai tertentu dari token dinamis. Misalnya, jika token dinamis mewakili kode kesalahan HTTP, maka nilai token bisa jadi 501.

Deteksi pola juga digunakan dalam detektor anomali CloudWatch Log dan fitur perbandingan. Untuk informasi selengkapnya, lihat [Deteksi anomali log](#) dan [Bandingkan \(diff\) dengan rentang waktu sebelumnya](#).

Memulai dengan analisis pola

Deteksi pola dilakukan secara otomatis dalam kueri Wawasan CloudWatch Log apa pun. Kueri yang tidak menyertakan `pattern` perintah mendapatkan peristiwa log dan pola dalam hasil.

Jika Anda menyertakan `pattern` perintah dalam kueri Anda, analisis pola dilakukan pada seluruh rangkaian peristiwa log yang cocok. Ini memberi Anda hasil pola yang lebih akurat, tetapi peristiwa log mentah tidak dikembalikan saat Anda menggunakan `pattern` perintah. Ketika kueri tidak disertakan `pattern`, hasil pola didasarkan pada 1000 peristiwa log pertama yang dikembalikan, atau pada nilai batas yang Anda gunakan dalam kueri Anda. Jika Anda menyertakan `pattern` dalam kueri, maka hasil yang ditampilkan di tab Pola berasal dari semua peristiwa log yang cocok dengan kueri.

Untuk memulai analisis pola di Wawasan CloudWatch Log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Wawasan Log.

Pada halaman Wawasan Log, editor kueri berisi kueri default yang menampilkan 20 peristiwa log terbaru.

3. Hapus `| limit 20` baris di kotak kueri, sehingga kueri terlihat seperti berikut:

```
fields @timestamp, @message, @logStream, @log
| sort @timestamp desc
```

4. Di menu drop-down Pilih grup log, pilih satu atau beberapa grup log untuk kueri.
5. (Opsional) Gunakan pemilih interval waktu untuk memilih periode waktu yang ingin Anda kueri.

Anda dapat memilih antara interval 5 menit dan 30 menit; Interval 1 jam, 3 jam, dan 12 jam; atau kerangka waktu khusus.

6. Pilih Jalankan kueri untuk memulai kueri.

Saat kueri selesai berjalan, tab Log menampilkan tabel peristiwa log yang dikembalikan oleh kueri. Di atas tabel adalah pesan tentang berapa banyak catatan yang cocok dengan kueri, mirip dengan Menampilkan 10.000 dari 71.101 catatan yang cocok.

7. Pilih tab Patterns.
8. Tabel sekarang menampilkan pola yang ditemukan dalam kueri. Karena kueri tidak menyertakan pattern perintah, tab ini hanya menampilkan pola yang ditemukan di antara 10.000 peristiwa log yang ditampilkan dalam tabel di tab Log.

Untuk setiap pola, informasi berikut ditampilkan:

- Pola, dengan setiap token dinamis ditampilkan sebagai `<string-number>`. *string* Ini adalah deskripsi dari jenis data yang diwakili oleh token. *number* Menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.
- Hitungan Peristiwa, yang merupakan berapa kali pola muncul dalam peristiwa log yang ditanyakan. Pilih judul kolom Event count untuk mengurutkan pola berdasarkan frekuensi.
- Rasio peristiwa, yang merupakan persentase dari peristiwa log kueri yang berisi pola ini.
- Jenis Keparahan, yang akan menjadi salah satu dari yang berikut:
 - ERROR jika pola berisi kata Error.
 - PERINGATKAN jika pola berisi kata Warn tetapi tidak mengandung Error.
 - INFO jika pola tidak mengandung Warn atau Error.

Pilih judul kolom Info keparahan untuk mengurutkan pola berdasarkan tingkat keparahan.

9. Sekarang ubah kueri. Ganti `| sort @timestamp desc` baris dalam kueri dengan `| pattern @message`, sehingga kueri lengkapnya adalah sebagai berikut:

```
fields @timestamp, @message, @logStream, @log
| pattern @message
```

10. Pilih Run query (Jalankan kueri).

Saat kueri selesai, tidak ada hasil di tab Log. Namun, tab Patterns kemungkinan memiliki jumlah pola yang lebih besar yang terdaftar, tergantung pada jumlah total peristiwa log yang ditanyakan.

11. Terlepas dari apakah Anda termasuk `pattern` dalam kueri Anda, Anda dapat memeriksa lebih lanjut pola yang dikembalikan kueri. Untuk melakukannya, pilih ikon di kolom Inspect untuk salah satu pola.

Panel pemeriksaan Pola muncul dan menampilkan yang berikut:

- Pola. Pilih token dalam pola untuk menganalisis nilai token tersebut.
- Histogram yang menunjukkan jumlah kemunculan pola selama rentang waktu yang ditanyakan. Ini dapat membantu Anda mengidentifikasi tren menarik seperti peningkatan tiba-tiba dalam terjadinya suatu pola.
- Tab Log samples menampilkan beberapa peristiwa log yang cocok dengan pola yang dipilih.
- Tab Nilai Token menampilkan nilai token dinamis yang dipilih, jika Anda telah memilihnya.

 Note

Maksimal 10 nilai token ditangkap untuk setiap token. Jumlah token mungkin tidak tepat. CloudWatch Log menggunakan penghitung probabilistik untuk menghasilkan jumlah token, bukan nilai absolut.

- Tab pola Terkait menampilkan pola lain yang sering terjadi mendekati waktu yang sama dengan pola yang Anda periksa. Misalnya, jika pola untuk ERROR pesan biasanya disertai dengan peristiwa log lain yang ditandai INFO dengan detail tambahan, pola itu ditampilkan di sini.

Detail tentang perintah pola

Bagian ini berisi rincian lebih lanjut tentang pattern perintah dan penggunaannya.

- Dalam tutorial sebelumnya, kami menghapus `sort` perintah ketika kami menambahkan `pattern` karena kueri tidak valid jika menyertakan `pattern` perintah setelah `sort` perintah. Adalah sah untuk memiliki `pattern` sebelum `asort`.

Untuk detail selengkapnya tentang `pattern` sintaks, lihat [pola](#).

- Bila Anda menggunakan `pattern` dalam query, `@message` harus menjadi salah satu bidang yang dipilih dalam `pattern` perintah.
- Anda dapat menyertakan `filter` perintah sebelum `pattern` perintah untuk menyebabkan hanya kumpulan peristiwa log yang difilter untuk digunakan sebagai masukan untuk analisis pola.
- Untuk melihat hasil pola untuk bidang tertentu, seperti bidang yang berasal dari `parse` perintah, gunakan `pattern @fieldname`.

- Kueri dengan output non-log, seperti kueri dengan `stats` perintah, tidak mengembalikan hasil pola.

Simpan dan jalankan kembali kueri CloudWatch Logs Insights

Setelah Anda membuat kueri, Anda dapat menyimpannya, dan menjalankannya lagi nanti. Kueri disimpan dalam struktur folder, sehingga Anda dapat mengaturnya. Anda dapat menyimpan sebanyak 1000 kueri per wilayah dan per akun.

Kueri disimpan pada tingkat khusus Wilayah, bukan tingkat khusus pengguna. Jika Anda membuat dan menyimpan kueri, pengguna lain yang memiliki akses ke CloudWatch Log di Wilayah yang sama dapat melihat semua kueri yang disimpan dan struktur foldernya di Wilayah.

Untuk menyimpan kueri, Anda harus masuk ke peran yang memiliki `logs:PutQueryDefinition`. Untuk melihat daftar kueri yang disimpan, Anda harus masuk ke peran yang memiliki `logs:DescribeQueryDefinitions`.

Note

Anda dapat membuat dan menyimpan kueri dengan parameter — template yang dapat digunakan kembali dengan placeholder bernama. Alih-alih menyimpan beberapa variasi kueri yang sama dengan nilai yang berbeda, buat satu templat dan berikan nilai parameter yang berbeda saat Anda menjalankannya. Fungsionalitas ini saat ini didukung untuk kueri hanya menggunakan bahasa kueri Wawasan Log. Untuk informasi selengkapnya, lihat [Menggunakan kueri tersimpan dengan parameter](#).

Console

Untuk menyimpan kueri

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Buat kueri di editor kueri.
4. Pilih Simpan.
5. Masukkan nama untuk kueri.

6. (Opsional) Pilih folder tempat Anda ingin menyimpan kueri. Pilih Create new (Buat baru) untuk membuat folder. Jika Anda membuat folder baru, Anda dapat menggunakan karakter garis miring (/) dalam nama folder untuk menentukan struktur folder. Sebagai contoh, menamai folder baru dengan **folder-level-1/folder-level-2** akan membuat folder tingkat atas yang disebut **folder-level-1**, dengan folder lain yang bernama **folder-level-2** di dalam folder itu. Kueri disimpan dalam **folder-level-2**.
7. (Opsional) Ubah grup log kueri atau teks kueri.
8. (Opsional) Untuk menggunakan parameter dalam kueri Anda, ikuti langkah-langkah tambahan ini:
 - a. Tambahkan parameter ke kueri Anda. Ganti nilai statis dengan placeholder menggunakan `{{parameter}}` sintaks (tanda kurung ganda sebelum dan sesudah nama parameter).

Contoh: Kueri asli dengan nilai statis:

```
fields @timestamp, @message
| filter level = "Error"
| filter applicationName = "OrderService"
```

Kueri diperbarui dengan parameter:

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

- b. Tentukan parameter yang digunakan dalam kueri Anda. Untuk setiap parameter placeholder, tentukan:
 - Nama: Harus sama persis dengan nama placeholder (misalnya, `logLevel`, `applicationName`).
 - Nilai default (opsional): Nilai yang akan digunakan jika tidak ada nilai parameter yang disediakan.
 - Deskripsi (opsional): Menjelaskan tujuan parameter.
 - c. Query dengan parameter dapat dijalankan dengan menggunakan nama query dengan \$ awalan dan meneruskan nama parameter sebagai pasangan kunci-nilai. Lihat Untuk menjalankan kueri tersimpan untuk detailnya.

9. Pilih Simpan.

AWS CLI

Untuk menyimpan kueri, gunakan `put-query-definition`:

```
aws logs put-query-definition \
  --name "ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = \"ERROR\"" \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

(Opsional) Untuk menyimpan kueri dengan parameter, tambahkan `--parameters` opsi dan gunakan `{{parameterName}}` placeholder dalam string kueri:

```
aws logs put-query-definition \
  --name "ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = {{logLevel}} | filter applicationName = {{applicationName}}" \
  --parameters '[{"name": "logLevel", "defaultValue": "ERROR", "description": "Log level to filter"}, {"name": "applicationName", "defaultValue": "OrderService", "description": "Application name to filter"}]' \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

Untuk menyimpan kueri di folder, awali nama kueri dengan jalur folder:

```
aws logs put-query-definition \
  --name "my-folder/ErrorsByLevel" \
  --query-string "fields @timestamp, @message | filter level = {{logLevel}}" \
  --parameters '[{"name": "logLevel", "defaultValue": "ERROR", "description": "Log level to filter"}]' \
  --log-group-names "/aws/lambda/my-function" \
  --region us-east-1
```

API

Untuk menyimpan kueri, hubungi [PutQueryDefinition](#):

```
{
  "name": "ErrorsByLevel",
  "queryString": "fields @timestamp, @message | filter level = \"ERROR\"",
  "logGroupNames": ["/aws/lambda/my-function"]
}
```

```
}
```

(Opsional) Untuk menyimpan kueri dengan parameter, sertakan parameters bidang dan gunakan `{{parameterName}}` placeholder dalam string kueri:

```
{
  "name": "ErrorsByLevel",
  "queryString": "fields @timestamp, @message | filter level = {{logLevel}} | filter
applicationName = {{applicationName}}",
  "logGroupNames": ["/aws/lambda/my-function"],
  "parameters": [
    {
      "name": "logLevel",
      "defaultValue": "ERROR",
      "description": "Log level to filter"
    },
    {
      "name": "applicationName",
      "defaultValue": "OrderService",
      "description": "Application name to filter"
    }
  ]
}
```

Tip

Anda dapat membuat folder untuk kueri yang disimpan dengan `PutQueryDefinition`. Untuk membuat folder untuk kueri yang disimpan, gunakan garis miring (/) untuk mengawali nama kueri yang Anda inginkan dengan nama folder yang Anda inginkan: `<folder-name>/<query-name>` Untuk informasi lebih lanjut tentang tindakan ini, lihat [PutQueryDefinition](#).

Console

Untuk menjalankan kueri yang disimpan

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Di sebelah kanan, pilih Queries (Kueri).

4. Pilih kueri Anda dari daftar Kueri tersimpan. Teks kueri muncul di editor kueri.
5. (Opsional) Untuk menggunakan kueri dengan parameter:
 - a. Pilih ikon+di sebelah nama kueri di panel samping Kueri tersimpan.
 - b. Kueri dengan parameter muncul di editor kueri. Misalnya, jika Anda memilih ikon + di sebelahErrorsByLevel, editor kueri diisi dengan: `$ErrorsByLevel(level=, applicationName=)`
 - c. Berikan nilai untuk parameter (level, applicationName) dan jalankan kueri. Misalnya: `$ErrorsByLevel(level= "ERROR", applicationName= "OrderService")`
6. Pilih Jalankan.

AWS CLI

Untuk menjalankan kueri tersimpan dengan parameter

Gunakan `start-query` dengan `$QueryName()` sintaks:

```
aws logs start-query \  
  --log-group-names "/aws/lambda/my-function" \  
  --start-time 1707566400 --end-time 1707570000 \  
  --query-string '$ErrorsByLevel(level= "ERROR", applicationName= "OrderService")' \  
  --region us-east-1
```

API

Untuk menjalankan kueri tersimpan dengan parameter

Panggil [StartQuery](#) dengan `$QueryName()` sintaks di `queryString` lapangan:

```
{  
  "logGroupNames": ["/aws/lambda/my-function"],  
  "startTime": 1707566400,  
  "endTime": 1707570000,  
  "queryString": "$ErrorsByLevel(level=\"ERROR\", applicationName= \"OrderService  
  \")\"  
}
```


Untuk menyimpan versi baru dari kueri tersimpan

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Di sebelah kanan, pilih Queries (Kueri).
4. Pilih kueri dari daftar Saved queries (Kueri tersimpan). Itu akan muncul di editor kueri.
5. Modifikasi kueri. Jika Anda perlu menjalankannya untuk memeriksa pekerjaan Anda, pilih Run query (Jalankan kueri).
6. Saat Anda siap untuk menyimpan versi baru, pilih Actions (Tindakan), Save as (Simpan sebagai).
7. Masukkan nama untuk kueri.
8. (Opsional) Pilih folder tempat Anda ingin menyimpan kueri. Pilih Create new (Buat baru) untuk membuat folder. Jika Anda membuat folder baru, Anda dapat menggunakan karakter garis miring (/) dalam nama folder untuk menentukan struktur folder. Sebagai contoh, menamai folder baru dengan **folder-level-1/folder-level-2** akan membuat folder tingkat atas yang disebut **folder-level-1**, dengan folder lain yang bernama **folder-level-2** di dalam folder itu. Kueri disimpan dalam **folder-level-2**.
9. (Opsional) Ubah grup log kueri atau teks kueri.
10. Pilih Simpan.

Untuk menghapus kueri, Anda harus masuk ke peran yang memiliki izin `logs:DeleteQueryDefinition`.

Untuk mengedit atau menghapus kueri tersimpan

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Di sebelah kanan, pilih Queries (Kueri).
4. Pilih kueri dari daftar Saved queries (Kueri tersimpan). Itu akan muncul di editor kueri.
5. Pilih Actions (Tindakan), Edit atau Actions (Tindakan), Delete (Hapus).

Menggunakan kueri yang disimpan dengan parameter

Kueri tersimpan dengan parameter adalah templat kueri yang dapat digunakan kembali dengan placeholder bernama. Alih-alih mempertahankan beberapa salinan kueri yang hampir identik, Anda dapat menyimpan template dan memberikan nilai parameter yang berbeda saat menjalankan kueri. Parameter hanya didukung dalam bahasa kueri Wawasan CloudWatch Log.

Cara kerjanya

Saat menyimpan kueri, placeholder mengidentifikasi nilai yang dapat Anda berikan pada waktu eksekusi kueri. Placeholder menggunakan sintaks. `{{parameterName}}` Berikut ini adalah contoh dari query tersimpan bernama `ErrorsByLevel` dengan dua parameter `logLevel` dan `applicationName`.

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

Untuk menjalankan kueri tersimpan, Anda dapat memanggilnya menggunakan nama kueri yang diawali dengan `$` dan meneruskan nilai parameter. Mesin kueri CloudWatch Logs Insights menggantikan setiap placeholder. Jika parameter berisi nilai default, maka nilai-nilai tersebut digunakan jika tidak ada nilai lain yang disediakan.

```
# Run query by using query name and passing parameter values explicitly
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService")

# Run query without specifying parameter values - default values are used in this case.
$ErrorsByLevel()
```

Nama kueri tersimpan yang berisi spasi atau karakter khusus harus diapit dengan backticks:

```
$`Errors By Level`(logLevel = "WARN")
```

Contoh kueri yang disimpan dengan parameter

Menambahkan batas hasil sebagai parameter

Nama kueri: `ErrorsByLevel` dengan parameter `logLevel` (default:"ERROR"), `applicationName` (default:"OrderService"), dan `maxResults` (default:50)

```
fields @timestamp, @message, @logStream
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
| sort @timestamp desc
| limit {{maxResults}}
```

```
# Run the query using the query name and passing parameter values
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService", maxResults = 100)
```

Menggunakan beberapa kueri tersimpan dengan parameter

Contoh di bawah ini menggunakan `ErrorsByLevel` dan kueri tersimpan kedua `RecentN` yang didefinisikan sebagai `sort @timestamp desc | limit {{count}}` (dengan parameter `count`, default `20`). Mesin kueri CloudWatch Logs Insights memperluas setiap kueri sebelum menjalankannya.

```
# Using multiple queries with parameters in sequence
$ErrorsByLevel(logLevel = "WARN", applicationName = "OrderService")
| $RecentN(count = 10)

# Each of the queries is expanded, resulting in the following query when it is run.
fields @timestamp, @message
| filter level = "WARN"
| filter applicationName = "OrderService"
| sort @timestamp desc
| limit 10
```

Kuota dan penanganan kesalahan

Note

Setiap kueri yang disimpan dapat memiliki maksimum 20 parameter.

String kueri yang diperluas tidak dapat melebihi 10.000 karakter. Nama parameter harus dimulai dengan huruf atau garis bawah. Kueri tersimpan tidak dapat mereferensikan kueri lain yang disimpan (pemanggilan bersarang tidak didukung).

Kesalahan umum

Kesalahan	Penyebab
Parameter hanya didukung untuk bahasa kueri CWLI	Parameter hanya didukung dalam bahasa kueri Wawasan CloudWatch Log.
Parameter yang diperlukan tidak ditemukan di QueryString	Nama parameter di <code>--parameters</code> tidak memiliki pencocokan <code>{{placeholder}}</code> dalam string kueri.
Jumlah parameter melebihi maksimum 20	Kueri yang disimpan saat ini hanya mendukung 20 parameter.
Nama parameter duplikat	Definisi kueri memiliki parameter duplikat di <code>parameters</code> .

 Note

Untuk membuat atau memperbarui kueri yang disimpan dengan parameter, Anda memerlukan `logs:PutQueryDefinition` izin. Untuk menjalankannya, Anda perlu `logs:StartQuery` dan `logs:DescribeQueryDefinitions`.

Tambahkan kueri ke dasbor atau ekspor hasil kueri

Setelah menjalankan kueri, Anda dapat menambahkan kueri ke CloudWatch dasbor atau menyalin hasilnya ke clipboard.

Kueri yang ditambahkan ke dasbor akan dijalankan setiap kali Anda memuat dasbor dan setiap kali dasbor disegarkan. Kueri ini dihitung terhadap batas 100 kueri Wawasan CloudWatch Log bersamaan.

Untuk menambahkan hasil kueri ke dasbor

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Pilih satu atau beberapa grup log dan jalankan kueri.
4. Pilih Tambahkan ke dasbor.
5. Pilih dasbor, atau pilih Create new (Buat baru) untuk membuat dasbor untuk hasil kueri.

6. Pilih jenis widget yang akan digunakan untuk hasil kueri.
7. Masukkan nama untuk widget.
8. Pilih Tambahkan ke dasbor.

Untuk menyalin hasil kueri ke clipboard atau mengunduh hasil kueri

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Pilih satu atau beberapa grup log dan jalankan kueri.
4. Pilih Export results (Ekspor hasil), lalu pilih opsi yang Anda inginkan.

Lihat kueri atau riwayat kueri yang sedang berjalan

Anda dapat melihat kueri yang sedang berlangsung serta riwayat kueri terbaru Anda.

Kueri yang sedang berjalan mencakup kueri yang telah ditambahkan ke dasbor. Anda dibatasi hingga 100 kueri Wawasan CloudWatch Log bersamaan per akun, termasuk kueri yang ditambahkan ke dasbor. Selain itu, Anda dapat menjalankan 15 kueri bersamaan untuk OpenSearch Service PPL atau OpenSearch Service SQL.

Untuk melihat riwayat kueri terbaru Anda

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, lalu pilih Wawasan Log.
3. Pilih Riwayat, jika Anda menggunakan desain baru untuk konsol CloudWatch Log. Jika Anda menggunakan desain lama, pilih Actions (Tindakan), View query history for this account (Lihat riwayat kueri untuk akun ini).

Daftar kueri terbaru Anda akan muncul. Anda dapat menjalankan kembali salah satunya dengan memilih kueri dan memilih Run (Jalankan).

Di bawah Status, CloudWatch Log ditampilkan Sedang berlangsung untuk kueri apa pun yang sedang berjalan.

Enkripsi hasil kueri dengan AWS Key Management Service

Secara default, CloudWatch Log mengenkripsi hasil tersimpan dari kueri Wawasan CloudWatch Log. Anda menggunakan metode enkripsi sisi server CloudWatch Log default. Anda dapat memilih untuk menggunakan AWS KMS kunci untuk mengenkripsi hasil ini sebagai gantinya. Jika Anda mengaitkan AWS KMS kunci dengan hasil enkripsi Anda, maka CloudWatch Log menggunakan kunci tersebut untuk mengenkripsi hasil yang disimpan dari semua kueri di akun.

Jika nanti Anda memisahkan kunci dari hasil kueri, CloudWatch Log akan kembali ke metode enkripsi default untuk kueri selanjutnya. Tetapi kueri yang berjalan saat kunci dikaitkan masih dienkripsi dengan kunci itu. CloudWatch Log masih dapat mengembalikan hasil tersebut setelah kunci KMS dipisahkan, karena CloudWatch Log masih dapat terus mereferensikan kunci. Namun, jika kunci kemudian dinonaktifkan, maka CloudWatch Log tidak dapat membaca hasil kueri yang dienkripsi dengan kunci itu.

Important

CloudWatch Log hanya mendukung kunci KMS simetris. Jangan gunakan kunci asimetris untuk mengenkripsi hasil kueri Anda. Untuk informasi selengkapnya, lihat [Menggunakan Kunci Simetris dan Asimetris](#).

Batas

- Untuk melakukan langkah-langkah berikut, Anda harus memiliki izin berikut: `kms:CreateKey`, `kms:GetKeyPolicy`, dan `kms:PutKeyPolicy`.
- Setelah Anda mengaitkan atau memisahkan kunci dari hasil kueri Anda, diperlukan waktu hingga lima menit agar operasi diterapkan.
- Jika Anda mencabut akses CloudWatch Log ke kunci terkait atau menghapus kunci KMS terkait, data terenkripsi Anda di CloudWatch Log tidak dapat diambil lagi.
- Anda tidak dapat menggunakan CloudWatch konsol untuk mengaitkan kunci, Anda harus menggunakan AWS CLI atau CloudWatch Logs API.

Langkah 1: Buat AWS KMS key

Untuk membuat kunci KMS gunakan perintah [create-key](#) berikut:

aws kms create-key

Output berisi ID kunci dan Amazon Resource Name (ARN) dari kunci. Berikut ini adalah output contoh:

```
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1478910250.94,
    "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-e40cb0d29f59",
    "AWSAccountId": "123456789012",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

Langkah 2: Tetapkan izin pada tombol KMS

Secara default, semua kunci KMS bersifat pribadi. Hanya pemilik sumber daya yang dapat menggunakannya untuk mengenkripsi dan mendekripsi data. Namun, pemilik sumber daya dapat memberikan izin untuk mengakses kunci ke pengguna dan sumber daya lain. Dengan langkah ini, Anda memberikan izin utama layanan CloudWatch Log untuk menggunakan kunci. Prinsipal layanan ini harus berada di AWS Wilayah yang sama di mana kunci disimpan.

Sebagai praktik terbaik, kami menyarankan Anda membatasi penggunaan kunci hanya untuk AWS akun yang Anda tentukan.

Pertama, simpan kebijakan default untuk kunci KMS Anda seperti `policy.json` menggunakan [get-key-policy](#) perintah berikut:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./policy.json
```

Buka file `policy.json` di editor teks dan tambahkan bagian dalam huruf tebal dari salah satu pernyataan berikut. Pisahkan pernyataan yang ada dari pernyataan baru dengan koma. Pernyataan ini menggunakan Condition bagian untuk meningkatkan keamanan AWS KMS kunci. Untuk informasi selengkapnya, lihat [AWS KMS kunci dan konteks enkripsi](#).

ConditionBagian dalam contoh ini membatasi penggunaan AWS KMS kunci untuk hasil kueri Wawasan CloudWatch Log di akun yang ditentukan.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.region.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt*",
        "kms:Decrypt*",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
```



```

    "aws:SourceArn": "arn:aws:logs:us-east-1:111122223333:query-
result:*"
  },
  "StringEquals": {
    "aws:SourceAccount": "111122223333"
  }
}
]
}

```

Terakhir, tambahkan kebijakan yang diperbarui menggunakan [put-key-policy](#) perintah berikut:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Langkah 3: Kaitkan kunci KMS dengan hasil kueri Anda

Untuk mengaitkan kunci KMS dengan hasil kueri di akun

Gunakan [disassociate-kms-key](#) perintah sebagai berikut:

```
aws logs associate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-
result:*" --kms-key-id "key-arn"
```

Langkah 4: Lepaskan kunci dari hasil kueri di akun

Untuk memisahkan kunci KMS yang terkait dengan hasil kueri, gunakan perintah berikut:

[disassociate-kms-key](#)

```
aws logs disassociate-kms-key --resource-identifier "arn:aws:logs:region:account-
id:query-result:*"
```

Hasilkan ringkasan bahasa alami dari hasil kueri CloudWatch Log Insights

Menganalisis data log sangat penting untuk memahami perilaku aplikasi Anda, tetapi menafsirkan volume besar entri log dapat memakan waktu. CloudWatch Logs Insights sekarang menawarkan

kemampuan meringkas bahasa alami yang mengubah hasil kueri yang kompleks menjadi ringkasan yang jelas dan ringkas. Kemampuan ini membantu Anda mengidentifikasi masalah dengan cepat dan mendapatkan wawasan yang dapat ditindaklanjuti dari data log Anda.

Cara kerjanya

CloudWatch Wawasan Log dapat menghasilkan ringkasan yang dapat dibaca manusia dari hasil kueri menggunakan Amazon Bedrock. Fitur ini mendukung semua bahasa kueri Wawasan CloudWatch Log dan memberikan wawasan yang jelas dan dapat ditindaklanjuti dari data log Anda.

Ketersediaan regional dan pemrosesan data

Important

Saat Anda menggunakan fitur ini, hasil kueri Anda mungkin akan diproses dengan cara yang berbeda Wilayah AWS. Misalnya, jika Anda menjalankan kueri di US East (Virginia N.), ringkasan mungkin terjadi di AS Barat (Oregon).

Tabel berikut mencantumkan kemungkinan pemrosesan Wilayah AWS untuk geografi yang berbeda di mana fitur hasil kueri tersedia:

Geografi CloudWatch Log yang Didukung	Wilayah Pemrosesan yang Mungkin
Amerika Serikat (US)	Wilayah AS Timur (N. Virginia)
	Wilayah AS Timur (Ohio)
	Wilayah AS Barat (Oregon)
Eropa	Wilayah Eropa (Frankfurt)
	Wilayah Eropa (Irlandia)
	Wilayah Eropa (Paris)
	Wilayah Eropa (Stockholm)
	Wilayah Eropa (London)

Geografi CloudWatch Log yang Didukung	Wilayah Pemrosesan yang Mungkin
Asia Pasifik	Wilayah AS Timur (N. Virginia)
	Wilayah AS Timur (Ohio)
	Wilayah AS Barat (Oregon)
Amerika Selatan	Wilayah AS Timur (N. Virginia)
	Wilayah AS Timur (Ohio)
	Wilayah AS Barat (Oregon)

Memulai

Untuk menghasilkan ringkasan bahasa alami

1. Jalankan kueri Wawasan CloudWatch Log Anda.
2. Setelah kueri selesai, pilih Ringkas hasil.

Izin

Anda harus memiliki salah satu dari yang berikut:

- Izin `CloudWatchLogsFullAccess`
- Izin `CloudWatchLogsReadOnlyAccess`
- Kebijakan IAM khusus
termasuk `cloudwatch:GenerateQueryResultsSummary`, `logs:GetQueryResults`,
`logs:DescribeQueries` dan tindakan `logs:FilterLogEvents`

Privasi data

Hasil kueri Anda diproses dengan aman dan tidak digunakan untuk melatih atau meningkatkan Wawasan CloudWatch Log atau Amazon Bedrock. Jika Anda memilih untuk memberikan umpan balik pada ringkasan hasil kueri menggunakan tombol umpan balik, umpan balik Anda menunjukkan tingkat kepuasan Anda dengan kemampuan yang disediakan dalam Wawasan CloudWatch Log.

Mengotomatiskan analisis log dengan kueri terjadwal

Kueri terjadwal memungkinkan Anda mengotomatiskan eksekusi kueri Wawasan CloudWatch Log pada jadwal reguler. Alih-alih menjalankan kueri secara manual untuk menganalisis data log Anda, Anda dapat mengonfigurasi kueri terjadwal agar berjalan secara otomatis dan memberikan hasil ke tujuan seperti bucket Amazon S3 atau bus acara Amazon. EventBridge Otomatisasi ini sangat ideal untuk menghasilkan laporan rutin, memantau tren, atau memicu proses hilir berdasarkan hasil analisis log.

Kueri terjadwal mendukung ketiga bahasa kueri yang tersedia di Wawasan CloudWatch Log:

- [Bahasa kueri Wawasan Log \(Wawasan Log QL\)](#)
- [OpenSearch Service Piped Processing Language \(PPL\)](#)
- [OpenSearch Layanan Structured Query Language \(SQL\)](#)

Daftar Isi

- [Memahami konsep kueri terjadwal](#)
- [Referensi ekspresi jadwal](#)
- [Praktik terbaik](#)
- [Memulai dengan kueri terjadwal](#)
- [Mengonfigurasi tujuan S3 untuk kueri terjadwal](#)
- [Memecahkan masalah kueri terjadwal](#)

Memahami konsep kueri terjadwal

Sebelum membuat kueri terjadwal, pahami konsep-konsep kunci ini yang memengaruhi cara kueri Anda berjalan dan di mana hasil dikirimkan.

Pemisahan peran IAM

Kueri terjadwal memerlukan dua peran IAM terpisah: satu untuk mengeksekusi kueri dan satu lagi untuk memberikan hasil ke bus acara Amazon S3 atau Amazon. EventBridge Memahami mengapa pemisahan ini ada membantu Anda mengonfigurasi izin dengan benar dan menggunakan manfaat keamanan dan operasional yang diberikannya.

Arsitektur dua peran membagi tanggung jawab antara akses data dan pengiriman data. Peran eksekusi kueri mengakses data log Anda dan menjalankan kueri, sementara peran pengiriman tujuan menulis hasil ke tujuan yang Anda pilih. Pemisahan ini mengikuti prinsip hak istimewa terkecil—setiap peran hanya memiliki izin yang dibutuhkan untuk fungsi spesifiknya.

Peran eksekusi kueri

Memungkinkan CloudWatch CloudWatch Log menjalankan kueri Wawasan Log atas nama Anda. Peran ini memerlukan izin untuk mengakses grup log Anda dan menjalankan kueri, tetapi tidak memerlukan akses ke sumber daya tujuan. Izin yang diperlukan:

- `logs:StartQuery`
- `logs:StopQuery`
- `logs:GetQueryResults`
- `logs:DescribeLogGroups`
- `logs:UnmaskIfDataIsOpen` jika data terbuka diperlukan

Untuk grup log terenkripsi KMS: **`kms:Decrypt`** dan **`kms:DescribeKey`** izin untuk kunci KMS yang digunakan untuk mengenkripsi grup log. Izin ini perlu ditambahkan juga.

Persyaratan hubungan kepercayaan: Peran eksekusi kueri harus menyertakan kebijakan kepercayaan yang memungkinkan layanan CloudWatch Log (`logs.amazonaws.com`) untuk mengambil peran tersebut. Tanpa hubungan kepercayaan ini, kueri terjadwal akan gagal dengan kesalahan izin.

Contoh kebijakan kepercayaan untuk peran eksekusi kueri:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Contoh kebijakan izin untuk peran eksekusi kueri:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:StartQuery",
        "logs:StopQuery",
        "logs:GetQueryResults",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Peran pengiriman tujuan

Memungkinkan CloudWatch Log untuk mengirimkan hasil kueri ke tujuan yang Anda pilih. Peran ini hanya memerlukan izin untuk layanan tujuan tertentu, mengikuti prinsip hak istimewa paling sedikit. Izin yang diperlukan bervariasi menurut jenis tujuan.

Persyaratan hubungan kepercayaan: Peran pengiriman tujuan juga harus menyertakan kebijakan kepercayaan yang memungkinkan layanan CloudWatch Log (`logs.amazonaws.com`) untuk mengambil peran tersebut.

Contoh kebijakan izin untuk peran pengiriman tujuan S3:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::your-scheduled-query-results-bucket/*"
    }
  ]
}
```

Pemisahan ini memberikan manfaat praktis untuk operasi Anda. Dari perspektif keamanan, jika Anda perlu mengubah tempat hasil dikirimkan, Anda hanya mengubah peran pengiriman tujuan tanpa mengubah izin eksekusi kueri. Untuk kepatuhan dan audit, Anda dapat dengan jelas melacak peran mana yang mengakses data log sensitif dan peran mana yang ditulis ke sistem eksternal. Ini memudahkan untuk menunjukkan bahwa infrastruktur analisis log Anda mengikuti praktik terbaik keamanan.

Penggunaan lintas wilayah dan lintas akun

Kueri terjadwal dibuat di wilayah tertentu dan berjalan di wilayah tersebut. Namun, Anda dapat menanyakan grup log dan memberikan hasil di seluruh wilayah dan akun. Anda perlu menyiapkan satu atau beberapa AWS akun sebagai akun pemantauan dan menautkannya dengan beberapa akun sumber. Akun pemantauan adalah AWS akun pusat yang dapat melihat dan berinteraksi dengan data observabilitas yang dihasilkan dari akun sumber. Akun sumber adalah AWS akun individu yang menghasilkan data observabilitas untuk sumber daya yang berada di dalamnya. Akun sumber membagikan data observabilitas mereka dengan akun pemantauan. Jadi, Anda dapat mengatur kueri terjadwal dari akun pemantauan menggunakan grup log dari semua akun yang ditautkan.

Menanyakan grup log lintas wilayah

Kueri terjadwal Anda dapat mengakses grup log di wilayah mana pun. Tentukan grup log menggunakan format ARN lengkapnya: `arn:aws:logs:region:account-id:log-group:log-group-name`. Kebutuhan peran eksekusi kueri `logs:StartQuery` dan `logs:GetQueryResults` izin untuk grup log di semua wilayah target.

Important

Saat menanyakan grup log atau memberikan hasil di seluruh wilayah, data log melintasi batas regional. Pertimbangkan hal berikut:

- Persyaratan residensi data - Pastikan transfer data lintas wilayah sesuai dengan kebijakan tata kelola data organisasi Anda dan persyaratan peraturan
- Biaya transfer data - Transfer data lintas wilayah menimbulkan biaya tambahan
- Latensi jaringan - Kueri yang mengakses grup log di wilayah yang jauh mungkin mengalami latensi yang lebih tinggi

Untuk kinerja optimal dan efisiensi biaya, buat kueri terjadwal di wilayah yang sama dengan grup log utama Anda.

Pendekatan alternatif: Gunakan [sentralisasi CloudWatch Log](#) untuk mereplikasi data log dari beberapa akun dan wilayah menjadi akun pemantauan pusat. Ini memungkinkan Anda membuat kueri terjadwal di satu wilayah yang mengakses semua log terpusat Anda, menghindari kueri lintas wilayah, dan menyederhanakan pengelolaan izin IAM.

Menjadwalkan ekspresi dan penanganan zona waktu

Jadwal yang Anda tentukan menentukan kapan kueri Anda berjalan dan seberapa sering dijalankan. Memilih ekspresi jadwal yang tepat memengaruhi saat Anda menerima hasil dan berapa banyak data yang Anda kueri. Memahami jenis ekspresi membantu Anda memilih antara kesederhanaan dan presisi.

Ekspresi cron memberikan kontrol yang tepat atas waktu, memungkinkan Anda menentukan waktu yang tepat, hari dalam seminggu, atau hari dalam sebulan. Gunakan ekspresi cron saat Anda membutuhkan kueri untuk dijalankan pada jam kerja tertentu atau sejajar dengan jadwal operasional. Di konsol Anda juga dapat menjadwalkan kueri menggunakan opsi kalender mudah.

Ekspresi Cron

Jalankan kueri pada waktu tertentu. Format: `cron(minute hour day-of-month month day-of-week year)`. Contoh:

- `cron(0 9 * * ? *)`- Setiap hari pukul 09.00 UTC
- `cron(0 18 ? * MON-FRI *)`- Hari kerja pukul 6:00 PM UTC
- `cron(0 0 1 * ? *)`- Hari pertama setiap bulan pada tengah malam UTC
- `cron(0 12 ? * SUN *)`- Setiap Minggu siang UTC
- `cron(30 8 1 1 ? *)`- 1 Januari pukul 8:30 UTC

Semua kueri terjadwal berjalan di UTC, terlepas dari zona waktu lokal Anda atau di mana AWS sumber daya Anda berada. Ini sangat penting ketika Anda menjadwalkan kueri untuk jam kerja atau analisis sensitif waktu. Misalnya, jika bisnis Anda beroperasi di Waktu Timur AS dan Anda menginginkan laporan harian pada pukul 9 pagi ET, Anda perlu memperhitungkan offset UTC (14:00

UTC selama waktu musim panas, 13:00 UTC sebaliknya). Rencanakan ekspresi jadwal Anda dengan mempertimbangkan UTC untuk memastikan kueri berjalan pada waktu yang ditentukan.

Memilih bahasa kueri

Kueri terjadwal mendukung tiga bahasa kueri yang berbeda, dan pilihan Anda memengaruhi cara Anda menulis kueri dan seberapa mudah tim Anda dapat mempertahankannya. Bahasa yang tepat tergantung pada persyaratan analisis Anda dan keterampilan tim Anda yang ada.

Jika Anda terutama memfilter dan menggabungkan data log, CloudWatch Logs Insights Query Language menawarkan sintaks yang paling mudah. Untuk transformasi data yang kompleks di mana Anda perlu membentuk kembali atau memperkaya data melalui beberapa langkah, pendekatan pipeline PPL membuat logika lebih mudah diikuti. Ketika Anda perlu melakukan gabungan atau agregasi kompleks yang mirip dengan operasi database, SQL menyediakan sintaks akrab yang dapat diadopsi oleh tim berpengalaman database dengan cepat.

CloudWatch Bahasa Kueri Wawasan Log (CWLI)

Dibuat khusus untuk analisis log dengan sintaks intuitif. Terbaik untuk:

- Analisis dan penyaringan log berbasis teks
- Agregasi dan statistik deret waktu
- Tim baru untuk analisis log

OpenSearch Service Piped Processing Language (PPL)

Bahasa kueri berbasis pipeline dengan kemampuan transformasi data yang kuat. Terbaik untuk:

- Transformasi dan pengayaan data yang kompleks
- Alur kerja pemrosesan data multi-langkah
- Tim yang akrab dengan pemrosesan berbasis pipa

OpenSearch Layanan Structured Query Language (SQL)

Sintaks SQL standar untuk kueri gaya database yang sudah dikenal. Terbaik untuk:

- Kompleks bergabung dan agregasi
- Intelijen bisnis dan pelaporan
- Tim dengan pengalaman SQL yang kuat

Pemilihan tujuan dan kasus penggunaan

Di mana Anda mengirim hasil kueri menentukan apa yang dapat Anda lakukan dengan mereka. Pilihan ini membentuk seluruh alur kerja hilir Anda—apakah Anda sedang membangun analitik jangka panjang, memicu respons otomatis, atau keduanya. Memahami kekuatan setiap jenis tujuan membantu Anda merancang arsitektur yang tepat untuk kasus penggunaan Anda.

Tujuan Amazon S3 dioptimalkan untuk penyimpanan dan pemrosesan batch. Saat Anda perlu menyimpan hasil kueri selama berbulan-bulan atau bertahun-tahun, menganalisis tren dari waktu ke waktu, atau memasukkan data ke dalam platform analitik, Amazon S3 menyediakan penyimpanan hemat biaya dengan retensi tak terbatas. EventBridge tujuan dioptimalkan untuk otomatisasi waktu nyata. Ketika hasil kueri harus memicu tindakan segera—seperti mengirim peringatan, memulai alur kerja, atau memperbarui sistem—EventBridge memberikan hasil sebagai peristiwa yang dapat ditanggapi oleh aplikasi Anda secara instan. Secara default, semua peristiwa penyelesaian kueri secara otomatis dikirim sebagai peristiwa ke bus peristiwa default, memungkinkan integrasi dengan sistem pemrosesan hilir, fungsi Lambda, atau arsitektur berbasis peristiwa lainnya. Hasil hanya dipublikasikan ke tujuan ketika kueri dijalankan dengan sukses.

Tujuan Amazon S3

Simpan hasil kueri sebagai file JSON untuk retensi jangka panjang dan pemrosesan batch.

Terbaik untuk:

- Analisis historis dan pengarsipan data
- Integrasi dengan data lake dan platform analitik
- Persyaratan kepatuhan dan audit
- Penyimpanan hemat biaya dari set hasil besar

EventBridge destinasi

Kirim hasil kueri sebagai acara untuk pemrosesan dan otomatisasi waktu nyata. Anda dapat mengambil hasil kueri menggunakan QueryID yang dikirim dalam acara hingga 30 hari hanya karena kami menyimpan hasil selama 30 hari. Terbaik untuk:

- Memicu respons otomatis terhadap hasil kueri
- Integrasi dengan alur kerja tanpa server dan fungsi Lambda
- Sistem peringatan dan notifikasi waktu nyata
- Arsitektur dan layanan mikro yang digerakkan oleh acara

Format dan struktur hasil kueri

Untuk tujuan Amazon S3 - Hasil kueri dikirimkan dalam format JSON dengan struktur yang sama dengan respons API. GetQueryResults Untuk Amazon, EventBridge memahami format hasil kueri terjadwal membantu Anda merancang alur kerja pemrosesan dan integrasi hilir.

Hasil kueri disampaikan dalam format JSON dengan struktur berikut:

```
{
  "version": "0",
  "id": "be72061b-eca2-e068-a7e1-83e01d6fe807",
  "detail-type": "Scheduled Query Completed",
  "source": "aws.logs",
  "account": "123456789012",
  "time": "2025-11-18T11:31:48Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:logs:us-east-1:123456789012:scheduled-query:477b4380-b098-474e-9c5e-e10a8cc2e6e7"
  ],
  "detail": {
    "queryId": "2038fd57-ab4f-4018-bb2f-61d363f4a004",
    "queryString": "fields @timestamp, @message, @logStream, @log\n| sort\n@timestamp desc\n| limit 10000",
    "logGroupIdentifiers": [
      "/aws/lambda/my-function"
    ],
    "status": "Complete",
    "startTime": 1763465460,
    "statistics": {
      "recordsMatched": 0,
      "recordsScanned": 0,
      "estimatedRecordsSkipped": 0,
      "bytesScanned": 0,
      "estimatedBytesSkipped": 0,
      "logGroupsScanned": 1
    }
  }
}
```

Elemen kunci meliputi:

- **statistics**- Metrik kinerja kueri termasuk catatan yang cocok, dipindai, byte diproses, dan diperkirakan data yang dilewati
- **startTime**- Saat eksekusi kueri dimulai (stempel waktu Unix)
- **queryString**- Kueri aktual yang dieksekusi
- **queryId**- ID kueri dari kueri yang menggunakan hasil mana yang dapat diambil
- **logGroupIdentifiers**- Daftar grup log yang ditanyakan
- **status**- Status eksekusi kueri (Lengkap, Gagal, dll.)

Referensi ekspresi jadwal

Gunakan tabel referensi ini untuk membuat ekspresi jadwal untuk kueri terjadwal Anda. Format waktunya adalah UTC.

Sintaks ekspresi cron

Format: `cron(minute hour day-of-month month day-of-week year)`

Kasus Penggunaan	Ekspresi Cron	Deskripsi	Gunakan Kapan
Jadwal Harian	<code>cron(0 9 * * ? *)</code>	Setiap hari pukul 09.00 UTC	Laporan harian
	<code>cron(0 */6 * * ? *)</code>	Setiap 6 jam (00:00, 06:00, 12:00, 18:00 UTC)	Pemantauan yang sering
	<code>cron(30 2 * * ? *)</code>	Setiap hari pukul 02:30 UTC	Analisis di luar puncak
Jam Kerja	<code>cron(0 9-17 ? * MON-FRI *)</code>	Setiap jam dari jam 9 pagi sampai jam 5 sore, Senin sampai Jumat UTC	Pemantauan bisnis
	<code>cron(0 18 ? * MON-FRI *)</code>	Hari kerja pukul 6:00 PM UTC	Akhir hari kerja

Kasus Penggunaan	Ekspresi Cron	Deskripsi	Gunakan Kapan
	<code>cron(0 8,12,17 ? * MON-FRI *)</code>	8 pagi, siang, dan 5 sore pada hari kerja UTC	Waktu bisnis utama
Jadwal Mingguan	<code>cron(0 12 ? * SUN *)</code>	Setiap hari Minggu di siang hari UTC	Ringkasan mingguan
	<code>cron(0 9 ? * MON *)</code>	Setiap Senin pukul 9:00 UTC	Laporan awal minggu
	<code>cron(0 23 ? * FRI *)</code>	Setiap Jumat pukul 11:00 WIB UTC	Pembersihan akhir minggu
Jadwal Bulanan	<code>cron(0 0 1 * ? *)</code>	Hari pertama setiap bulan di tengah malam UTC	Laporan bulanan
	<code>cron(0 9 L * ? *)</code>	Hari terakhir setiap bulan pukul 9:00 UTC	Pemrosesan akhir bulan
	<code>cron(0 10 1 1,4,7,10 ? *)</code>	Hari pertama setiap kuartal pukul 10:00 UTC	Analisis triwulanan
Frekuensi Tinggi	<code>cron(* /15 * * * ? *)</code>	Setiap 15 menit	Pemantauan waktu nyata
	<code>cron(0,30 * * * ? *)</code>	Setiap 30 menit (pukul 00 dan:30)	Pemeriksaan yang sering
	<code>cron(0 */2 * * ? *)</code>	Setiap 2 jam	Interval reguler

Kasus Penggunaan	Ekspresi Cron	Deskripsi	Gunakan Kapan
Kasus Khusus	<code>cron(30 8 1 1 ? *)</code>	1 Januari pukul 8:30 UTC	Laporan tahunan
	<code>cron(0 6 * * SAT,SUN *)</code>	Akhir pekan pukul 6:00 UTC	Pemrosesan akhir pekan
	<code>cron(0 0 ? * MON#1 *)</code>	Senin pertama setiap bulan di tengah malam UTC	Perencanaan bulanan

Referensi bidang ekspresi cron

Bidang	Nilai	Wildcard	Contoh
Menit (1)	0-59	<code>*</code> , <code>-</code> , <code>/</code>	0(atas jam), <code>*/15</code> (setiap 15 menit), <code>0, 30</code> (dua kali per jam)
Jam (2)	0-23	<code>*</code> , <code>-</code> , <code>/</code>	9(9 AM), <code>*/2</code> (setiap 2 jam), 9-17 (jam kerja)
Day-of-month (3)	1-31, L, W	<code>*</code> , <code>-</code> , <code>/</code> , <code>?</code>	1(Hari pertama), L (hari terakhir), <code>?</code> (saat menggunakan day-of-week)
Bulan (4)	1-12 atau JAN-DES	<code>*</code> , <code>-</code> , <code>/</code>	1(Januari), JAN, 1, 4, 7, 10 (triwulanan)
Day-of-week (5)	1-7 atau MGG-SBT	<code>*</code> , <code>-</code> , <code>/</code> , <code>?</code> , <code>#</code> , L	MON-FRI(hari kerja), SUN, MON#1 (Senin 1)
Tahun (6)	1970-2199	<code>*</code> , <code>-</code> , <code>/</code>	<code>*</code> (setiap tahun), 2024 (tahun tertentu), 2024-2026 (kisaran)

Karakter wildcard dan ekspresi khusus

* (tanda bintang)

Cocokkan semua nilai di lapangan. Contoh: * di bidang jam berarti setiap jam.

?(tanda tanya)

Tidak ada nilai spesifik. Gunakan di day-of-month atau day-of-week ketika yang lain ditentukan.

Contoh: Gunakan ? day-of-month saat menentukan MON-FRI di day-of-week.

- (dasbor)

Rentang nilai. Contoh: MON-FRI (Senin sampai Jumat), 9-17 (9 pagi sampai jam 5 sore).

,(koma)

Beberapa nilai spesifik. Contoh: MON, WED, FRI (Senin, Rabu, Jumat), 8, 12, 17 (8 pagi, siang, 5 sore).

/ (tebasan)

Nilai langkah atau kenaikan. Contoh: 0/15 dalam menit berarti setiap 15 menit dimulai pada menit 0 (0, 15, 30, 45). */2 dalam jam berarti setiap 2 jam.

L (terakhir)

Hari terakhir bulan atau kejadian terakhir hari kerja. Contoh: L day-of-month artinya hari terakhir dalam sebulan. FRIL berarti Jumat terakhir bulan.

W (hari kerja)

Hari kerja terdekat. Contoh: 15W berarti hari kerja terdekat hingga tanggal 15 setiap bulan.

(kejadian ke-n)

Ke-n terjadinya hari kerja dalam sebulan. Contoh: MON#1 berarti Senin pertama setiap bulan, FRI#2 berarti Jumat kedua setiap bulan.

Pola umum dan praktik terbaik

- Untuk aplikasi bisnis: Gunakan MON-FRI dan jam kerja (misalnya, 9-17) untuk menghindari menjalankan kueri selama akhir pekan atau di luar jam kerja.
- Untuk pemantauan frekuensi tinggi: Gunakan kenaikan seperti */15 (setiap 15 menit) tetapi perhatikan batas konkurensi kueri.

- Untuk efisiensi sumber daya: Jadwalkan kueri intensif sumber daya selama jam-jam di luar sibuk menggunakan waktu pagi seperti UTC. 2–6
- Untuk laporan bulanan: Gunakan L untuk hari terakhir bulan atau tanggal tertentu seperti 1 untuk hari pertama untuk memastikan waktu yang konsisten.

Praktik terbaik

Ikuti praktik terbaik ini untuk memastikan operasi kueri terjadwal yang andal dan efisien:

Pengoptimalan kueri

- Uji kueri secara manual sebelum penjadwalan untuk memverifikasi kinerja dan hasil
- Gunakan indeks filter di awal kueri Anda untuk mengurangi pemrosesan data
- Batasi rentang waktu untuk menghindari batas waktu dengan grup log volume tinggi
- Pertimbangkan kompleksitas kueri dan batas waktu eksekusi

Jadwal perencanaan

- Hindari eksekusi yang tumpang tindih dengan memastikan kueri selesai sebelum jadwal berjalan berikutnya
- Pertimbangkan penundaan konsumsi log saat mengatur rentang waktu
- Gunakan ekspresi cron untuk waktu tertentu
- Sebarkan jadwal untuk memastikan Anda tidak mencapai batas konkurensi kueri

Pemantauan dan pemeliharaan

- Memantau riwayat eksekusi secara teratur untuk mengidentifikasi kegagalan atau masalah kinerja
- Meninjau dan memperbarui peran IAM secara berkala untuk menjaga keamanan
- Uji aksesibilitas tujuan sebelum menerapkan ke produksi

Otorisasi

- Semua kueri terjadwal APIs untuk mengotorisasi sumber daya kueri terjadwal dan bukan pada sumber daya yang diperlukan dalam input seperti grup log. Siapkan kebijakan IAM yang sesuai
- Mengelola otorisasi grup log menggunakan peran eksekusi yang diteruskan di APIs

Memulai dengan kueri terjadwal

Saat membuat kueri terjadwal, Anda akan mengonfigurasi beberapa komponen utama yang menentukan cara kueri Anda berjalan dan di mana hasil dikirim. Memahami komponen-komponen ini akan membantu Anda mengatur analisis log otomatis yang efektif.

Setiap kueri terjadwal terdiri dari komponen kunci berikut:

Konfigurasi kueri

String kueri CloudWatch Logs Insights, grup log target, dan bahasa kueri yang akan digunakan untuk analisis.

Ekspresi jadwal

Ekspresi cron atau kalender frekuensi yang menentukan kapan kueri berjalan. Anda dapat menentukan pengaturan zona waktu untuk memastikan kueri berjalan pada waktu lokal yang benar. Konsol menampilkan deskripsi jadwal Anda yang dapat dibaca manusia, seperti “Jalankan kueri setiap hari Selasa pukul 15:10 untuk rentang waktu 5 menit, efektif segera, di UTC, hingga tanpa batas waktu.”

Rentang waktu

Periode lookback untuk setiap eksekusi kueri, ditentukan oleh waktu mulai offset dari waktu eksekusi. Ini menentukan berapa banyak data historis yang akan dianalisis setiap eksekusi kueri.

Pratinjau jadwal eksekusi

Konsol menunjukkan tiga kueri terjadwal berikutnya berjalan dengan tanggal dan waktu yang tepat (misalnya, 2025/10/28 15:10, UTC; 2025/11/04 15:10, UTC; 2025/11/11 15:10, UTC), membantu Anda memverifikasi bahwa jadwal dikonfigurasi dengan benar.

Tujuan

Di mana hasil kueri dikirim setelah eksekusi berhasil. Tujuan yang didukung termasuk bucket Amazon S3 dan secara default metadata hasil dikirim ke bus acara default.

Peran eksekusi

Peran IAM yang diasumsikan CloudWatch Log untuk mengeksekusi kueri dan memberikan hasil ke tujuan yang ditentukan.

Sebelum membuat kueri terjadwal, pastikan Anda memiliki izin dan sumber daya yang diperlukan yang dikonfigurasi.

Membuat kueri terjadwal

Buat kueri terjadwal yang secara otomatis menjalankan kueri Wawasan CloudWatch Log dan memberikan hasil ke tujuan yang Anda pilih.

Prasyarat

Sebelum membuat kueri terjadwal, pastikan Anda memiliki yang berikut:

- Grup log - Satu atau beberapa grup log yang berisi data yang ingin Anda analisis
- Eksekusi peran IAM - Peran IAM dengan izin berikut:
 - `logs:StartQuery`- Izin untuk memulai CloudWatch kueri Wawasan Log
 - `logs:GetQueryResults`- Izin untuk mengambil hasil kueri
 - `logs:DescribeLogGroups`- Izin untuk mengakses informasi grup log. Ini hanya diperlukan untuk grup log berbasis awalan untuk penemuan grup log
- Izin tujuan - Izin IAM tambahan untuk tujuan yang Anda pilih:
 - Untuk tujuan Amazon S3: `s3:PutObject`
- Untuk AWS CLI dan penggunaan API - AWS Kredensi yang dikonfigurasi dengan izin untuk memanggil Log CloudWatch APIs

Untuk contoh kebijakan IAM yang mendetail, lihat [Manajemen identitas dan akses untuk Amazon CloudWatch Logs](#). Juga perlu dicatat Anda hanya dapat memiliki 1000 kueri terjadwal per akun.

Console

Untuk membuat kueri terjadwal (konsol)

1. Buka konsol CloudWatch Log di <https://us-east-1.console.aws.amazon.com/cloudwatch/rumah?region=us-east-1> # logsV2: logs-wawasan.
2. Di panel navigasi, pilih Wawasan Logs.
3. Pilih Buat kueri terjadwal.
4. Di bagian Definisi kueri:
 - a. Untuk bahasa Kueri, pilih bahasa kueri yang akan digunakan dari daftar.
 - b. Untuk string Kueri, masukkan kueri Wawasan CloudWatch Log Anda di kotak.
 - c. Untuk grup Log, pilih grup log untuk kueri dari daftar.

5. Di bagian Pengaturan jadwal:
 - a. Untuk ekspresi Jadwal, konfigurasi saat kueri berjalan. Pilih dari opsi yang telah ditentukan atau masukkan ekspresi cron kustom.
 - b. Untuk Efektif saat pembuatan, tentukan kapan jadwal menjadi aktif. Pilih untuk segera memulai atau pada tanggal dan waktu tertentu menggunakan YYYY/MM/DD format.
 - c. Untuk Rentang waktu, tentukan periode lookback untuk setiap eksekusi kueri. Masukkan durasi dalam menit yang menentukan seberapa jauh dari waktu eksekusi ke kueri.
 - d. Untuk Lanjutkan tanpa batas waktu, tentukan kapan jadwal berakhir. Pilih untuk menjalankan tanpa batas waktu atau sampai tanggal dan waktu tertentu menggunakan YYYY/MM/DD format.
6. Konsol menampilkan tiga kueri terjadwal berikutnya berdasarkan konfigurasi Anda, menunjukkan tanggal dan waktu yang tepat di UTC saat kueri akan dijalankan.
7. Dalam hasil kueri Posting ke S3 - bagian opsional (jika menggunakan tujuan S3):
 - a. Untuk bucket S3, pilih Akun ini jika bucket tujuan berada di AWS akun yang sama, atau pilih Akun lain jika bucket berada di AWS akun yang berbeda dan berikan ID akun pemilik ember sebagai input.
 - b. Untuk URI Amazon S3, masukkan bucket Amazon S3 dan awalan tempat hasil akan disimpan (misalnya, `s3://my-bucket/query-results/`). Jika Anda memilih Akun ini, Anda dapat memilih Jelajahi Amazon S3 untuk menavigasi dan memilih lokasi Amazon S3 yang ada.
 - c. (Opsional) Untuk ARN kunci KMS, masukkan ARN kunci yang AWS KMS dikelola pelanggan untuk mengenkripsi hasil kueri menggunakan SSE-KMS. Kuncinya harus berada di AWS Wilayah yang sama dengan bucket Amazon S3 tujuan.
8. Dalam peran IAM untuk memposting hasil kueri ke bagian Amazon S3, pilih salah satu opsi berikut:
 - a. Pilih Buat otomatis peran baru dengan izin default untuk secara otomatis menyiapkan peran IAM dengan izin yang diperlukan untuk CloudWatch Log untuk mengirimkan hasil kueri ke Amazon S3.
 - b. Pilih Gunakan peran yang ada untuk memilih peran IAM yang ada dengan kebijakan yang diperlukan untuk CloudWatch Log untuk mengirimkan hasil kueri ke Amazon S3. Gunakan bidang pencarian untuk menemukan dan memilih peran IAM yang sesuai dari daftar.

9. Di peran IAM untuk bagian eksekusi kueri terjadwal, pilih salah satu opsi berikut:
 - a. Pilih Auto-create peran baru dengan izin default untuk secara otomatis mengatur peran IAM dengan izin yang diperlukan untuk CloudWatch Log untuk menjalankan kueri terjadwal.
 - b. Pilih Gunakan peran yang ada untuk memilih peran IAM yang ada dengan kebijakan yang diperlukan untuk CloudWatch Log untuk menjalankan kueri terjadwal. Gunakan bidang pencarian untuk menemukan dan memilih peran IAM yang sesuai dari daftar.
10. Pilih Buat jadwal untuk membuat kueri terjadwal.

AWS CLI

Untuk membuat query terjadwal (AWS CLI)

- Gunakan `create-scheduled-query` perintah untuk membuat kueri terjadwal baru:

```
aws logs create-scheduled-query \
  --name "ErrorAnalysisQuery" \
  --query-language "CWL" \
  --query-string "fields @timestamp, @message | filter @message like /ERROR/ |
stats count() by bin(5m)" \
  --schedule-expression "cron(8 * * * ? *)" \
  --execution-role-arn "arn:aws:iam::123456789012:role/
CloudWatchLogsScheduledQueryRole" \
  --log-group-identifiers "/aws/lambda/my-function" "/aws/apigateway/my-api" \
  --state "ENABLED"
```

API

Untuk membuat kueri terjadwal (API)

- Gunakan `CreateScheduledQuery` tindakan untuk membuat kueri terjadwal baru. Contoh berikut membuat kueri terjadwal yang berjalan setiap jam:

```
{
  "name": "ErrorAnalysisQuery",
  "queryLanguage": "CWL",
  "queryString": "fields @timestamp, @message | filter @message like /ERROR/ |
stats count() by bin(5m)",
```

```
"scheduleExpression": "cron(8 * * * ? *)",
"executionRoleArn": "arn:aws:iam::123456789012:role/
CloudWatchLogsScheduledQueryRole",
"logGroupIdentifiers": ["/aws/lambda/my-function", "/aws/apigateway/my-
api"],
"state": "ENABLED"
}
```

Setelah membuat kueri terjadwal, Anda dapat melihat dan mengelolanya dari halaman Kueri terjadwal dan menggunakan ListScheduledQueries API, yang menampilkan semua kueri terjadwal dengan nama, tanggal pembuatan, status proses terakhir, waktu pemicu terakhir, dan frekuensi pengulangan.

Melihat dan mengelola kueri terjadwal

Informasi berikut tersedia untuk setiap kueri:

Nama

Nama unik yang Anda tetapkan untuk kueri terjadwal. Pilih nama untuk melihat konfigurasi terperinci dan riwayat eksekusi.

Tanggal pembuatan

Tanggal ketika kueri terjadwal dibuat, ditampilkan dalam YYYY-MM-DD format.

Status Last Run

Status eksekusi dari kueri terbaru yang dijalankan. Nilai yang mungkin termasuk:

- Selesai - Kueri berhasil dieksekusi dan hasilnya dikirim ke semua tujuan yang dikonfigurasi.
- Gagal - Eksekusi kueri atau pengiriman hasil gagal. Periksa riwayat eksekusi untuk detail kesalahan.
- Kueri Tidak Valid - Kueri tidak valid dan memiliki masalah sintaksis
- Timeout - Kueri telah habis waktunya. Waktu kueri habis secara otomatis setelah 60 menit

Waktu terakhir dipicu

Tanggal dan waktu ketika kueri terakhir dieksekusi, ditampilkan dalam format YYYY-MM-DD HH:MM:SS. Menampilkan Tidak Pernah jika kueri belum berjalan.

Mengulangi setiap

Frekuensi jadwal untuk kueri. Menampilkan Kustom untuk kueri menggunakan ekspresi cron atau deskripsi frekuensi tertentu untuk jadwal yang lebih sederhana.

Halaman kueri terjadwal memberikan ikhtisar semua kueri terjadwal Anda, menunjukkan status saat ini dan riwayat eksekusi sehingga Anda dapat melihat, memantau, dan mengelola semua kueri terjadwal Anda dari lokasi terpusat. Gunakan informasi ini untuk memantau kinerja kueri, mengidentifikasi masalah, dan mengelola alur kerja analisis log otomatis Anda.

Console

Untuk melihat kueri terjadwal (konsol)

1. Buka konsol CloudWatch Log di <https://us-east-1.console.aws.amazon.com/cloudwatch/rumah?region=us-east-1> # logsV2: logs-wawasan.
2. Di konsol CloudWatch Log, pilih Kueri terjadwal, Lihat kueri terjadwal.

AWS CLI

Untuk membuat daftar kueri terjadwal (AWS CLI)

- Gunakan `list-scheduled-queries` perintah untuk membuat daftar semua kueri terjadwal:

```
aws logs list-scheduled-queries --max-results 10
```

API

Untuk membuat daftar kueri terjadwal (API)

- Gunakan `ListScheduledQueries` tindakan untuk mengambil semua kueri terjadwal:

```
{
  "maxResults": 10
}
```

Header halaman kueri terjadwal menunjukkan jumlah total kueri terjadwal di akun Anda, membantu Anda melacak penggunaan dan mengelola alur kerja analisis log otomatis Anda secara efektif.

Melihat riwayat eksekusi kueri terjadwal

Gunakan riwayat eksekusi untuk memantau kinerja kueri terjadwal Anda dan memecahkan masalah apa pun dengan eksekusi kueri atau pengiriman hasil.

Riwayat eksekusi menunjukkan status setiap kueri yang dijalankan, termasuk eksekusi yang berhasil, kegagalan, dan hasil pemrosesan tujuan. Anda dapat menggunakan informasi ini untuk mengidentifikasi pola, mendiagnosis masalah, dan memverifikasi bahwa kueri Anda berjalan seperti yang diharapkan.

Console

Untuk melihat riwayat eksekusi (konsol)

1. Di konsol CloudWatch Log, pilih Kueri terjadwal, Lihat kueri terjadwal.
2. Pilih kueri terjadwal yang ingin Anda periksa.
3. Pilih tab Riwayat eksekusi.

AWS CLI

Untuk melihat riwayat eksekusi (AWS CLI)

1. Gunakan `get-scheduled-query-history` perintah untuk mengambil riwayat eksekusi untuk kueri terjadwal:

```
aws logs get-scheduled-query-history \
  --identifier "DailyErrorMonitoring" \
  --start-time 1743379200 \
  --end-time 1743465600 \
  --max-results 10
```

2. Untuk memfilter berdasarkan status eksekusi, tambahkan `--execution-statuses` parameter:

```
aws logs get-scheduled-query-history \
  --identifier "DailyErrorMonitoring" \
  --start-time 1743379200 \
```

```
--end-time 1743465600 \  
--max-results 1 \  
--execution-statuses "SUCCEEDED"
```

API

Untuk melihat riwayat eksekusi (API)

- Gunakan `GetScheduledQueryHistory` tindakan untuk mengambil riwayat eksekusi:

```
{  
  "identifier": "DailyErrorMonitoring",  
  "startTime": 1743379200,  
  "endTime": 1743465600,  
  "maxResults": 10,  
  "executionStatuses": ["SUCCEEDED", "FAILED"]  
}
```

Riwayat eksekusi menampilkan:

- Status eksekusi - Berjalan, Selesai, Gagal, Batas Waktu, atau `InvalidQuery`
- Waktu yang dipicu - Saat kueri dijalankan
- Tujuan - Status pemrosesan untuk setiap tujuan yang dikonfigurasi termasuk S3 dan EventBridge
- Pesan galat - Detail tentang kegagalan dalam eksekusi kueri atau pemrosesan tujuan

Memperbarui kueri terjadwal

Ubah konfigurasi kueri terjadwal Anda untuk mengubah string kueri, jadwal, tujuan, atau peran eksekusi saat persyaratan Anda berkembang.

Anda dapat memperbarui aspek apa pun dari kueri terjadwal, termasuk string kueri, ekspresi jadwal, tujuan, dan peran eksekusi. Perubahan segera berlaku untuk eksekusi di masa depan.

Console

Untuk memperbarui kueri terjadwal (konsol)

1. Di konsol CloudWatch Log, pilih Kueri terjadwal, Lihat kueri terjadwal.

2. Pilih kueri terjadwal yang ingin Anda perbarui.
3. Pilih Edit.
4. Ubah konfigurasi sesuai kebutuhan.
5. Pilih Simpan perubahan.

AWS CLI

Untuk memperbarui kueri terjadwal (AWS CLI)

- Gunakan `update-scheduled-query` perintah untuk memodifikasi kueri terjadwal yang ada:

```
aws logs update-scheduled-query \
  --identifier "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f" \
  --description "Monitor for ERROR level logs daily" \
  --query-language "LogsQL" \
  --query-string "fields @timestamp, @message | filter @message like /ERROR/"
\
  --log-group-identifiers "/aws/lambda/my-function-1" "/aws/lambda/my-
function-2"
```

API

Untuk memperbarui kueri terjadwal (API)

1. Gunakan `UpdateScheduledQuery` tindakan untuk mengubah konfigurasi kueri terjadwal:

```
{
  "identifier": "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f",
  "queryString": "fields @timestamp, @message | filter @message like /WARNING|
ERROR/ | stats count() by bin(5m)",
  "scheduleExpression": "cron(0 */2 * * ? *)",
  "state": "ENABLED"
}
```

2. Untuk memperbarui beberapa parameter konfigurasi sekaligus:

```
{
  "identifier": "arn:aws:logs:us-east-1:111122223333:scheduled-
query:5e0c0228-1c29-4d26-904f-59f1f1ba3c8f",
  "queryString": "fields @timestamp, @message, @level | filter @level =
'ERROR'",
  "scheduleExpression": "cron(0 8,12,16 * * ? *)",
  "executionRoleArn": "arn:aws:iam::111122223333:role/
UpdatedScheduledQueryRole",
  "logGroupIdentifiers": ["/aws/lambda/my-function", "/aws/lambda/another-
function"],
  "destinationConfiguration": {
    "s3Configuration": {
      "destinationIdentifier": "s3://111122223333-sqn-results-bucket/
processed-results",
      "roleArn": "arn:aws:iam::111122223333:role/Admin"
    }
  }
}
```

Mengkonfigurasi tujuan S3 untuk kueri terjadwal

Konfigurasi Amazon S3 sebagai tujuan untuk menyimpan hasil kueri terjadwal Anda sebagai file JSON untuk retensi dan analisis jangka panjang.

Saat menggunakan Amazon S3 sebagai tujuan, hasil kueri disimpan sebagai file JSON di bucket dan awalan yang ditentukan. Opsi ini sangat ideal untuk pengarsipan hasil, melakukan analisis batch, atau mengintegrasikan dengan AWS layanan lain yang memproses data S3.

Anda dapat mengirimkan hasil kueri ke bucket Amazon S3 di AWS akun yang sama dengan kueri terjadwal atau ke bucket di akun lain AWS. Anda juga dapat mengenkripsi hasil kueri secara opsional menggunakan AWS KMS kunci terkelola pelanggan (SSE-KMS).

Memberikan hasil ke bucket Amazon S3 di akun yang sama

Jika bucket Amazon S3 tujuan berada di AWS akun yang sama dengan kueri terjadwal, Anda dapat menelusuri dan memilih bucket langsung dari konsol.

Untuk mengonfigurasi tujuan Amazon S3 akun yang sama (konsol)

1. Di bagian Posting hasil kueri ke S3, untuk bucket S3, pilih Akun ini.

2. Untuk URI Amazon S3, masukkan bucket Amazon S3 dan awalan tempat hasil akan disimpan (misalnya `s3://my-bucket/query-results/`), atau pilih Jelajahi Amazon S3 untuk menavigasi dan memilih lokasi Amazon S3 yang ada.
3. (Opsional) Untuk mengenkripsi hasil dengan AWS KMS kunci yang dikelola pelanggan, masukkan ARN kunci di bidang AWS KMS ARN kunci KMS. Kuncinya harus berada di AWS Wilayah yang sama dengan bucket Amazon S3 tujuan. Jika Anda tidak menentukan AWS KMS kunci, setelah enkripsi default bucket akan berlaku.
4. Di peran IAM untuk memposting hasil kueri ke Amazon S3, pilih Buat otomatis peran baru dengan izin default untuk secara otomatis menyiapkan izin yang diperlukan, atau pilih Gunakan peran yang ada untuk memilih peran IAM yang ada dengan kebijakan yang diperlukan.

Peran IAM pengiriman tujuan memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::my-bucket/prefix/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "111122223333"
        }
      }
    }
  ]
}
```

Memberikan hasil ke bucket Amazon S3 di akun lain

Anda dapat mengirimkan hasil kueri terjadwal ke bucket Amazon S3 di akun lain AWS . Saat menggunakan bucket lintas akun, Anda harus memberikan URI Amazon S3 dan ID akun akun pemilik ember.

Untuk mengonfigurasi tujuan Amazon S3 lintas akun (konsol)

1. Di bagian Posting hasil kueri ke S3, untuk bucket S3, pilih Akun lain dan berikan ID akun akun pemilik ember sebagai input.

2. Untuk URI Amazon S3, masukkan URI Amazon S3 lengkap dari bucket tujuan dan awalan di akun lain (misalnya,). `s3://cross-account-bucket/query-results/`
3. (Opsional) Untuk mengenkripsi hasil dengan AWS KMS kunci yang dikelola pelanggan, masukkan ARN kunci di bidang AWS KMS ARN kunci KMS. Kuncinya harus berada di AWS Wilayah yang sama dengan bucket Amazon S3 tujuan.
4. Di peran IAM untuk memposting hasil kueri ke Amazon S3, pilih Buat otomatis peran baru dengan izin default untuk secara otomatis menyiapkan izin yang diperlukan, atau pilih Gunakan peran yang ada untuk memilih peran IAM yang ada dengan kebijakan yang diperlukan.

Pengiriman lintas akun memerlukan izin di kedua sisi. Peran IAM pengiriman tujuan di akun sumber memerlukan izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::cross-account-bucket/prefix/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Kebijakan bucket Amazon S3 di akun tujuan harus memberikan izin peran IAM akun sumber untuk menulis objek:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowScheduledQueryRolePutObject",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/my-s3-delivery-role"
      },

```

```

        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::cross-account-bucket/prefix/*"
    }
]
}

```

Mengenkripsi hasil dengan kunci yang dikelola pelanggan AWS KMS

Anda dapat menentukan AWS KMS kunci terkelola pelanggan secara opsional untuk mengenkripsi hasil kueri yang dikirimkan ke Amazon S3 menggunakan SSE-KMS. AWS KMS Kuncinya bisa di akun yang sama dengan kueri terjadwal atau di akun yang berbeda.

Saat Anda menentukan AWS KMS kunci, kueri terjadwal menggunakan kunci tersebut untuk mengenkripsi hasil melalui SSE-KMS. Jika Anda tidak menentukan AWS KMS kunci, setelah enkripsi default bucket akan berlaku. Jika bucket dikonfigurasi dengan enkripsi SSE-KMS default menggunakan kunci terkelola pelanggan, peran IAM pengiriman tujuan harus tetap memiliki `kms:GenerateDataKey` izin pada kunci tersebut.

Peran IAM pengiriman tujuan memerlukan `kms:GenerateDataKey` izin pada AWS KMS kunci. Contoh berikut menunjukkan izin yang diperlukan untuk tujuan Amazon S3 dengan kunci yang dikelola pelanggan AWS KMS :

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::my-bucket/prefix/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "111122223333"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "kms:GenerateDataKey",
      "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-id",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "s3.us-east-1.amazonaws.com"
        }
      }
    }
  ]
}

```

```

    },
    "StringLike": {
      "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::my-bucket*"
    }
  }
}
]
}

```

Jika AWS KMS kunci berada di akun yang berbeda dari peran IAM pengiriman tujuan, kebijakan AWS KMS kunci di akun pemilik kunci harus secara eksplisit memberikan akses peran:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowScheduledQueryRoleToEncrypt",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/my-s3-delivery-role"
      },
      "Action": "kms:GenerateDataKey",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:ViaService": "s3.us-east-1.amazonaws.com"
        },
        "StringLike": {
          "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::my-bucket*"
        }
      }
    }
  ]
}

```

Note

Ketika peran IAM AWS KMS kunci dan pengiriman tujuan berada di akun yang sama, kebijakan identitas IAM saja sudah cukup jika kebijakan AWS KMS kunci menyertakan pernyataan root “Aktifkan kebijakan IAM” default. Pemberian kebijakan AWS KMS kunci eksplisit hanya diperlukan jika kebijakan kunci tidak didelegasikan ke IAM.

Peran IAM untuk memposting hasil kueri ke Amazon S3 harus dikonfigurasi secara terpisah dari peran IAM untuk eksekusi kueri terjadwal. Pemisahan ini memungkinkan kontrol akses berbutir halus, di mana peran eksekusi dapat menjalankan kueri sementara peran Amazon S3 secara khusus menangani pengiriman hasil. Kedua peran harus menyertakan kebijakan kepercayaan yang memungkinkan layanan CloudWatch Log (`logs.amazonaws.com`) untuk mengambil peran.

Memecahkan masalah kueri terjadwal

Gunakan topik pemecahan masalah ini untuk menyelesaikan masalah umum dengan kueri terjadwal.

Eksekusi kueri gagal dengan kesalahan izin

Mengatasi kesalahan izin yang mencegah kueri terjadwal mengeksekusi atau mengirimkan hasil ke tujuan.

Kesalahan izin terjadi ketika peran eksekusi tidak memiliki izin yang diperlukan untuk membaca dari grup log atau menulis ke sumber daya tujuan.

Untuk mengatasi kesalahan izin

1. Verifikasi bahwa peran eksekusi memiliki `logs:StartQuerylogs:GetQueryResults`, dan `logs:DescribeLogGroups` izin untuk grup log target.
2. Pastikan peran eksekusi memiliki izin menulis untuk sumber daya tujuan (seperti `s3:PutObject` untuk bucket S3).
3. Periksa apakah kebijakan kepercayaan memungkinkan CloudWatch Log untuk mengambil peran eksekusi. Peran harus mempercayai prinsipal layanan log (`logs.amazonaws.com`) dalam kebijakan kepercayaannya.

Penyebab umum termasuk izin IAM yang hilang, sumber daya yang salah ARNs dalam kebijakan, atau masalah konfigurasi kebijakan kepercayaan.

Untuk mencegah kesalahan izin, gunakan prinsip hak istimewa paling sedikit saat membuat peran eksekusi dan izin uji sebelum menerapkan kueri terjadwal ke produksi.

Waktu kueri habis

Mengatasi kesalahan batas waktu yang terjadi ketika kueri terjadwal melebihi batas waktu eksekusi maksimum.

Batas waktu kueri terjadi ketika kueri membutuhkan waktu lebih dari 60 menit untuk memproses rentang data yang ditentukan, seringkali karena kumpulan data besar atau logika kueri yang kompleks.

Untuk mengatasi kesalahan batas waktu

1. Kurangi rentang waktu dengan mengurangi offset waktu mulai untuk memproses lebih sedikit data per eksekusi.
2. Optimalkan kueri dengan menambahkan filter di awal kueri untuk mengurangi jumlah data yang diproses. Gunakan indeks filter untuk mengurangi ukuran pemindaian data.
3. Pertimbangkan untuk memecah kueri kompleks menjadi kueri yang lebih sederhana dan lebih terfokus.

Penyebab umum termasuk menanyakan rentang waktu yang besar, memproses grup log volume tinggi, atau menggunakan agregasi kompleks tanpa penyaringan yang tepat.

Untuk mencegah batas waktu, uji kueri secara manual di Wawasan CloudWatch Log dengan volume data yang diharapkan dan optimalkan kinerja sebelum penjadwalan.

Pemrosesan tujuan gagal

Mengatasi kegagalan yang terjadi saat hasil kueri terjadwal tidak dapat dikirimkan ke tujuan yang dikonfigurasi.

Kegagalan pemrosesan tujuan terjadi ketika bucket Amazon S3 target atau bus EventBridge acara tidak dapat diakses atau tidak dikonfigurasi dengan benar.

Untuk mengatasi kegagalan di mana hasil kueri tidak dipublikasikan ke tujuan

1. Verifikasi bahwa bucket Amazon S3 yang ditentukan ada dan dapat diakses.
2. Periksa konfigurasi tujuan untuk benar URIs.
3. Pastikan peran eksekusi memiliki izin yang diperlukan untuk menulis ke tujuan.

Penyebab umum termasuk sumber daya tujuan yang dihapus atau diganti namanya, tujuan yang salah URIs, atau masalah konektivitas jaringan.

Untuk mencegah kegagalan tujuan, validasi konfigurasi tujuan secara teratur dan pantau ketersediaan sumber daya tujuan.

Kesalahan kueri tidak valid

Mengatasi kesalahan sintaks dan logika dalam string kueri terjadwal yang mencegah eksekusi berhasil.

Kesalahan kueri tidak valid terjadi ketika string kueri berisi kesalahan sintaks, mereferensikan bidang yang tidak ada, atau menggunakan fitur bahasa kueri yang tidak didukung.

Untuk mengatasi kesalahan kueri yang tidak valid

1. Uji kueri Anda secara manual di Wawasan CloudWatch Log untuk memverifikasi sintaks dan logika.
2. Periksa apakah semua bidang log yang direferensikan ada di grup log target Anda.
3. Verifikasi bahwa fitur bahasa kueri yang Anda gunakan didukung untuk kueri terjadwal.

Penyebab umum termasuk kesalahan ketik dalam nama bidang, sintaks kueri yang salah, atau menggunakan fitur kueri yang tidak didukung dalam lingkungan eksekusi terjadwal.

Untuk mencegah kesalahan kueri yang tidak valid, selalu uji kueri secara interaktif sebelum menjadwalkan dan gunakan fitur penemuan bidang untuk memverifikasi nama bidang.

Kesalahan Konkurensi Kueri

Ada beberapa poin penting yang disebutkan di bawah ini untuk diingat ketika kesalahan konkurensi dilihat sebagai kueri terjadwal menggunakan kuota yang sama dengan kueri wawasan Cloudwatch Logs. Disarankan untuk menyebarkan jadwal Anda untuk menghindari mencapai batas konkurensi.

- Kuota: Anda dapat menjalankan hingga 100 kueri Wawasan CloudWatch Log bersamaan per akun. AWS
- Dasbor: Kueri yang ditambahkan ke CloudWatch dasbor juga dihitung terhadap batas konkurensi ini, karena dijalankan saat dasbor dimuat atau di-refresh.
- OpenSearch Layanan PPL/SQL: Anda dapat menjalankan hingga 15 kueri OpenSearch PPL atau SQL bersamaan per akun. OpenSearch AWS
- Kueri lintas akun: Kuota konkurensi berlaku untuk kueri tunggal dan lintas akun. Saat menggunakan observabilitas CloudWatch lintas akun, kueri yang dimulai dalam akun pemantauan terhadap akun sumber tertaut juga dihitung terhadap batas konkurensi akun pemantauan.
- Grup Log Akses Jarang: Untuk grup log di kelas log akses yang jarang, jumlah maksimum kueri Wawasan Log bersamaan dibatasi hingga lima.

Deteksi anomali log

Anda dapat mendeteksi anomali dalam data log Anda dengan dua cara: dengan membuat detektor anomali log untuk pemantauan berkelanjutan, atau dengan menggunakan [anomaly detection](#) perintah dalam kueri Wawasan CloudWatch Log untuk analisis sesuai permintaan.

Detektor anomali log memindai peristiwa log yang dicerna ke dalam grup log dan menemukan anomali dalam data log secara otomatis. Deteksi anomali menggunakan pembelajaran mesin dan pengenalan pola untuk menetapkan dasar konten log yang khas. Untuk analisis sesuai permintaan, Anda dapat menggunakan `anomaly detection` perintah dalam kueri Wawasan CloudWatch Log untuk mengidentifikasi pola yang tidak biasa dalam data deret waktu. Untuk informasi lebih lanjut tentang deteksi anomali berbasis kueri, lihat. [Menggunakan deteksi anomali di Wawasan Log CloudWatch](#)

Setelah Anda membuat detektor anomali untuk grup log, ia berlatih menggunakan dua minggu terakhir peristiwa log di grup log untuk pelatihan. Periode pelatihan bisa memakan waktu hingga 15 menit. Setelah pelatihan selesai, ia mulai menganalisis log masuk untuk mengidentifikasi anomali, dan anomali ditampilkan di konsol CloudWatch Log untuk Anda periksa.

CloudWatch Pengenalan pola log mengekstrak pola log dengan mengidentifikasi konten statis dan dinamis di log Anda. Pola berguna untuk menganalisis kumpulan log besar karena sejumlah besar peristiwa log sering dapat dikompresi menjadi beberapa pola.

Misalnya, lihat contoh berikut dari tiga peristiwa log.

```
2023-01-01 19:00:01 [INFO] Calling DynamoDB to store for ResourceID: 12342342k124-12345
2023-01-01 19:00:02 [INFO] Calling DynamoDB to store for ResourceID: 324892398123-1234R
2023-01-01 19:00:03 [INFO] Calling DynamoDB to store for ResourceID: 3ff231242342-12345
```

Dalam sampel sebelumnya, ketiga peristiwa log mengikuti satu pola:

```
<Date-1> <Time-2> [INFO] Calling DynamoDB to store for resource id <ResourceID-3>
```

Bidang dalam pola disebut token. Bidang yang bervariasi dalam suatu pola, seperti ID permintaan atau stempel waktu, disebut sebagai token dinamis. Setiap nilai berbeda yang ditemukan untuk token dinamis disebut nilai token.

Jika CloudWatch Log dapat menyimpulkan jenis data yang diwakili oleh token dinamis, itu akan menampilkan token sebagai `<string-number>`. *string* Ini adalah deskripsi dari jenis data yang

diwakili oleh token. *number* Menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

CloudWatch Log menetapkan bagian string dari nama berdasarkan analisis konten peristiwa log yang berisi itu.

Jika CloudWatch Log tidak dapat menyimpulkan jenis data yang diwakili oleh token dinamis, ia menampilkan token sebagai <Token- *number* >, dan *number* menunjukkan di mana dalam pola token ini muncul, dibandingkan dengan token dinamis lainnya.

Contoh umum token dinamis termasuk kode kesalahan, alamat IP, stempel waktu, dan permintaan. IDs

Deteksi anomali log menggunakan pola-pola ini untuk menemukan anomali. Setelah periode pelatihan model detektor anomali, log dievaluasi terhadap tren yang diketahui. Detektor anomali menandai fluktuasi yang signifikan sebagai anomali.

Bab ini menjelaskan cara mengaktifkan deteksi anomali, melihat anomali, membuat alarm untuk detektor anomali log, dan metrik yang diterbitkan oleh detektor anomali log. Ini juga menjelaskan cara mengenkripsi detektor anomali dan hasilnya dengan. AWS Key Management Service

Membuat detektor anomali log tidak menimbulkan biaya.

Tingkat keparahan dan prioritas anomali dan pola

Setiap anomali yang ditemukan oleh detektor anomali log diberi prioritas. Setiap pola yang ditemukan diberi tingkat keparahan.

- Prioritas dihitung secara otomatis, dan didasarkan pada tingkat keparahan pola dan jumlah penyimpangan dari nilai yang diharapkan. Misalnya, jika nilai token tertentu tiba-tiba meningkat sebesar 500%, anomali itu mungkin ditetapkan sebagai HIGH prioritas bahkan jika tingkat keparahannya. NONE
- Tingkat keparahan hanya didasarkan pada kata kunci yang ditemukan dalam pola seperti FATAL, ERROR, dan WARN. Jika tidak satu pun dari kata kunci ini ditemukan, tingkat keparahan pola ditandai sebagai NONE.

Waktu visibilitas anomali

Saat Anda membuat detektor anomali, Anda menentukan periode visibilitas anomali maksimum untuknya. Ini adalah jumlah hari anomali ditampilkan di konsol dan dikembalikan oleh operasi [ListAnomalies](#) API. Setelah periode waktu ini berlalu untuk anomali, jika terus terjadi, itu secara otomatis diterima sebagai perilaku biasa dan model detektor anomali berhenti menandainya sebagai anomali.

Jika Anda tidak menyesuaikan waktu visibilitas saat membuat detektor anomali, 21 hari digunakan sebagai default.

Menekan anomali

Setelah anomali ditemukan, Anda dapat memilih untuk menekannya sementara atau permanen. Menekan anomali menyebabkan detektor anomali berhenti menandai kejadian ini sebagai anomali untuk jumlah waktu yang Anda tentukan. Ketika Anda menekan anomali, Anda dapat memilih untuk menekan hanya anomali spesifik itu, atau menekan semua anomali yang terkait dengan pola di mana anomali itu ditemukan.

Anda masih dapat melihat anomali yang ditekan di konsol. Anda juga dapat memilih untuk berhenti menekannya.

Pertanyaan umum

Apakah AWS menggunakan data saya untuk melatih algoritme pembelajaran mesin untuk AWS digunakan atau untuk pelanggan lain?

Tidak. Model deteksi anomali yang dibuat oleh pelatihan didasarkan pada peristiwa log dalam grup log dan hanya digunakan dalam grup log itu dan akun itu AWS .

Jenis peristiwa log apa yang bekerja dengan baik dengan deteksi anomali?

Deteksi anomali log sangat cocok untuk: Log aplikasi dan jenis log lainnya di mana sebagian besar entri log sesuai dengan pola tipikal. Grup log dengan peristiwa yang berisi tingkat log atau kata kunci tingkat keparahan seperti INFO, ERROR, dan DEBUG sangat cocok untuk mencatat deteksi anomali.

Deteksi anomali log tidak cocok untuk: Peristiwa log dengan struktur JSON yang sangat panjang, seperti Log. CloudTrail Analisis pola hanya menganalisis hingga 1500 karakter pertama dari garis log, sehingga karakter apa pun di luar batas itu dilewati.

Audit atau log akses, seperti log aliran VPC, juga akan kurang berhasil dengan deteksi anomali. Deteksi anomali dimaksudkan untuk menemukan masalah aplikasi, jadi mungkin tidak cocok untuk anomali jaringan atau akses.

Untuk membantu Anda menentukan apakah detektor anomali cocok untuk grup log tertentu, gunakan analisis pola CloudWatch Log untuk menemukan jumlah pola dalam peristiwa log dalam grup. Jika jumlah pola tidak lebih dari sekitar 300, deteksi anomali mungkin bekerja dengan baik. Untuk informasi lebih lanjut tentang analisis pola, lihat [Analisis pola](#).

Apa yang ditandai sebagai anomali?

Kejadian berikut dapat menyebabkan peristiwa log ditandai sebagai anomali:

- Peristiwa log dengan pola yang tidak terlihat sebelumnya di grup log.
- Variasi yang signifikan terhadap pola yang diketahui.
- Nilai baru untuk token dinamis yang memiliki serangkaian nilai biasa yang terpisah.
- Perubahan besar dalam jumlah kemunculan nilai untuk token dinamis.

Meskipun semua item sebelumnya mungkin ditandai sebagai anomali, itu tidak semua berarti bahwa aplikasi berkinerja buruk. Misalnya, higher-than-usual sejumlah nilai 200 keberhasilan mungkin ditandai sebagai anomali. Dalam kasus seperti ini, Anda mungkin mempertimbangkan untuk menekan anomali ini yang tidak menunjukkan masalah.

Apa yang terjadi dengan data sensitif yang sedang disembunyikan?

Setiap bagian dari peristiwa log yang disamarkan sebagai data sensitif tidak dipindai untuk anomali. Untuk informasi selengkapnya tentang menyembunyikan data sensitif, lihat [Membantu melindungi data log sensitif dengan masking](#).

Menggunakan deteksi anomali di Wawasan Log CloudWatch

Selain membuat detektor anomali log untuk pemantauan berkelanjutan, Anda juga dapat menggunakan `anomaly` perintah dalam kueri Wawasan CloudWatch Log untuk mengidentifikasi pola yang tidak biasa dalam data log sesuai permintaan. Perintah ini memperluas `pattern` fungsionalitas yang ada dan menggunakan pembelajaran mesin untuk mendeteksi lima jenis anomali termasuk perubahan frekuensi pola, pola baru, dan variasi token.

`anomaly` Perintah ini sangat berguna untuk:

- Analisis ad-hoc data log historis untuk mengidentifikasi pola yang tidak biasa
- Menyelidiki periode waktu tertentu untuk perilaku anomali
- Memantau aplikasi seperti fungsi Lambda untuk masalah eksekusi

Untuk informasi selengkapnya tentang menggunakan anomaly perintah dalam kueri Anda, lihat [anomaly](#).

Deteksi anomali berbasis kueri ini melengkapi detektor anomali kontinu yang dijelaskan di bagian berikut, memberi Anda kemampuan pemantauan waktu nyata dan analisis sesuai permintaan.

Aktifkan deteksi anomali pada grup log

Gunakan langkah-langkah berikut untuk menggunakan CloudWatch konsol untuk membuat detektor anomali log yang memindai grup log untuk mengetahui anomali.

Anda juga dapat membuat detektor anomali secara terprogram. Untuk informasi selengkapnya, lihat [CreateLogAnomalyDetector](#).

Untuk membuat detektor anomali log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Log, Log Anomali.
3. Pilih Buat detektor anomali.
4. Pilih grup log untuk membuat detektor anomali ini.
5. Masukkan nama untuk detektor dalam nama detektor anomali.
6. (Opsional) Ubah frekuensi Evaluasi dari default 5 menit. Tetapkan nilai ini sesuai dengan frekuensi grup log menerima log baru. Misalnya, jika grup log menerima peristiwa log baru dalam batch setiap 10 menit, maka pengaturan frekuensi evaluasi menjadi 15 menit mungkin tepat.
7. (Opsional) Untuk mengonfigurasi detektor anomali untuk mencari anomali hanya dalam peristiwa log yang berisi kata atau string tertentu, pilih Pola filter.

Kemudian, masukkan pola dalam pola filter deteksi anomali. Untuk informasi lebih lanjut tentang sintaks pola, [Filter sintaks pola untuk filter metrik, filter langganan, peristiwa log filter, dan Live Tail](#).

(Opsional) Untuk menguji pola filter Anda, masukkan beberapa pesan log ke dalam pesan peristiwa Log dan kemudian pilih Pola Uji.

8. (Opsional) Untuk mengubah periode visibilitas anomali dari default atau untuk mengaitkan AWS KMS kunci dengan detektor anomali ini, pilih Konfigurasi lanjutan.
 - a. Untuk mengubah periode visibilitas anomali dari default, masukkan nilai baru di Periode visibilitas anomali maksimum (hari).
 - b. Untuk mengaitkan AWS KMS kunci dengan detektor anomali ini, masukkan ARN di ARN kunci KMS. Jika Anda menetapkan kunci, informasi anomali yang ditemukan oleh detektor ini dienkripsi saat istirahat dengan kunci. Pengguna harus memiliki izin untuk kunci ini dan detektor anomali untuk mengambil informasi tentang anomali yang ditemukannya.

Anda juga harus memastikan bahwa kepala layanan CloudWatch Log memiliki izin untuk menggunakan kunci. Untuk informasi selengkapnya, lihat [Enkripsi detektor anomali dan hasilnya dengan AWS KMS](#).

9. Pilih Aktifkan Deteksi Anomali.

Detektor anomali dibuat dan mulai melatih modelnya, berdasarkan peristiwa log yang dicerna grup log. Setelah sekitar 15 menit, deteksi anomali aktif dan mulai menemukan dan anomali permukaan.

Lihat anomali yang telah ditemukan

Setelah Anda membuat satu atau lebih detektor anomali log, Anda dapat menggunakan CloudWatch konsol untuk melihat anomali yang mereka temukan.

Anda dapat melihat anomali secara terprogram. Untuk informasi selengkapnya, lihat [ListAnomalies](#).

Untuk melihat anomali yang ditemukan oleh semua detektor anomali log Anda

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Log, Log Anomali.

Tabel anomali Log muncul. Nomor di bagian atas di sebelah anomali Log menampilkan berapa banyak anomali log yang tercantum dalam tabel. Setiap baris dalam tabel menampilkan informasi berikut:

- Kolom Anomali menampilkan ringkasan singkat anomali. Ringkasan ini dihasilkan oleh CloudWatch Log.

- Prioritas anomali. Prioritas dihitung secara otomatis berdasarkan jumlah perubahan dalam peristiwa log, kata kunci seperti yang `Exception` terjadi dalam peristiwa log, dan banyak lagi.
 - Pola Log yang menjadi dasar anomali. Untuk informasi lebih lanjut tentang pola, lihat [Deteksi anomali log](#).
 - Tren log anomali menampilkan histogram yang menggambarkan volume log yang cocok dengan pola.
 - Waktu deteksi terakhir menampilkan waktu terbaru anomali ini ditemukan.
 - Waktu deteksi pertama menunjukkan pertama kali anomali ini ditemukan.
 - Detektor anomali menampilkan nama grup log yang berisi peristiwa log yang terkait dengan anomali ini. Anda dapat memilih nama ini untuk melihat halaman detail grup log.
3. Untuk memeriksa lebih lanjut satu anomali, pilih tombol radio di barisnya.

Panel pemeriksaan Pola muncul dan menampilkan yang berikut:

- Pola yang menjadi dasar anomali ini. Pilih token dalam pola untuk menganalisis nilai token tersebut.
- Histogram yang menunjukkan jumlah kemunculan anomali selama rentang waktu yang ditanyakan.
- Tab sampel Log menampilkan beberapa peristiwa log yang merupakan bagian dari anomali.
- Tab Nilai Token menampilkan nilai token dinamis yang dipilih, jika Anda telah memilihnya.

 Note

Maksimal 10 nilai token ditangkap untuk setiap token. Jumlah token mungkin tidak tepat. CloudWatch Log menggunakan penghitung probabilistik untuk menghasilkan jumlah token, bukan nilai absolut.

4. Untuk menekan anomali, pilih tombol radio di barisnya lalu lakukan hal berikut:
- a. Pilih Tindakan, Menekan Anomali.
 - b. Kemudian tentukan berapa lama Anda ingin anomali ditekan.
 - c. Untuk menekan semua anomali yang terkait dengan pola ini, pilih Suppress Pattern.
 - d. Pilih Menekan anomali.

Untuk melihat anomali yang ditemukan dalam satu grup log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Log, Grup log.
3. Pilih nama grup log, lalu pilih tab Deteksi anomali.

Tabel deteksi anomali muncul. Nomor di bagian atas di sebelah anomali Log menampilkan berapa banyak anomali log yang tercantum dalam tabel. Setiap baris dalam tabel menampilkan informasi berikut:

- Kolom Anomali menampilkan ringkasan singkat anomali. Ringkasan ini dihasilkan oleh CloudWatch Log.
 - Prioritas anomali. Prioritas dihitung secara otomatis berdasarkan jumlah perubahan dalam peristiwa log, kata kunci seperti yang `Exception` terjadi dalam peristiwa log, dan banyak lagi.
 - Pola Log yang menjadi dasar anomali. Untuk informasi lebih lanjut tentang pola, lihat [Deteksi anomali log](#).
 - Tren log anomali menampilkan histogram yang menggambarkan volume log yang cocok dengan pola.
 - Waktu deteksi terakhir menampilkan waktu terbaru anomali ini ditemukan.
 - Waktu deteksi pertama menunjukkan pertama kali anomali ini ditemukan.
4. Untuk memeriksa lebih lanjut satu anomali, pilih tombol radio di barisnya.

Panel pemeriksaan Pola muncul dan menampilkan yang berikut:

- Pola yang menjadi dasar anomali ini. Pilih token dalam pola untuk menganalisis nilai token tersebut.
- Histogram yang menunjukkan jumlah kemunculan anomali selama rentang waktu yang ditanyakan.
- Tab sampel Log menampilkan beberapa peristiwa log yang merupakan bagian dari anomali.
- Tab Nilai Token menampilkan nilai token dinamis yang dipilih, jika Anda telah memilihnya.

 Note

Maksimal 10 nilai token ditangkap untuk setiap token. Jumlah token mungkin tidak tepat. CloudWatch Log menggunakan penghitung probabilistik untuk menghasilkan jumlah token, bukan nilai absolut.

5. Untuk menekan anomali, pilih tombol radio di barisnya lalu lakukan hal berikut:
 - a. Pilih Tindakan, Menekan Anomali.
 - b. Kemudian tentukan berapa lama Anda ingin anomali ditekan.
 - c. Untuk menekan semua anomali yang terkait dengan pola ini, pilih Suppress Pattern.
 - d. Pilih Menekan anomali.

Buat alarm pada detektor anomali log

Anda dapat membuat alarm untuk detektor anomali log di grup log. Anda dapat menentukan agar alarm masuk ke ALARM keadaan ketika sejumlah anomali tertentu ditemukan di grup log selama periode waktu tertentu. Anda juga dapat menggunakan filter sehingga hanya anomali prioritas tertentu yang dihitung oleh alarm.

Untuk membuat alarm untuk detektor anomali log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Log Anomali.

Tabel detektor anomali log muncul.

3. Pilih tombol radio untuk detektor anomali yang ingin Anda atur alarmnya, dan pilih Buat alarm.

Wizard pembuatan CloudWatch alarm muncul. LogAnomalyDetectorBidang menampilkan nama detektor anomali yang Anda pilih. Bidang nama metrik ditampilkan AnomalyCount.

4. (Opsional) Untuk memfilter alarm ini untuk prioritas anomali, lakukan salah satu hal berikut:
 - Agar alarm hanya menghitung anomali prioritas tinggi, masukkan untuk. **HIGH** LogAnomalyPriority
 - Agar alarm hanya menghitung anomali prioritas tinggi dan menengah, masukkan. **MEDIUM** LogAnomalyPriority

Untuk informasi selengkapnya tentang tingkat prioritas, lihat [Tingkat keparahan dan prioritas anomali dan pola](#).

5. Pilih untuk menggunakan ambang deteksi anomali statis atau metrik untuk alarm. Pilihan ini menentukan bagaimana ambang alarm diatur. Ambang batas statis berarti bahwa ambang alarm adalah angka statis dan konstan yang Anda pilih. Ambang deteksi anomali berarti CloudWatch menentukan rentang nilai yang biasa, dan alarm memicu jika hitungan aktual melewati ambang

batas pita ini. Anda tidak harus memilih Deteksi anomali untuk alarm deteksi anomali log. Untuk informasi lebih lanjut tentang deteksi anomali metrik, lihat [Menggunakan deteksi CloudWatch anomali](#).

6. Untuk Kapanpun ***your-metric-name***, pilih Lebih Besar, Lebih Hebat/Sama, Lebih Rendah/Sama, atau Lebih Rendah. Kemudian untuk dari . . ., masukkan angka untuk nilai ambang batas Anda. Alarm masuk ke **ALARM** keadaan jika detektor anomali menemukan lebih dari jumlah alarm ini selama waktu yang ditentukan oleh Periode.
7. Pilih Konfigurasi tambahan. Untuk Titik data alarm, tentukan berapa banyak periode evaluasi (titik data) yang harus ada dalam status ALARM untuk memicu alarm. Jika kedua nilai di sini cocok, Anda membuat alarm yang beralih ke status ALARM jika terjadi pelanggaran selama sebanyak itu dalam periode berturut-turut.

Untuk membuat sebuah alarm M dari N, Anda harus menentukan angka untuk nilai pertama dengan nilai yang lebih rendah dari angka untuk nilai kedua. Untuk informasi selengkapnya, lihat [Mengevaluasi alarm](#).

8. Untuk Perlakuan data yang hilang, pilih perilaku alarm ketika ada beberapa titik data yang hilang. Untuk informasi selengkapnya, lihat [Mengonfigurasi cara CloudWatch alarm menangani data yang hilang](#).
9. Pilih Berikutnya.
10. Untuk Pemberitahuan, pilih Tambahkan pemberitahuan, lalu tentukan topik Amazon SNS yang akan diberi tahu saat alarm Anda bertransisi ke ALARM,, OK atau status. INSUFFICIENT_DATA
 - a. (Opsional) Untuk mengirimkan beberapa notifikasi untuk status alarm yang sama atau status alarm yang berbeda, silakan pilih Tambahkan notifikasi.

 Note

Kami menyarankan Anda untuk menyetel alarm untuk mengambil tindakan ketika alarm beralih statusnya menjadi data tidak mencukupi selain ketika beralih status menjadi Alarm. Hal ini dilakukan karena banyak masalah dengan fungsi Lambda yang terhubung ke sumber data yang dapat menyebabkan alarm beralih statusnya menjadi Data tidak mencukupi.

- b. (Opsional) Jika tidak ingin mengirimkan notifikasi Amazon SNS, silakan pilih Hapus.

11. (Opsional) Jika Anda ingin alarm melakukan tindakan untuk Amazon EC2 Auto Scaling, Amazon EC2, tiket AWS Systems Manager, atau, pilih tombol yang sesuai, dan tentukan status dan tindakan alarm.

 Note

Alarm Anda dapat melakukan tindakan Systems Manager hanya ketika alarm tersebut berada dalam status ALARM. Untuk informasi tentang tindakan Systems Manager, lihat [Mengkonfigurasi CloudWatch untuk membuat OpsItems](#) dan [Pembuatan insiden](#).

12. Pilih Berikutnya.
13. Pada Nama dan deskripsi, Anda harus memasukkan nama dan deskripsi untuk alarm Anda, dan kemudian pilih Berikutnya. Nama tersebut harus menggunakan karakter UTF-8, dan tidak dapat berisi karakter kontrol ASCII. Deskripsi dapat mencakup pemformatan penurunan harga, yang hanya ditampilkan di tab Detail alarm di CloudWatch konsol. Penurunan harga dapat Anda gunakan untuk menambahkan tautan ke runbook atau sumber daya internal lainnya.

 Tip

Nama alarm harus menggunakan karakter UTF-8 saja. Nama tersebut tidak boleh memuat karakter kontrol ASCII.

14. Pada Pratinjau dan buat, silakan Anda konfirmasi bahwa informasi dan kondisi alarm Anda sudah benar, dan kemudian pilih Buat alarm.

Metrik yang diterbitkan oleh detektor anomali log

CloudWatch Log menerbitkan AnomalyCountmetrik ke CloudWatch metrik. Metrik ini dipublikasikan ke AWS/Logs namespace.

AnomalyCountMetrik diterbitkan dengan dimensi berikut:

- LogAnomalyDetector— Nama detektor anomali
- LogAnomalyPriority— Tingkat prioritas anomali

Enkripsi detektor anomali dan hasilnya dengan AWS KMS

Data detektor anomali selalu dienkripsi di Log. CloudWatch Secara default, CloudWatch Log menggunakan enkripsi sisi server untuk data saat istirahat. Sebagai alternatif, Anda dapat menggunakan AWS Key Management Service enkripsi ini. Jika Anda melakukannya, enkripsi dilakukan dengan menggunakan AWS KMS kunci. Penggunaan enkripsi AWS KMS diaktifkan pada tingkat detektor anomali, dengan mengaitkan kunci KMS dengan detektor anomali.

Important

CloudWatch Log hanya mendukung kunci KMS simetris. Jangan gunakan kunci asimetris untuk mengenkripsi data dalam grup log Anda. Untuk informasi selengkapnya, lihat [Menggunakan Kunci Simetris dan Asimetris](#).

Batas

- Untuk melakukan langkah-langkah berikut, Anda harus memiliki izin berikut: `kms:CreateKey`, `kms:GetKeyPolicy`, dan `kms:PutKeyPolicy`.
- Setelah Anda mengaitkan atau melepaskan kunci dari detektor anomali, diperlukan waktu hingga lima menit agar operasi diterapkan.
- Jika Anda mencabut akses CloudWatch Log ke kunci terkait atau menghapus kunci KMS terkait, data terenkripsi Anda di CloudWatch Log tidak dapat diambil lagi.

Langkah 1: Buat AWS KMS kunci

Untuk membuat kunci KMS, gunakan perintah [create-key](#) berikut:

```
aws kms create-key
```

Output berisi ID kunci dan Amazon Resource Name (ARN) dari kunci. Berikut ini adalah output contoh:

```
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
```

```
{
  "KeyId": "key-default-1",
  "Description": "",
  "KeyManager": "CUSTOMER",
  "Enabled": true,
  "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
  "KeyUsage": "ENCRYPT_DECRYPT",
  "KeyState": "Enabled",
  "CreationDate": 1478910250.94,
  "Arn": "arn:aws:kms:us-west-2:123456789012:key/key-default-1",
  "AWSAccountId": "123456789012",
  "EncryptionAlgorithms": [
    "SYMMETRIC_DEFAULT"
  ]
}
```

Langkah 2: Tetapkan izin pada tombol KMS

Secara default, semua AWS KMS kunci bersifat pribadi. Hanya pemilik sumber daya yang dapat menggunakannya untuk mengenkripsi dan mendekripsi data. Namun, pemilik sumber daya dapat memberikan izin untuk mengakses kunci KMS ke pengguna dan sumber daya lain. Dengan langkah ini, Anda memberikan izin utama layanan CloudWatch Log untuk menggunakan kunci. Prinsipal layanan ini harus berada di AWS Wilayah yang sama di mana kunci KMS disimpan.

Sebagai praktik terbaik, kami menyarankan Anda membatasi penggunaan kunci KMS hanya untuk AWS akun atau detektor anomali yang Anda tentukan.

Pertama, simpan kebijakan default untuk kunci KMS Anda seperti `policy.json` menggunakan [get-key-policy](#) perintah berikut:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./policy.json
```

Buka file `policy.json` di editor teks dan tambahkan bagian dalam huruf tebal dari salah satu pernyataan berikut. Pisahkan pernyataan yang ada dari pernyataan baru dengan koma. Pernyataan ini menggunakan Condition bagian untuk meningkatkan keamanan AWS KMS kunci. Untuk informasi selengkapnya, lihat [AWS KMS kunci dan konteks enkripsi](#).

ConditionBagian dalam contoh ini membatasi penggunaan AWS KMS kunci ke akun yang ditentukan, tetapi dapat digunakan untuk detektor anomali apa pun.

JSON

```

{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "EnableIAMUserPermissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowCloudWatchLogsEncryption",
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "StringLike": {
          "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-east-1:123456789012:anomaly-detector:*"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:logs:us-east-1:123456789012:anomaly-detector:*"
        }
      }
    }
  ]
}

```

```

        "Sid": "AllowCloudWatchLogsDescribeKey",
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
        },
        "Action": "kms:DescribeKey",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "123456789012"
            }
        }
    },
    {
        "Sid": "AllowCloudWatchLogsReEncryption",
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.us-east-1.amazonaws.com"
        },
        "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "123456789012"
            },
            "StringLike": {
                "kms:EncryptionContext:aws-crypto-ec:aws:logs:arn":
                "arn:aws:logs:us-east-1:123456789012:anomaly-detector:*"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:logs:us-
east-1:123456789012:anomaly-detector:*"
            }
        }
    },
    {
        "Sid": "AllowCloudWatchLogsDescribeKeyForReEncryption",
        "Effect": "Allow",
        "Principal": {

```



```
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": "kms:DescribeKey",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Terakhir, tambahkan kebijakan yang diperbarui menggunakan [put-key-policy](#) perintah berikut:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Langkah 3: Kaitkan kunci KMS dengan detektor anomali

Anda dapat mengaitkan kunci KMS dengan detektor anomali saat Anda membuatnya di konsol atau menggunakan `aws kms` CLI APIs

Langkah 4: Lepaskan kunci dari detektor anomali

Setelah kunci dikaitkan dengan detektor anomali, Anda tidak dapat memperbarui kunci. Satu-satunya cara untuk menghapus kunci adalah dengan menghapus detektor anomali, dan kemudian membuatnya kembali.

Memecahkan Masalah dengan CloudWatch Log Live Tail

CloudWatch Logs Live Tail membantu Anda memecahkan masalah insiden dengan cepat dengan melihat daftar streaming peristiwa log baru saat tertelan. Anda dapat melihat, memfilter, dan menyorot log yang dicerna dalam waktu dekat, membantu Anda mendeteksi dan menyelesaikan masalah dengan cepat. Anda dapat memfilter log berdasarkan istilah yang Anda tentukan, dan juga menyorot log yang berisi istilah tertentu untuk membantu Anda menemukan apa yang Anda cari dengan cepat.

Sesi Live Tail dikenakan biaya berdasarkan waktu penggunaan sesi, per menit. Untuk informasi selengkapnya tentang harga, lihat tab Log di [CloudWatch Harga Amazon](#).

Live Tail hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

Bagian berikut menjelaskan cara menggunakan Live Tail di konsol dan di AWS CLI. Anda juga dapat memulai sesi Live Tail secara terprogram. Untuk informasi selengkapnya, lihat [StartLiveTail](#). Untuk contoh SDK, lihat [Memulai sesi Live Tail menggunakan AWS SDK](#).

Anda juga dapat menggunakan Live Tail di AWS Toolkit for Visual Studio Code. Untuk memulai sesi Live Tail dari VS Code Command Palette, lihat [bagian Amazon CloudWatch Logs Live Tail](#) dari Panduan AWS Toolkit for Visual Studio Code Pengguna.

Fitur Live Tail tersedia di semua AWS [Wilayah](#) komersial. Ini tidak tersedia di Wilayah Tiongkok atau AWS GovCloud Wilayah (AS).

Note

StartLiveTailAPI merutekan permintaan menggunakan injeksi awalan host SDK. Versi SDK yang dirilis sebelum 1 April 2026 dirutekan ke `streaming-logs.Region.amazonaws.com`, yang tidak mendukung titik akhir VPC. Versi SDK yang dirilis pada atau setelah 1 April 2026 dirutekan ke `stream-logs.Region.amazonaws.com`, yang mendukung titik akhir VPC. Untuk menyiapkan titik akhir VPC untuk API ini, lihat Membuat titik akhir [VPC](#) untuk Log. CloudWatch

Mulai sesi Live Tail menggunakan AWS CLI

`start-live-tail` AWS CLI Perintah memulai sesi streaming Live Tail untuk satu atau beberapa grup log di terminal. Sesi Live Tail dapat berlangsung hingga tiga jam. Jika lebih dari 500 peristiwa log per detik cocok dengan filter, peristiwa log yang ditampilkan adalah contoh dari total peristiwa log, untuk memberikan pengalaman tailing real-time. Untuk informasi selengkapnya tentang `start-live-tail` perintah, lihat [start-live-tail](#)

Anda dapat menggunakan `start-live-tail` dalam dua mode:

- `print-only` — ini adalah mode default
- interaktif

hanya cetak

Dalam `print-only` mode, peristiwa log dialirkan di terminal. Peristiwa baru ditambahkan di bagian bawah setiap detik, menciptakan pengalaman tailing mendekati waktu nyata yang mirip dengan `tail -f` di Linux.

Untuk memulai sesi Live Tail dalam mode cetak saja, masukkan perintah berikut.

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-east-1:111111222222:log-group:my-logs
```

Saat Anda menggunakan mode cetak saja, Anda juga dapat menyalurkannya dengan perintah Linux lainnya untuk meningkatkan kemampuan analitisnya. Contoh berikut menyaring peristiwa log dengan `error` kata kunci dan mencetak kolom kedua dan keempat dari peristiwa ini untuk membantu Anda mengekstrak informasi tertentu.

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-east-1:111111222222:log-group:my-logs --mode print-only | grep "error" | awk '{print $2, $4}'
```

interaktif

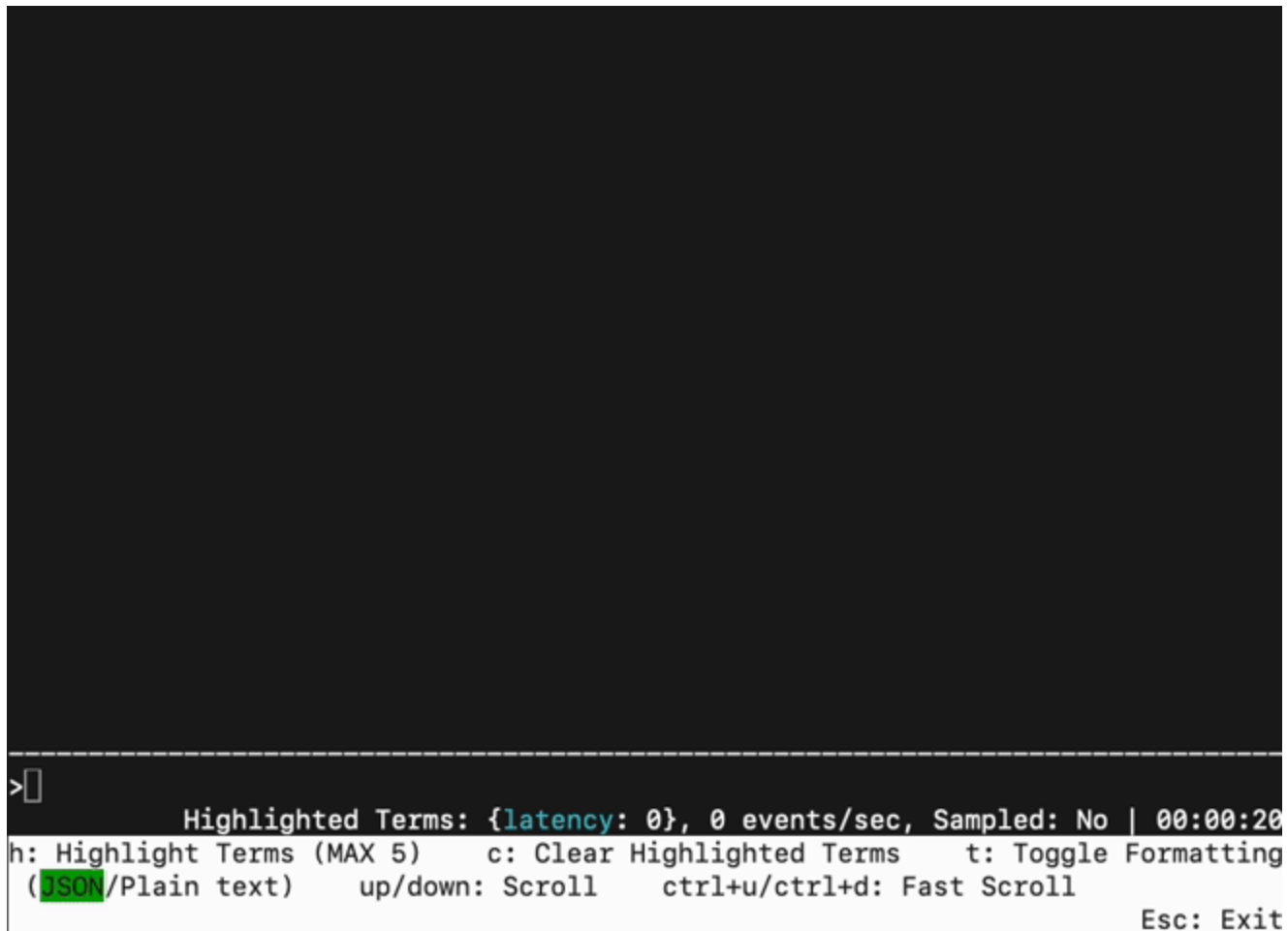
Dalam `interactive` mode, Anda dapat menyorot istilah dan beralih format peristiwa log keluaran antara JSON dan teks biasa. Mode interaktif juga menampilkan informasi tentang sesi Live Tail

seperti durasi sesi, apakah sesi sedang diambil sampelnya, dan istilah yang disorot saat ini serta jumlah waktu yang telah ditemui.

Untuk memulai sesi Live Tail dalam mode interaktif, masukkan perintah berikut.

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-east-1:111111222222:log-group:my-logs --mode interactive
```

Sesi Live Tail dimulai. Video berikut menunjukkan bagian dari sesi contoh.



Untuk menyorot istilah dalam log streaming, tekan h lalu masukkan istilah. Berikut ini menunjukkan layar setelah istilah `latency` telah disorot.

Untuk menghapus istilah yang disorot, tekan c lalu ketik angka yang mewakili istilah yang ingin Anda hentikan penyorotan.

Anda dapat menekan t untuk beralih format tampilan acara masuk antara JSON dan teks biasa. Fungsionalitas sakelar ini adalah upaya terbaik dan hanya terjadi jika format peristiwa log kompatibel.

Anda dapat menggunakan panah atas dan bawah tombol panah untuk menggulir, dan menggunakan CTRL+u dan CTRL+d untuk menggulir lebih cepat.

Gambar berikut menampilkan penyorotan latency istilah selama sesi Live Tail.

```

2024-06-27 12:34:56 [INFO] User login successful
2024-06-27 12:34:56 [ERROR] Disk space exhausted
2024-06-27 12:34:56 [WARN] Unauthorized access attempt
2024-06-27 12:34:56 [WARN] Disk space running low
2024-06-27 12:34:56 [INFO] User logout successful
2024-06-27 12:34:56 [WARN] High latency in network.
2024-06-27 12:34:57 [ERROR] Database connection failed
2024-06-27 12:34:57 [INFO] Database connection established
2024-06-27 12:34:57 [WARN] SSL certificate is about to expire
2024-06-27 12:34:57 [INFO] Scheduled task started
2024-06-27 12:34:57 [WARN] Network latency detected.
2024-06-27 12:34:57 [WARN] Outdated library version
2024-06-27 12:34:58 [INFO] New user registered
2024-06-27 12:34:58 [INFO] Database query executed
2024-06-27 12:34:58 [INFO] File uploaded successfully
2024-06-27 12:34:58 [WARN] Memory usage is high
2024-06-27 12:34:59 [ERROR] Unable to connect to server
[INFO] Connection established with the server
[WARN] SSL certificate is about to expire
[INFO] Scheduled task started

```

Highlighted Terms: {latency: 2}, 0 events/sec, Sampled: No | 00:08:21

h: Highlight Terms (MAX 5) c: Clear Highlighted Terms t: Toggle Formatting
 (JSON/Plain text) up/down: Scroll ctrl+u/ctrl+d: Fast Scroll Esc: Exit

Memulai sesi Live Tail di konsol

Anda menggunakan CloudWatch konsol untuk memulai sesi Live Tail. Prosedur berikut menjelaskan cara memulai sesi Live Tail dengan menggunakan opsi Live tail di panel navigasi kiri. Anda juga dapat memulai sesi Live Tail dari halaman Grup Log atau halaman Wawasan CloudWatch Log.

Jika Anda menggunakan kebijakan perlindungan data untuk menutupi data sensitif dalam grup log yang Anda lihat dengan Live Tail, data sensitif akan selalu muncul bertopeng di sesi Live Tail. Untuk informasi selengkapnya tentang menyembunyikan data sensitif di grup log, lihat [Membantu melindungi data log sensitif dengan masking](#).

⚠ Important

Jika tim keamanan jaringan Anda tidak mengizinkan penggunaan soket web, saat ini Anda tidak dapat mengakses bagian Live Tail CloudWatch konsol. Anda dapat menggunakan Live Tail dengan AWS CLI atau APIs. Untuk informasi selengkapnya, lihat [Mulai sesi Live Tail menggunakan AWS CLI](#) dan [StartLiveTail](#).

Untuk memulai sesi Live Tail

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Live tail.
3. Untuk Pilih grup log, pilih grup log tempat Anda ingin melihat peristiwa, di sesi Live Tail. Anda dapat memilih sebanyak 10 grup log.
4. (Opsional) Jika Anda memilih hanya satu grup log, Anda dapat memfilter sesi Live Tail Anda lebih lanjut dengan memilih satu atau beberapa aliran log untuk melihat peristiwa log. Untuk melakukannya, di bawah Pilih aliran log, pilih nama aliran log dari daftar drop-down. Atau, Anda dapat menggunakan kotak kedua di bawah Pilih aliran log untuk memasukkan awalan nama aliran log, dan kemudian semua aliran log dengan nama yang cocok dengan awalan akan dipilih.
5. (Opsional) Untuk menampilkan hanya peristiwa log yang berisi kata-kata tertentu atau string lainnya, masukkan kata atau string diAdd filter patterns.

Misalnya, untuk menampilkan hanya peristiwa log yang menyertakan kataWarning, masukkan**Warning**. Bidang filter peka huruf besar/kecil. Anda dapat menyertakan beberapa operator istilah dan pola di bidang ini:

- **error 404**hanya menampilkan peristiwa log yang mencakup keduanya error dan 404
- **?Error ?error**menampilkan peristiwa log yang mencakup salah satu Error atau error
- **-INFO**menampilkan semua peristiwa log yang tidak termasuk INFO
- **{ \$.eventType = "UpdateTrail" }**menampilkan semua peristiwa log JSON di mana nilai bidang jenis acara adalah UpdateTrail

Anda juga dapat menggunakan ekspresi reguler (regex) untuk memfilter:

- **%ERROR%**menggunakan regex untuk menampilkan semua peristiwa log yang terdiri dari kata kunci ERROR

- `{ $.names = %Steve% }` menggunakan regex untuk menampilkan peristiwa log JSON di mana Steve berada di properti "name"
- `[ w1 = %abc%, w2 ]` menggunakan regex untuk menampilkan peristiwa log yang dibatasi ruang di mana kata pertama adalah abc

Untuk informasi selengkapnya tentang sintaks pola, lihat [Filter sintaks pola](#).

6. (Optional) Untuk menyorot beberapa peristiwa log yang ditampilkan, masukkan istilah untuk dicari dan sorot di bawah Live Tail. Masukkan istilah sorotan satu per satu. Jika Anda menambahkan beberapa istilah untuk disorot, warna yang berbeda ditetapkan untuk mewakili setiap istilah. Indikator sorotan ditampilkan di sebelah kiri setiap peristiwa log yang berisi istilah yang ditentukan, dan juga muncul di bawah istilah itu sendiri ketika Anda memperluas peristiwa log di jendela utama untuk melihat peristiwa log lengkap.

Anda dapat menggunakan pemfilteran bersama dengan penyorotan untuk memecahkan masalah dengan cepat. Misalnya, Anda dapat memfilter peristiwa untuk menampilkan hanya peristiwa yang berisi `ERROR`, dan kemudian juga menyorot peristiwa yang berisi `404`.

7. Untuk memulai sesi, pilih Terapkan filter

Peristiwa log yang cocok mulai muncul di jendela. Informasi berikut juga ditampilkan:

- Timer menampilkan berapa lama sesi Live Tail telah aktif.
 - acara/detik menampilkan berapa banyak peristiwa log tertelan per detik yang cocok dengan filter yang telah Anda tetapkan.
 - Agar sesi tidak bergulir terlalu cepat karena banyak acara cocok dengan filter, CloudWatch Log mungkin hanya menampilkan beberapa peristiwa yang cocok. Jika ini terjadi, persentase peristiwa pencocokan yang ditampilkan di layar ditampilkan dalam % ditampilkan.
8. Untuk menjeda alur peristiwa untuk menyelidiki apa yang saat ini ditampilkan, klik di mana saja di jendela peristiwa.
 9. Selama sesi, Anda dapat menggunakan yang berikut ini untuk melihat detail lebih lanjut tentang setiap peristiwa log.
 - Untuk menampilkan seluruh teks untuk peristiwa log di jendela utama, pilih panah di sebelah peristiwa log itu.
 - Untuk menampilkan seluruh teks untuk peristiwa log di jendela samping, pilih kaca pembesar + di sebelah peristiwa log itu. Alur acara berhenti dan jendela samping muncul.

Menampilkan teks peristiwa log di jendela samping dapat berguna untuk membandingkan teksnya dengan peristiwa lain di jendela utama.

10. Untuk menghentikan sesi Live Tail, pilih Stop.
11. Untuk memulai ulang sesi, secara opsional gunakan panel Filter untuk memodifikasi kriteria pemfilteran, dan pilih Terapkan filter. Kemudian pilih Mulai.

Sentralisasi log lintas wilayah lintas akun

Sentralisasi data Amazon CloudWatch Logs berfungsi AWS Organizations untuk mengumpulkan data log dari beberapa akun anggota ke dalam satu repositori data menggunakan aturan sentralisasi lintas akun dan lintas wilayah. Anda menentukan aturan yang secara otomatis mereplikasi data log dari beberapa akun dan Wilayah AWS menjadi akun terpusat dalam organisasi Anda. Kemampuan ini menyederhanakan konsolidasi log untuk meningkatkan pemantauan, analisis, dan kepatuhan terpusat di seluruh infrastruktur Anda. AWS

CloudWatch Sentralisasi data log menawarkan fleksibilitas konfigurasi untuk memenuhi persyaratan operasional dan keamanan, seperti kemampuan untuk mengonfigurasi wilayah cadangan selama pengaturan aturan dalam akun tujuan untuk memastikan peningkatan ketahanan. Selain itu, Anda memiliki kontrol penuh atas perilaku enkripsi untuk grup log yang disalin dari akun sumber untuk menangani data yang awalnya dienkripsi dengan kunci KMS yang dikelola pelanggan.

Note

Fitur sentralisasi CloudWatch Log hanya memproses data log baru yang masuk ke akun sumber setelah Anda membuat aturan sentralisasi. Data log historis (log yang ada sebelum pembuatan aturan) tidak terpusat.

Konsep sentralisasi data

Sebelum Anda mulai menggunakan sentralisasi data CloudWatch Log, biasakan diri Anda dengan konsep-konsep berikut:

Aturan sentralisasi

Konfigurasi yang mendefinisikan bagaimana data log dari akun sumber dan wilayah direplikasi ke akun tujuan dan wilayah. Aturan menentukan kriteria sumber dan pengaturan tujuan.

Akun sumber

AWS Akun tempat data log berasal. Peristiwa log dari akun sumber direplikasi ke akun tujuan berdasarkan aturan sentralisasi yang Anda tetapkan.

Akun tujuan

AWS Akun tujuan tempat data log yang direplikasi disimpan. Akun ini berfungsi sebagai lokasi terpusat untuk analisis dan pemantauan log.

Wilayah Backup

Wilayah sekunder opsional dalam akun tujuan tempat data log dapat direplikasi untuk meningkatkan ketahanan dan tujuan pemulihan bencana.

Enkripsi dalam CloudWatch Log

Data grup log selalu dienkripsi di CloudWatch Log. Secara default, CloudWatch Log menggunakan enkripsi sisi server dengan Advanced Encryption Standard Galois/Counter Mode (AES-GCM) 256-bit untuk mengenkripsi data log saat istirahat. Sebagai alternatif, Anda dapat menggunakan Layanan Manajemen AWS Kunci untuk enkripsi ini. Untuk informasi selengkapnya, lihat [dokumentasi Enkripsi CloudWatch Log](#).

- Cara kerja enkripsi selama sentralisasi: Sentralisasi CloudWatch log secara aktif menyalin data log pada waktu konsumsi dari akun sumber ke akun tujuan. Selama proses ini, data Anda tetap dienkripsi dalam perjalanan menggunakan kunci layanan yang AWS dimiliki. Data saat istirahat di grup log sumber dan tujuan dienkripsi menggunakan metode enkripsi pilihan Anda (kunci KMS yang dikelola atau AWS dimiliki pelanggan). Jika Anda menggunakan kunci KMS yang dikelola pelanggan di grup log tujuan Anda, tambahkan tag `LogsManaged = true` ke kunci kms untuk layanan Sentralisasi untuk mengaksesnya.
- Ketika izin KMS diperlukan:
 - Jika Anda menggunakan kunci KMS yang dikelola pelanggan di akun sumber Anda, CloudWatch Log memerlukan [izin KMS dalam skenario](#) contoh berikut:
 - Manajemen Throughput: Ketika batas throughput sentralisasi tercapai, data log disimpan sementara dienkripsi dengan kunci KMS yang dikelola pelanggan Anda hingga bandwidth tersedia.
 - Perlindungan Data dan Redaksi: Ketika grup log sumber mengaktifkan kebijakan perlindungan data, CloudWatch Log memerlukan izin dekripsi untuk mengakses data log mentah untuk memusatkannya.

Important

Aturan sentralisasi dikelola oleh akun manajemen AWS Organizations atau administrator yang didelegasikan. Untuk mengecualikan grup log terenkripsi KMS yang dikelola

pelanggan dari sentralisasi, konfigurasi pengaturan aturan ke “Jangan memusatkan grup log yang dienkripsi dengan kunci KMS.” AWS

Menyiapkan sentralisasi log

Untuk menyiapkan Sentralisasi CloudWatch Log, Anda perlu mengonfigurasi aturan sentralisasi yang menentukan cara data log mengalir dari grup log di akun sumber untuk log grup di akun tujuan Anda.

Setelah aturan sentralisasi diaktifkan dan peristiwa log direplikasi ke akun tujuan, Anda dapat membuat filter metrik, langganan, dan akun pada grup log terpusat dengan kemampuan penyaringan yang ditingkatkan. Filter ini dapat menargetkan peristiwa log dari akun dan wilayah sumber tertentu, dan dapat memancarkan akun sumber dan informasi wilayah sebagai dimensi metrik. Untuk informasi selengkapnya, lihat [Membuat metrik dari peristiwa log dengan menggunakan filter](#).

Prasyarat

- AWS Organizations harus disiapkan dan akun sumber dan tujuan keduanya harus menjadi milik organisasi.
- Akses tepercaya harus diaktifkan untuk CloudWatch, akun manajemen dan akun tujuan sehingga memberikan akses ke data log.

Note

Disarankan untuk mengaktifkan akses tepercaya melalui konsol, yang secara otomatis membuat peran terkait layanan (SLR) yang diperlukan. Jika akses tepercaya diaktifkan melalui metode lain, peran terkait layanan perlu dibuat secara terpisah.

Menyesuaikan nama grup log tujuan

Saat membuat aturan sentralisasi, Anda dapat menyesuaikan bagaimana nama grup log tujuan disusun menggunakan atribut. Atribut ini secara otomatis diganti dengan nilai aktual saat grup log dibuat, memungkinkan Anda untuk mengatur log secara hierarkis di akun tujuan Anda. Secara default, hanya `${source.logGroup}` atribut yang digunakan, yang menggabungkan semua grup log dengan nama yang sama di akun tujuan. Jika variabel tidak dapat diselesaikan, ia mewarisi nilai dari variabel induknya dalam hierarki.

Atribut yang tersedia

Anda dapat menggunakan atribut berikut dalam pola nama grup log tujuan Anda:

Atribut nama grup log tujuan

Atribut	Deskripsi
<code>\${source.accountId}</code>	ID AWS akun tempat log berasal.
<code>\${source.region}</code>	Di Wilayah AWS mana log berasal.
<code>\${source.logGroup}</code>	Nama grup log asli dari akun sumber.
<code>\${source.org.id}</code>	AWS Organizations ID Anda dari akun sumber.
<code>\${source.org.ouId}</code>	ID unit organisasi dari akun sumber
<code>\${source.org.rootId}</code>	ID root organisasi
<code>\${source.org.path}</code>	Jalur organisasi lengkap dari akun ke root

Contoh

Pertahankan struktur grup log asli

Pola: `/centralized/${source.accountId}${source.logGroup}`

Hasil: `/centralized/123456789012/aws/lambda/my-function`

Atur berdasarkan akun dan wilayah

Pola: `/centralized/${source.accountId}/${source.region}`

Hasil: `/centralized/123456789012/us-east-1`

Mengatur berdasarkan struktur organisasi

Pola: `/logs/${source.org.id}/${source.org.ouId}/${source.accountId}`

Hasil: `/logs/o-abc123/ou-xyz-12345678/123456789012`

Struktur datar sederhana

Pola: `/centralized-logs`

Hasil: /centralized-logs

Praktik terbaik

- Sertakan ID akun sumber untuk dengan mudah mengidentifikasi log akun mana yang berasal.
- Sertakan wilayah sumber jika Anda memusatkan dari beberapa wilayah.
- Struktur nama grup log tujuan berada di bawah 512 karakter. CloudWatch Log memberlakukan panjang nama grup log maksimum 512 karakter.

Membuat aturan sentralisasi

Gunakan prosedur berikut untuk membuat aturan sentralisasi yang mereplikasi data log dari akun sumber ke akun tujuan Anda.

Untuk membuat aturan sentralisasi

1. Arahkan ke CloudWatch konsol di akun Manajemen atau Administrator Delegasi organisasi.
2. Pilih Pengaturan.
3. Arahkan ke tab Organisasi.
4. Pilih Konfigurasi aturan.
5. Tentukan detail sumber dengan menyetel bidang berikut, lalu pilih Berikutnya:
 - a. Nama aturan sentralisasi: Masukkan nama unik untuk aturan sentralisasi.
 - b. Akun sumber: Tentukan kriteria pemilihan sumber untuk memilih akun dari mana data telemetry akan dipusatkan. Kriteria seleksi dapat mencakup:
 - Daftar akun anggota dalam organisasi
 - Daftar unit organisasi dalam organisasi
 - Seluruh organisasi

Anda dapat memberikan kriteria pemilihan dalam dua mode:

- Builder: Pengalaman berbasis klik untuk menghasilkan kriteria pemilihan sumber
- Editor: Kotak teks bentuk bebas untuk memberikan kriteria pemilihan sumber

Sintaks yang didukung untuk kriteria pemilihan sumber:

- Kunci yang Didukung: OrganizationId OrganizationUnitId | AccountId | *
- Operator yang Didukung: = | IN | OR

c. Wilayah Sumber: Pilih daftar wilayah untuk mencari data telemetri yang akan dipusatkan.

6. Tentukan detail tujuan dengan menyetel bidang berikut, lalu pilih Berikutnya:

- a. Akun tujuan: Pilih akun di organisasi yang bertindak sebagai tujuan utama untuk data telemetri.
- b. Wilayah Tujuan: Pilih wilayah utama yang menyimpan salinan data telemetri terpusat.
- c. Wilayah Cadangan: Secara opsional pilih wilayah yang menyimpan salinan kedua dari data telemetri terpusat.

7. Tentukan data telemetri dengan menyetel bidang berikut, lalu pilih Berikutnya:

- a. Grup log: Pilih salah satu opsi berikut:
 - Semua grup log: Memusatkan log dari semua grup log di akun sumber.
 - Filter grup log: Memusatkan log dari subset grup log di akun sumber, sesuai dengan kriteria pemilihan. Anda dapat memberikan kriteria pemilihan dalam dua mode:
 - Builder: Pengalaman berbasis klik untuk menghasilkan kriteria seleksi
 - Editor: Kotak teks bentuk bebas untuk memberikan kriteria pemilihan

Ada dua kriteria pemilihan yang dapat Anda gunakan untuk memfilter log:

- Kriteria pemilihan grup log: Kriteria pemilihan yang menentukan grup log sumber mana yang akan dipusatkan.
 - Kunci yang Didukung: LogGroupName | *
 - Operator yang Didukung: = | != | DALAM | TIDAK DI | DAN | ATAU | SUKA | TIDAK SUKA
- Kriteria pemilihan sumber data: Kriteria seleksi yang menentukan sumber data mana yang akan dipusatkan.
 - Kunci yang Didukung: DataSourceName | DataSourceType
 - Operator yang Didukung: = | != | DALAM | TIDAK DI | DAN | ATAU | SUKA | TIDAK SUKA

Ketika kriteria pemilihan grup log dan kriteria pemilihan sumber data ditentukan, peristiwa log harus cocok dengan kedua kriteria untuk dipusatkan.

b. Grup Log Terenkripsi KMS

 Important

CloudWatch aturan sentralisasi akan gagal mengirimkan log dari akun sumber ke grup log tujuan jika Kunci KMS yang disediakan dalam aturan Sentralisasi tidak mengizinkan CloudWatch Log untuk menggunakannya. Jika Anda menggunakan kunci KMS yang dikelola pelanggan di grup log tujuan Anda, tambahkan tag `LogsManaged = true` ke kunci kms. Untuk informasi selengkapnya, lihat [Langkah 2: Tetapkan izin pada tombol KMS](#).

Pilih salah satu opsi berikut:

- Memusatkan grup log sumber yang dienkripsi dengan kunci KMS yang dikelola pelanggan menggunakan kunci KMS yang dikelola pelanggan khusus tujuan: Pusatkan peristiwa log dari grup log sumber yang dienkripsi dengan kunci KMS yang dikelola pelanggan ke grup log tujuan yang dienkripsi dengan kunci KMS yang dikelola pelanggan di akun tujuan.

Ketika pengaturan ini dipilih, Anda juga harus mengatur yang berikut:

- Kunci enkripsi tujuan ARN: ARN kunci KMS yang dikelola pelanggan di akun tujuan dan wilayah tujuan utama, untuk dikaitkan dengan grup log tujuan yang baru dibuat.
- Kunci enkripsi tujuan cadangan ARN (jika wilayah cadangan dipilih): ARN kunci KMS yang dikelola pelanggan di akun tujuan dan wilayah tujuan cadangan, untuk dikaitkan dengan grup log tujuan yang baru dibuat.
- Lewati sentralisasi ke grup log tujuan yang tidak terenkripsi (opsional): Jika grup log sudah ada tanpa kunci KMS yang dikelola pelanggan, CloudWatch tidak dapat memperbarui enkripsi. Pilih opsi ini untuk melewati sentralisasi peristiwa log dari grup log sumber yang dienkripsi dengan kunci KMS yang dikelola pelanggan ke grup log tujuan yang tidak terkait dengan kunci KMS yang dikelola pelanggan.
- Memusatkan grup log yang dienkripsi dengan kunci KMS yang dikelola pelanggan di akun tujuan dengan kunci KMS yang AWS dimiliki: Pusatkan peristiwa log dari grup log sumber yang dienkripsi dengan kunci KMS yang dikelola pelanggan ke grup log tujuan yang baru dibuat yang dienkripsi menggunakan kunci KMS yang dimiliki AWS.

- Jangan memusatkan grup log yang dienkrpsi dengan kunci KMS yang dikelola pelanggan: Lewati sentralisasi peristiwa log dari grup log sumber yang dienkrpsi dengan kunci KMS yang dikelola pelanggan.
8. Tinjau aturan sentralisasi, secara opsional lakukan pengeditan menit-menit terakhir, dan pilih Buat kebijakan Sentralisasi.

Memodifikasi aturan sentralisasi

Gunakan prosedur berikut untuk memodifikasi aturan sentralisasi yang ada.

Untuk memodifikasi aturan sentralisasi

1. Arahkan ke CloudWatch konsol di akun Manajemen atau Administrator Delegasi organisasi.
2. Pilih Pengaturan.
3. Arahkan ke tab Organisasi.
4. Pilih Kelola aturan.
5. Pilih aturan yang akan diperbarui dan pilih Edit.
6. Perbarui konfigurasi aturan sesuai kebutuhan, pilih Berikutnya untuk melanjutkan setiap langkah.
7. Pada Langkah 4, Tinjau dan konfigurasi, pilih Perbarui kebijakan sentralisasi.

Melihat aturan sentralisasi

Gunakan prosedur berikut untuk melihat rincian aturan sentralisasi yang ada.

Untuk melihat aturan sentralisasi

1. Arahkan ke CloudWatch konsol di akun Manajemen atau Administrator Delegasi organisasi.
2. Pilih Pengaturan.
3. Arahkan ke tab Organisasi.
4. Pilih Kelola aturan.
5. Lihat daftar semua aturan sentralisasi yang ada dan pilih nama aturan tertentu untuk melihat detailnya.

Menghapus aturan sentralisasi

Gunakan prosedur berikut untuk menghapus aturan sentralisasi yang ada.

Untuk menghapus aturan sentralisasi

1. Arahkan ke CloudWatch konsol di akun Manajemen atau Administrator Delegasi organisasi.
2. Pilih Pengaturan.
3. Arahkan ke tab Organisasi.
4. Pilih Kelola aturan.
5. Pilih aturan yang akan dihapus dan pilih Hapus.
6. Konfirmasikan penghapusan dan pilih Hapus.

Pemantauan dan pemecahan masalah aturan sentralisasi

Anda dapat memantau status dan kinerja aturan sentralisasi menggunakan CloudWatch metrik, konsol CloudWatch Log, dan AWS CloudTrail log. Ini membantu Anda memastikan bahwa data log direplikasi dengan sukses dan mengidentifikasi masalah apa pun dengan konfigurasi sentralisasi Anda.

CloudWatch Log menyediakan:

1. Aturan kesehatan per aturan Sentralisasi
 - a. Pilih Pengaturan.
 - b. Arahkan ke tab Organisasi.
 - c. Pilih Kelola aturan.
2. Log panggilan API dengan AWS CloudTrail
3. CloudWatch juga menerbitkan metrik untuk sentralisasi, termasuk peristiwa log yang direplikasi, kesalahan, dan pelambatan. Untuk informasi selengkapnya tentang metrik ini dan dimensinya, lihat [Metrik dan dimensi sentralisasi](#).

Status kesehatan aturan sentralisasi

Setiap aturan sentralisasi memiliki status kesehatan yang menunjukkan apakah itu beroperasi dengan benar. Anda dapat memeriksa kesehatan aturan melalui konsol atau menggunakan API secara terprogram.

Status kesehatan aturan meliputi:

- **HEALTHY**: Aturan beroperasi secara normal dan mereplikasi data log seperti yang dikonfigurasi
- **UNHEALTHY**: Aturan mengalami masalah dan mungkin tidak mereplikasi data dengan benar
- **PROVISIONING**: Sentralisasi untuk organisasi sedang dalam proses didirikan.

Ketika aturan ditandai sebagai TIDAK SEHAT, `FailureReason` bidang tersebut memberikan rincian tentang masalah spesifik yang perlu ditangani.

Memantau panggilan API sentralisasi dengan AWS CloudTrail

AWS CloudTrail log panggilan API yang dilakukan ke layanan sentralisasi, memungkinkan Anda melacak perubahan konfigurasi dan memecahkan masalah untuk akun yang menjadi anggota Anda. **AWS Organizations**

CloudTrail Peristiwa penting untuk sentralisasi meliputi:

- `CreateCentralizationRuleForOrganization`: Ketika aturan sentralisasi baru dibuat
- `UpdateCentralizationRuleForOrganization`: Ketika aturan yang ada dimodifikasi
- `DeleteCentralizationRuleForOrganization`: Ketika aturan dihapus
- `GetCentralizationRuleForOrganization`: Ketika rincian aturan diambil
- `ListCentralizationRulesForOrganization`: Ketika aturan terdaftar

Anda dapat menggunakan CloudTrail log untuk mengaudit perubahan konfigurasi sentralisasi dan menghubungkannya dengan masalah kinerja atau kegagalan replikasi.

Rekomendasi pemantauan

Untuk memastikan sentralisasi berfungsi dengan benar, kami sarankan untuk menyiapkan CloudWatch alarm pada metrik sentralisasi kunci yang kami jual ke Metrik. CloudWatch Pemantauan proaktif ini membantu Anda mendeteksi masalah sejak dini dan mempertahankan sentralisasi log yang andal di seluruh organisasi Anda.

Metrik utama untuk dipantau meliputi:

- `IncomingCopiedBytes`: Pantau volume data log yang berhasil direplikasi ke akun tujuan Anda. Penurunan mendadak atau tidak adanya metrik ini dapat mengindikasikan masalah sentralisasi.

- **CentralizationError**: Siapkan alarm untuk setiap kesalahan dalam proses sentralisasi untuk mengidentifikasi dan menyelesaikan masalah dengan cepat.
- **CentralizationThrottled**: Pantau peristiwa pelambatan yang dapat memengaruhi kinerja replikasi log.

Untuk daftar lengkap metrik sentralisasi yang tersedia dan dimensinya, lihat. [Metrik dan dimensi sentralisasi](#)

Jika log tidak terpusat seperti yang diharapkan, tinjau skenario umum berikut yang dapat mencegah sentralisasi log.

Data log historis

Fitur sentralisasi CloudWatch Log hanya memproses data log baru yang masuk ke akun sumber setelah Anda membuat aturan sentralisasi. Data log historis (log yang ada sebelum pembuatan aturan) tidak terpusat.

Izin kunci KMS

Aturan sentralisasi akan gagal mengirimkan log dari akun sumber ke grup log tujuan jika kunci KMS yang disediakan dalam aturan sentralisasi tidak mengizinkan CloudWatch Log untuk menggunakannya. Pastikan bahwa kebijakan kunci KMS memberikan izin yang diperlukan untuk Log. CloudWatch Untuk informasi selengkapnya, lihat [Langkah 2: Tetapkan izin pada tombol KMS](#).

Konfigurasi kunci KMS yang Dikelola Pelanggan

Jika Anda memilih Jangan memusatkan grup log yang dienkripsi dengan kunci KMS yang Dikelola Pelanggan selama pembuatan aturan, peristiwa log dari grup log sumber yang dienkripsi dengan kunci KMS yang Dikelola Pelanggan akan dilewati dan tidak terpusat.

Ketidakcocokan enkripsi tujuan

Jika grup log tujuan sudah ada dengan konfigurasi enkripsi KMS yang berbeda dari yang ditentukan oleh aturan sentralisasi, dan resolusi konflik diatur ke SKIP, catatan akan dihapus dan **DestinationEncryptionMismatch** kesalahan akan dipancarkan. Misalnya, ini terjadi ketika tujuan memiliki enkripsi default tetapi aturan menentukan kunci KMS yang dikelola pelanggan.

Akses tepercaya tidak diaktifkan

Akses tepercaya harus diaktifkan AWS Organizations untuk CloudWatch akun manajemen dan akun tujuan untuk menyediakan akses ke data log.

Kriteria pemilihan sumber

Verifikasi bahwa kriteria pemilihan sumber aturan sentralisasi Anda dikonfigurasi dengan benar:

- Akun dan wilayah: Pastikan akun sumber dan wilayah tempat log berasal disertakan dalam aturan. Grup log dari akun atau wilayah yang tidak ditentukan dalam aturan tidak akan terpusat.
- Filter grup log: Jika Anda mengonfigurasi filter grup log, hanya grup log yang cocok dengan kriteria yang ditentukan yang akan dipusatkan. Verifikasi bahwa kriteria pemilihan grup log Anda mencakup grup log yang Anda harapkan untuk dipusatkan.
- Keanggotaan organisasi: Akun sumber dan tujuan harus milik AWS Organizations organisasi yang sama. Akun di luar organisasi tidak dapat berpartisipasi dalam sentralisasi.

Batas kuota grup log tercapai

Jika akun tujuan telah mencapai batas kuota grup log, grup log baru tidak dapat dibuat untuk sentralisasi. Verifikasi bahwa akun tujuan memiliki kuota yang cukup untuk mengakomodasi grup log terpusat dari semua akun sumber. Anda dapat meminta kenaikan kuota jika diperlukan.

Batas panjang nama aliran log terlampaui

Nama aliran log memiliki batasan panjang maksimum. Saat sentralisasi mereplikasi aliran log ke akun tujuan, akhirnya ditambahkan ke nama aliran log. Jika nama aliran log yang dihasilkan melebihi panjang maksimum yang diizinkan, catatan akan dihapus dan `InvalidLogStream` kesalahan akan dipancarkan ke akun pelanggan.

Aturan status kesehatan

Periksa status kesehatan aturan sentralisasi di konsol atau menggunakan `GetCentralizationRuleForOrganization` API. Jika aturan ditandai sebagai TIDAK SEHAT, tinjau `FailureReason` bidang untuk detail spesifik tentang masalah tersebut.

Untuk mendiagnosis masalah sentralisasi, tinjau status kesehatan aturan sentralisasi di konsol, periksa CloudWatch metrik untuk kesalahan dan pelambatan, dan periksa AWS CloudTrail log untuk kegagalan panggilan API. Untuk informasi selengkapnya tentang metrik sentralisasi, lihat [Metrik dan dimensi sentralisasi](#)

Bekerja dengan grup log dan aliran log

Pengaliran log adalah urutan log acara yang berbagi sumber yang sama. Setiap sumber log yang terpisah di CloudWatch Log membentuk aliran log terpisah.

Grup log adalah grup log stream yang memiliki pengaturan retensi, pemantauan, dan kontrol akses yang sama. Anda dapat menentukan grup log dan menentukan log stream untuk dimasukkan ke dalam setiap grup. Tidak ada batas jumlah log stream yang dapat bergabung dalam satu grup log.

Untuk organisasi yang perlu mengkonsolidasikan data log dari beberapa akun dan wilayah, Anda dapat menggunakan Sentralisasi CloudWatch Log untuk secara otomatis mereplikasi grup log ke akun pusat. Untuk informasi selengkapnya, lihat [Sentralisasi log lintas wilayah lintas akun](#).

Anda dapat menggunakan prosedur di bagian ini untuk bekerja dengan grup log dan aliran log.

Buat grup log di CloudWatch Log

Saat Anda menginstal agen CloudWatch Log di instans Amazon EC2 menggunakan langkah-langkah di bagian sebelumnya dari Panduan Pengguna Amazon CloudWatch Logs, grup log dibuat sebagai bagian dari proses tersebut. Anda juga dapat membuat grup log langsung di CloudWatch konsol.

Untuk membuat grup log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Manajemen Log.
3. Pilih Actions (Tindakan), lalu pilih Create log group (Buat grup log).
4. Masukkan nama untuk grup log, lalu pilih Create log group (Buat grup log).

Tip

Anda dapat grup log favorit, serta dasbor dan alarm, dari menu Favorit dan terbaru di panel navigasi. Di bawah kolom Baru dikunjungi, arahkan kursor ke grup log yang ingin Anda sukai, dan pilih simbol bintang di sebelahnya.

Mengirim log ke grup log

CloudWatch Log secara otomatis menerima peristiwa log dari beberapa AWS layanan. Anda juga dapat mengirim peristiwa log lainnya ke CloudWatch Log menggunakan salah satu metode berikut:

- CloudWatch agen — CloudWatch Agen terpadu dapat mengirim metrik dan log ke CloudWatch Log. Untuk informasi tentang menginstal dan menggunakan CloudWatch agen, lihat [Mengumpulkan Metrik dan Log dari Instans Amazon EC2 dan Server Lokal dengan CloudWatch Agen](#) di Panduan Pengguna Amazon. CloudWatch
- AWS CLI [put-log-events](#)—Mengunggah kumpulan peristiwa log ke Log. CloudWatch
- Secara terprogram - [PutLogEvents](#) API memungkinkan Anda untuk mengunggah batch peristiwa log secara terprogram ke Log. CloudWatch

Lihat data log yang dikirim ke CloudWatch Log

Anda dapat melihat dan menggulir data log stream-by-stream berdasarkan yang dikirim ke CloudWatch Log oleh agen CloudWatch Log. Anda dapat menentukan rentang waktu untuk data log yang akan dilihat.

Untuk melihat data log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Manajemen Log.
3. Untuk Log Groups (Grup Log), pilih grup log untuk melihat pengaliran.
4. Dalam daftar grup log, pilih nama grup log yang ingin Anda lihat.
5. Dalam daftar pengaliran log, pilih nama pengaliran log yang ingin Anda lihat.
6. Untuk mengubah cara data log ditampilkan, lakukan salah satu hal berikut:
 - Untuk memperluas satu log acara, pilih tanda panah di samping log acara tersebut.
 - Untuk memperluas semua log acara dan melihatnya sebagai teks biasa, di atas daftar log acara, pilih Text (Teks).
 - Untuk memfilter log acara, masukkan filter pencarian yang diinginkan di kolom pencarian. Untuk informasi selengkapnya, lihat [Membuat metrik dari peristiwa log dengan menggunakan filter](#).

- Untuk melihat data log untuk tanggal dan rentang waktu yang ditentukan, di samping filter pencarian, pilih tanda panah di samping tanggal dan waktu. Untuk menentukan rentang tanggal dan waktu, pilih Absolute (Absolut). Untuk memilih jumlah menit, jam, hari, atau minggu yang telah ditentukan, pilih Relative (Relatif). Anda juga dapat beralih antara UTC dan zona waktu lokal.

Cari data log menggunakan pola filter

Anda dapat mencari data log Anda menggunakan [Filter sintaks pola untuk filter metrik, filter langganan, peristiwa log filter, dan Live Tail](#). Anda dapat mencari semua aliran log dalam grup log, atau dengan menggunakan AWS CLI Anda juga dapat mencari aliran log tertentu. Saat pencarian berjalan, akan dihasilkan halaman pertama data yang ditemukan dan token untuk mengambil halaman berikutnya dari data atau untuk melanjutkan pencarian. Jika tidak ada hasil yang dikembalikan, Anda dapat melanjutkan pencarian.

Anda dapat mengatur rentang waktu yang ingin Anda kuerikan untuk membatasi cakupan pencarian Anda. Anda bisa mulai dengan rentang yang lebih besar untuk melihat tempat garis log yang Anda inginkan, dan kemudian mempersingkat rentang waktu untuk membuat cakupan tampilan log dalam rentang waktu yang Anda inginkan.

Anda juga dapat beralih langsung dari metrik yang diekstraksi log ke log yang sesuai.

Jika Anda masuk ke akun yang disiapkan sebagai akun pemantauan dalam pengamatan CloudWatch lintas akun, Anda dapat mencari dan memfilter peristiwa log dari akun sumber yang ditautkan ke akun pemantauan ini. Untuk informasi lebih lanjut, lihat [CloudWatch observabilitas lintas akun](#).

Cari entri log menggunakan konsol

Anda dapat mencari entri log yang memenuhi kriteria tertentu menggunakan konsol.

Untuk mencari log menggunakan konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Manajemen Log.
3. Untuk Grup Log, pilih nama grup log yang berisi log stream yang akan dicari.
4. Untuk Log Stream, pilih nama log stream yang akan dicari.
5. Di bawah Log events (Log acara), masukkan sintaks filter yang akan digunakan.

Untuk mencari semua entri log untuk rentang waktu menggunakan konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Manajemen Log.
3. Untuk Log Groups (Grup Log), pilih nama grup log yang berisi pengaliran log yang akan dicari.
4. Pilih Search log group (Cari grup log).
5. Untuk Log events (Log acara), pilih tanggal dan rentang waktu, dan masukkan sintaks filter.

Cari entri log menggunakan AWS CLI

Anda dapat mencari entri log yang memenuhi kriteria tertentu menggunakan AWS CLI

Untuk mencari entri log menggunakan AWS CLI

Pada prompt perintah, jalankan [filter-log-events](#) perintah berikut. Gunakan `--filter-pattern` untuk membatasi hasil ke pola filter yang ditentukan dan `--log-stream-names` untuk membatasi hasil ke pengaliran log tertentu.

```
aws logs filter-log-events --log-group-name my-group [--log-stream-  
names LIST_OF_STREAMS_TO_SEARCH] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

Untuk mencari entri log selama rentang waktu tertentu menggunakan AWS CLI

Pada prompt perintah, jalankan [filter-log-events](#) perintah berikut:

```
aws logs filter-log-events --log-group-name my-group [--log-stream-  
names LIST_OF_STREAMS_TO_SEARCH] [--start-time 1482197400000] [--end-  
time 1482217558365] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

Pivot dari metrik ke log

Anda bisa beralih ke entri log tertentu dari bagian lain dari konsol.

Untuk beralih dari widget dasbor ke log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, silakan pilih Dasbor.
3. Pilih dasbor.

4. Di widget, pilih ikon View logs (Lihat log), lalu pilih View logs in this time range (Lihat log dalam rentang waktu ini). Jika terdapat lebih dari satu filter metrik, pilih salah satu dari daftar. Jika ada lebih banyak filter metrik dari yang dapat kita tampilkan dalam daftar, pilih More metric filters (Lebih banyak filter metrik) dan pilih atau cari filter metrik.

Untuk beralih dari metrik ke log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, silakan pilih Metrik.
3. Di bidang pencarian di tab All metrics, ketik nama metrik dan tekan Enter.
4. Pilih satu atau beberapa metrik dari hasil pencarian Anda.
5. Pilih Actions (Tindakan), View logs (Lihat log). Jika terdapat lebih dari satu filter metrik, pilih salah satu dari daftar. Jika ada lebih banyak filter metrik dari yang dapat kita tampilkan dalam daftar, pilih More metric filters (Lebih banyak filter metrik) dan pilih atau cari filter metrik.

Pemecahan masalah

Pencarian membutuhkan waktu terlalu lama untuk diselesaikan

Jika Anda memiliki banyak data log, pencarian mungkin memerlukan waktu lama untuk diselesaikan. Untuk mempercepat pencarian, Anda dapat melakukan hal berikut:

- Jika Anda menggunakan AWS CLI, Anda dapat membatasi pencarian hanya pada aliran log yang Anda minati. Misalnya, jika grup log Anda memiliki 1000 aliran log, tetapi Anda hanya ingin melihat tiga aliran log yang Anda tahu relevan, Anda dapat menggunakan AWS CLI untuk membatasi pencarian Anda hanya pada tiga aliran log dalam grup log.
- Gunakan rentang waktu yang lebih pendek dan lebih terperinci, yang mengurangi jumlah data yang akan dicari dan mempercepat kueri.

Ubah penyimpanan data log di CloudWatch Log

Secara default, data log disimpan di CloudWatch Log tanpa batas waktu. Namun, Anda dapat mengonfigurasi berapa lama data log disimpan dalam grup log. Data apa pun yang lebih lama dari pengaturan retensi saat ini akan dihapus. Anda dapat mengubah retensi log untuk setiap grup log kapan saja.

 Note

CloudWatch Logs tidak segera menghapus peristiwa log ketika mereka mencapai pengaturan retensi mereka. Biasanya memakan waktu hingga 72 jam setelah itu sebelum peristiwa log dihapus, tetapi dalam situasi yang jarang terjadi mungkin memakan waktu lebih lama.

Ini berarti bahwa jika Anda mengubah grup log untuk memiliki pengaturan retensi yang lebih lama ketika berisi peristiwa log yang melewati tanggal kedaluwarsa, tetapi belum benar-benar dihapus, peristiwa log tersebut akan memakan waktu hingga 72 jam untuk dihapus setelah tanggal penyimpanan baru tercapai. Untuk memastikan bahwa data log dihapus secara permanen, simpan grup log pada pengaturan retensi yang lebih rendah hingga 72 jam berlalu setelah akhir periode penyimpanan sebelumnya, atau Anda telah mengonfirmasi bahwa peristiwa log lama akan dihapus.

Ketika peristiwa log mencapai pengaturan retensi mereka, mereka ditandai untuk dihapus. Setelah ditandai untuk dihapus, mereka tidak menambah biaya penyimpanan arsip Anda lagi, bahkan jika mereka tidak benar-benar dihapus sampai nanti. Peristiwa log yang ditandai untuk dihapus ini juga tidak disertakan saat Anda menggunakan API untuk mengambil `storedBytes` nilai guna melihat berapa banyak byte yang disimpan grup log.

Untuk mengubah pengaturan retensi log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Temukan grup log yang akan diperbarui.
4. Di kolom Retensi untuk grup log tersebut, pilih setelan retensi saat ini, seperti Jangan Pernah Kedaluwarsa.
5. Di Setelan retensi, untuk peristiwa kedaluwarsa setelahnya, pilih nilai retensi log, lalu pilih Simpan.

Melindungi grup log dari penghapusan

Anda dapat mengaktifkan perlindungan penghapusan secara opsional untuk mencegah penghapusan grup log penting secara tidak sengaja. Untuk informasi rinci tentang perlindungan penghapusan, lihat. [Melindungi grup log dari penghapusan](#)

Melindungi grup log dari penghapusan

Mengaktifkan perlindungan penghapusan

Anda dapat mengaktifkan perlindungan penghapusan saat membuat grup log baru atau pada grup log yang ada. Selama pembuatan grup log, pilih “Perlindungan penghapusan yang diaktifkan” atau dengan meneruskan parameter. `--deletion-protection-enabled` Secara default, perlindungan penghapusan tidak diaktifkan.

Untuk mengaktifkan atau menonaktifkan perlindungan penghapusan pada grup log yang ada (konsol)

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Manajemen Log.
3. Pilih grup log yang ingin Anda lindungi.
4. Pilih Tindakan, Edit perlindungan penghapusan.
5. Di kotak dialog, tinjau lalu kirimkan perubahan.

Jika menggunakan AWS CLI, untuk mengaktifkan perlindungan penghapusan pada grup log yang ada:

```
aws logs put-log-group-deletion-protection \  
--log-group-identifier "/my-application/logs" \  
--deletion-protection-enabled
```

Untuk menghapus perlindungan penghapusan pada grup log yang ada:

```
aws logs put-log-group-deletion-protection \  
--log-group-identifier "/my-application/logs" \  
--no-deletion-protection-enabled
```

Penanganan kesalahan

Jika Anda mencoba menghapus grup log dengan perlindungan penghapusan diaktifkan, Anda menerima pesan `ValidationException` dengan: “Tidak dapat menghapus grup log dengan perlindungan penghapusan diaktifkan. Nonaktifkan perlindungan penghapusan terlebih dahulu.”

Tandai grup log di Amazon CloudWatch Logs

Anda dapat menetapkan metadata Anda sendiri ke grup log yang Anda buat di Amazon CloudWatch Logs dalam bentuk tag. Tanda adalah pasangan nilai-kunci yang Anda tetapkan untuk grup log. Menggunakan tag adalah cara sederhana namun ampuh untuk mengelola AWS sumber daya dan mengatur data, termasuk data penagihan.

Note

Anda dapat menggunakan tag untuk mengontrol akses ke sumber CloudWatch Log, termasuk grup log dan tujuan. Akses ke aliran log dikontrol pada tingkat grup log, karena hubungan hierarkis antara grup log dan aliran log. Untuk informasi selengkapnya tentang penggunaan tanda untuk mengendalikan akses, lihat [Mengendalikan akses ke sumber daya Amazon Web Services menggunakan tanda](#).

Daftar Isi

- [Dasar-dasar tag](#)
- [Melacak biaya menggunakan penandaan](#)
- [Pembatasan tag](#)
- [Menandai grup log menggunakan AWS CLI](#)
- [Menandai grup log menggunakan API CloudWatch Log](#)

Dasar-dasar tag

Anda menggunakan AWS CloudFormation AWS CLI, atau CloudWatch Logs API untuk menyelesaikan tugas-tugas berikut:

- Menambahkan tanda ke grup log saat Anda membuatnya.
- Menambahkan tanda ke grup log yang sudah ada.
- Mendaftar tanda untuk grup log.
- Menghapus tanda dari grup log.

Anda dapat menggunakan tanda untuk mengkategorikan grup log Anda. Misalnya, Anda dapat mengkategorikannya berdasarkan tujuan, pemilik, atau lingkungan. Karena Anda menentukan kunci

dan nilai untuk setiap tanda, Anda dapat membuat serangkaian kategori khusus untuk memenuhi kebutuhan spesifik Anda. Misalnya, Anda dapat menentukan satu set tanda yang membantu Anda melacak grup log berdasarkan pemilik dan aplikasi terkait. Berikut adalah beberapa contoh tanda:

- Proyek: Nama proyek
- Pemilik: Nama
- Tujuan: Pengujian beban
- Aplikasi: Nama aplikasi
- Lingkungan: Produksi

Melacak biaya menggunakan penandaan

Anda dapat menggunakan tag untuk mengkategorikan dan melacak biaya Anda AWS . Saat Anda menerapkan tag ke AWS sumber daya Anda, termasuk grup log, laporan alokasi AWS biaya Anda mencakup penggunaan dan biaya yang dikumpulkan berdasarkan tag. Anda dapat menerapkan tag yang mewakili kategori bisnis (seperti pusat biaya, nama aplikasi, atau pemilik) untuk mengatur biaya Anda di berbagai layanan. Untuk informasi selengkapnya, lihat [Menggunakan Tanda Alokasi Biaya untuk Laporan Penagihan Khusus](#) dalam Panduan Pengguna AWS Billing .

Pembatasan tag

Batasan berikut berlaku untuk tanda.

Batasan dasar

- Jumlah maksimum tanda per grup log adalah 50.
- Kunci dan nilai tag peka huruf besar dan kecil.
- Anda tidak dapat mengubah atau mengedit tanda untuk grup log yang dihapus.

Batasan kunci tanda

- Setiap kunci tanda harus unik. Jika Anda menambahkan tanda dengan kunci yang sudah digunakan, tanda baru akan menimpa pasangan nilai-kunci yang sudah ada.
- Anda tidak dapat memulai kunci tag aws : karena awalan ini dicadangkan untuk digunakan oleh AWS. AWS membuat tag yang dimulai dengan awalan ini atas nama Anda, tetapi Anda tidak dapat mengedit atau menghapusnya.

- Kunci tanda harus memiliki panjang antara 1 dan 128 karakter Unicode.
- Kunci tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan karakter khusus berikut: `_ . / = + - @`.

Batasan nilai tanda

- Panjang nilai tanda harus antara 0 dan 255 karakter Unicode.
- Nilai tanda dapat kosong. Jika tidak, nilai tanda harus terdiri dari karakter berikut: huruf Unicode, digit, spasi, dan salah satu karakter khusus berikut: `_ . / = + - @`.

Menandai grup log menggunakan AWS CLI

Anda dapat menambahkan, mendaftar, dan menghapus tanda menggunakan AWS CLI. Untuk contoh, lihat dokumentasi berikut:

[create-log-group](#)

Membuat grup log. Anda dapat secara opsional menambahkan tanda ketika membuat grup log.

[tag-sumber daya](#)

Menetapkan satu atau beberapa tag (pasangan kunci-nilai) ke sumber Log yang ditentukan CloudWatch .

[list-tags-for-resource](#)

Menampilkan tag yang terkait dengan sumber daya CloudWatch Log.

[untag-sumber daya](#)

Menghapus satu atau beberapa tag dari sumber CloudWatch Log yang ditentukan.

Menandai grup log menggunakan API CloudWatch Log

Anda dapat menambahkan, membuat daftar, dan menghapus tag menggunakan API CloudWatch Log. Untuk contoh, lihat dokumentasi berikut:

[CreateLogGroup](#)

Membuat grup log. Anda dapat secara opsional menambahkan tanda ketika membuat grup log.

[TagResource](#)

Menetapkan satu atau beberapa tag (pasangan kunci-nilai) ke sumber Log yang ditentukan CloudWatch .

[ListTagsForResource](#)

Menampilkan tag yang terkait dengan sumber daya CloudWatch Log.

[UntagResource](#)

Menghapus satu atau beberapa tag dari sumber CloudWatch Log yang ditentukan.

Enkripsi data log di CloudWatch Log menggunakan AWS Key Management Service

Data grup log selalu dienkripsi di CloudWatch Log. Secara default, CloudWatch Log menggunakan enkripsi sisi server dengan Advanced Encryption Standard Galois/Counter Mode (AES-GCM) 256-bit untuk mengenkripsi data log saat istirahat. Sebagai alternatif, Anda dapat menggunakan AWS Key Management Service enkripsi ini. Jika Anda melakukannya, enkripsi dilakukan dengan menggunakan AWS KMS kunci. Penggunaan enkripsi AWS KMS diaktifkan pada tingkat grup log, dengan mengaitkan kunci KMS dengan grup log, baik saat Anda membuat grup log atau setelah ada.

Important

CloudWatch Log sekarang mendukung konteks enkripsi, menggunakan `kms:EncryptionContext:aws:logs:arn` sebagai kunci dan ARN dari grup log sebagai nilai untuk kunci itu. Jika Anda memiliki grup log yang telah dienkripsi dengan kunci KMS, dan Anda ingin membatasi kunci yang akan digunakan dengan satu akun dan grup log, Anda harus menetapkan kunci KMS baru yang menyertakan kondisi dalam kebijakan IAM. Untuk informasi selengkapnya, lihat [AWS KMS kunci dan konteks enkripsi](#).

Important

CloudWatch Log sekarang mendukung `kms:ViaService` yang memungkinkan log untuk melakukan AWS KMS panggilan atas nama Anda. Anda harus menambahkan ini ke peran Anda yang memanggil CloudWatch Log baik dalam Kebijakan Kunci Anda atau di IAM. Untuk informasi lebih lanjut, lihat [kms: ViaService](#).

Setelah Anda mengaitkan kunci KMS dengan grup log, semua data yang baru dicerna untuk grup log dienkripsi menggunakan kunci ini. Data ini disimpan dalam format terenkripsi selama periode retensi. CloudWatch Log mendekripsi data ini setiap kali diminta. CloudWatch Log harus memiliki izin untuk kunci KMS setiap kali data terenkripsi diminta.

Jika Anda kemudian memisahkan kunci KMS dari grup CloudWatch log, Log mengenkripsi data yang baru dicerna menggunakan metode enkripsi default Log. CloudWatch Semua data yang dicerna sebelumnya yang dienkripsi dengan kunci KMS tetap dienkripsi dengan kunci KMS. CloudWatch Log masih dapat mengembalikan data tersebut setelah kunci KMS dipisahkan, karena CloudWatch Log masih dapat terus mereferensikan kunci tersebut. Namun, jika kuncinya kemudian dinonaktifkan, maka CloudWatch Log tidak dapat membaca log yang dienkripsi dengan kunci itu.

Important

CloudWatch Log hanya mendukung kunci KMS simetris. Jangan gunakan kunci asimetris untuk mengenkripsi data dalam grup log Anda. Untuk informasi selengkapnya, lihat [Menggunakan Kunci Simetris dan Asimetris](#).

Batas

- Untuk melakukan langkah-langkah berikut, Anda harus memiliki izin berikut: `kms:CreateKey`, `kms:GetKeyPolicy`, dan `kms:PutKeyPolicy`.
- Setelah Anda mengaitkan atau memisahkan kunci dari grup log, diperlukan waktu hingga lima menit agar operasi diterapkan.
- Jika Anda mencabut akses CloudWatch Log ke kunci terkait atau menghapus kunci KMS terkait, data terenkripsi Anda di CloudWatch Log tidak dapat diambil lagi.
- Anda tidak dapat mengaitkan kunci KMS dengan grup log yang ada menggunakan CloudWatch konsol.

Langkah 1: Buat AWS KMS kunci

Untuk membuat kunci KMS, gunakan perintah [create-key](#) berikut:

```
aws kms create-key
```


Output berisi ID kunci dan Amazon Resource Name (ARN) dari kunci. Berikut ini adalah output contoh:

```
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1478910250.94,
    "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-
e40cb0d29f59",
    "AWSAccountId": "123456789012",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

Langkah 2: Tetapkan izin pada tombol KMS

Secara default, semua AWS KMS kunci bersifat pribadi. Hanya pemilik sumber daya yang dapat menggunakannya untuk mengenkripsi dan mendekripsi data. Namun, pemilik sumber daya dapat memberikan izin untuk mengakses kunci KMS ke pengguna dan sumber daya lain. Dengan langkah ini, Anda memberikan izin utama layanan CloudWatch Log dan peran pemanggil untuk menggunakan kunci. Prinsipal layanan ini harus berada di AWS Wilayah yang sama di mana kunci KMS disimpan.

Sebagai praktik terbaik, kami menyarankan Anda membatasi penggunaan kunci KMS hanya untuk AWS akun atau grup log yang Anda tentukan.

Pertama, simpan kebijakan default untuk kunci KMS Anda seperti `policy.json` menggunakan [get-key-policy](#) perintah berikut:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

Buka file `policy.json` di editor teks dan tambahkan bagian dalam huruf tebal dari salah satu pernyataan berikut. Pisahkan pernyataan yang ada dari pernyataan baru dengan koma. Pernyataan ini menggunakan Condition bagian untuk meningkatkan keamanan AWS KMS kunci. Untuk informasi selengkapnya, lihat [AWS KMS kunci dan konteks enkripsi](#).

Bagian Condition dalam contoh ini membatasi kunci pada satu ARN grup log.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-east-1:111122223333:log-group:log-group-name"
        }
      }
    }
  ]
}
```

```
    ]
  }
}
```

Bagian Condition dalam contoh ini membatasi penggunaan kunci AWS KMS pada akun tertentu, tetapi dapat digunakan untuk grup log apa pun.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-east-1:123456789012:*"
        }
      }
    }
  ]
}
```

```
]
}
```

Langkah 3: Tetapkan izin pada prinsipal IAM panggilan

Tambahkan izin ke prinsipal IAM (pengguna atau peran) yang akan memanggil CloudWatch Log. APIs Izin yang diperlukan tergantung pada operasi mana yang harus dilakukan oleh prinsipal. Anda dapat menambahkan izin ini di kebijakan AWS KMS kunci atau melalui IAM pada peran itu sendiri. CloudWatch Log digunakan `kms:ViaService` untuk melakukan panggilan ke AWS KMS atas nama pelanggan. Untuk informasi lebih lanjut, lihat [kms: ViaService](#).

Izin untuk mengaitkan kunci KMS dengan grup log

Prinsipal IAM yang memanggil `CreateLogGroup` dengan `kmsKeyId` parameter, atau panggilan `AssociateKmsKey`, harus memiliki `kms:DescribeKey` izin pada kunci KMS yang ditentukan. Jika penelepon tidak memiliki izin ini, panggilan API akan gagal dengan `fileAccessDeniedException`.

Contoh pernyataan kebijakan kunci berikut memberikan izin minimum yang diperlukan untuk mengaitkan kunci KMS dengan grup log:

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account_id:role/role_name"
  },
  "Action": [
    "kms:Describe*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "logs.region.amazonaws.com"
      ]
    }
  }
}
```

Izin untuk membaca dan menulis data log terenkripsi

Jika prinsipal IAM juga perlu membaca atau menulis data log terenkripsi (misalnya, memanggil, `PutLogEvents`, `GetLogEventsFilterLogEvents`, atau `StartQuery` pada grup log yang dienkripsi dengan kunci KMS yang dikelola pelanggan), prinsipal memerlukan izin tambahan. AWS KMS Contoh pernyataan kebijakan kunci berikut memberikan set lengkap izin yang diperlukan untuk mengaitkan kunci dan membaca atau menulis data log terenkripsi:

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account_id:role/role_name"
  },
  "Action": [
    "kms:Encrypt",
    "kms:ReEncrypt*",
    "kms:Decrypt",
    "kms:GenerateDataKey*",
    "kms:Describe*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "logs.region.amazonaws.com"
      ]
    }
  }
}
```

Sebagai praktik terbaik, lingkup kebijakan hanya pada peran yang akan berinteraksi dengan grup log AWS KMS terenkripsi.

Atau, untuk memberikan set lengkap izin untuk mengaitkan kunci dan membaca atau menulis data log terenkripsi melalui IAM, Anda dapat menambahkan kebijakan berikut ke peran pemanggil. Ini dapat ditambahkan ke kebijakan peran yang ada atau dilampirkan ke peran sebagai kebijakan terpisah tambahan. Jika Anda menggunakan metode ini, sebagai praktik terbaik, lingkup kebijakan hanya untuk AWS KMS kunci yang akan digunakan untuk enkripsi log. Untuk informasi selengkapnya, lihat [Mengedit kebijakan IAM](#).

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:ReEncrypt*",
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:Describe*"
      ],
      "Condition": {
        "StringEquals": {
          "kms:ViaService": [
            "logs.us-east-1.amazonaws.com"
          ]
        }
      },
      "Resource": "arn:aws:kms:us-east-1:444455556666:key/key_id"
    }
  ]
}
```

Terakhir, tambahkan kebijakan yang diperbarui menggunakan [put-key-policy](#) perintah berikut:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Langkah 4: Kaitkan kunci KMS dengan grup log

Anda dapat mengaitkan kunci KMS dengan grup log saat Anda membuatnya atau setelah itu ada.

Untuk mengetahui apakah grup log sudah memiliki kunci KMS yang terkait, gunakan [describe-log-groups](#) perintah berikut:

```
aws logs describe-log-groups --log-group-name-prefix "log-group-name-prefix"
```

Jika outputnya mencakup bidang `kmsKeyId`, grup log terkait dengan kunci yang ditampilkan untuk nilai bidang tersebut.

Untuk mengaitkan kunci KMS dengan grup log saat Anda membuatnya

Gunakan [create-log-group](#) perintah sebagai berikut:

```
aws logs create-log-group --log-group-name my-log-group --kms-key-id "key-arn"
```

Untuk mengaitkan kunci KMS dengan grup log yang ada

Gunakan [associate-kms-key](#) perintah sebagai berikut:

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id "key-arn"
```

Langkah 5: Pisahkan kunci dari grup log

Untuk memisahkan kunci KMS yang terkait dengan grup log, gunakan perintah berikut: [disassociate-kms-key](#)

```
aws logs disassociate-kms-key --log-group-name my-log-group
```

AWS KMS kunci dan konteks enkripsi

Untuk meningkatkan keamanan AWS Key Management Service kunci Anda dan grup log terenkripsi Anda, CloudWatch Log sekarang menempatkan grup log ARNs sebagai bagian dari konteks enkripsi yang digunakan untuk mengenkripsi data log Anda. Konteks enkripsi adalah seperangkat pasangan nilai-kunci yang digunakan sebagai data terautentikasi tambahan. Konteks enkripsi memungkinkan Anda menggunakan kondisi kebijakan IAM untuk membatasi akses ke AWS KMS kunci Anda berdasarkan AWS akun dan grup log. Untuk informasi selengkapnya, lihat [Konteks enkripsi](#) and [Elemen Kebijakan JSON IAM: Syarat](#).

Kami menyarankan Anda menggunakan kunci KMS yang berbeda untuk setiap grup log terenkripsi Anda.

Jika Anda memiliki grup log yang Anda enkripsi sebelumnya dan sekarang ingin mengubah grup log untuk menggunakan kunci KMS baru yang hanya berfungsi untuk grup log itu, ikuti langkah-langkah ini.

Untuk mengonversi grup log terenkripsi untuk menggunakan kunci KMS dengan kebijakan yang membatasi grup log tersebut

1. Masukkan perintah berikut untuk menemukan ARN dari kunci grup log saat ini:

```
aws logs describe-log-groups
```

Outputnya mencakup baris berikut. Perhatikan ARN. Anda perlu menggunakannya di langkah 7.

```
...  
"kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/01234567-89ab-  
cdef-0123-456789abcdef"  
...
```

2. Masukkan perintah berikut untuk membuat kunci KMS baru:

```
aws kms create-key
```

3. Masukkan perintah berikut untuk menyimpan kebijakan kunci baru ke file `policy.json`:

```
aws kms get-key-policy --key-id new-key-id --policy-name default --output text > ./  
policy.json
```

4. Gunakan editor teks untuk membuka `policy.json` dan menambahkan ekspresi Condition ke kebijakan:

JSON

```
{  
  "Version": "2012-10-17",  
  "Id": "key-default-1",  
  "Statement": [  
    {  
      "Sid": "Enable IAM User Permissions",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::111122223333:root"  
      },  
      "Action": "kms:*",  
      "Resource": "*"   
    },  
  ],  
}
```



```

    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-
east-1:111122223333:log-group:LOG-GROUP-NAME"
        }
      }
    }
  ]
}

```

5. Masukkan perintah berikut untuk menambahkan kebijakan yang diperbarui ke kunci KMS baru:

```
aws kms put-key-policy --key-id new-key-ARN --policy-name default --policy file://policy.json
```

6. Masukkan perintah berikut untuk mengaitkan kebijakan dengan grup log Anda:

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id new-key-ARN
```

CloudWatch Log sekarang mengenkripsi semua data baru menggunakan kunci baru.

7. Selanjutnya, cabut semua izin kecuali Decrypt dari kunci lama. Pertama, masukkan perintah berikut untuk mengambil kebijakan lama:

```
aws kms get-key-policy --key-id old-key-ARN --policy-name default --output text
> ./policy.json
```

8. Gunakan editor teks untuk membuka `policy.json` dan hapus semua nilai dari daftar Action, kecuali untuk `kms:Decrypt`

JSON

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.region.amazonaws.com"
      },
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```

9. Masukkan perintah berikut untuk menambahkan kebijakan yang diperbarui ke kunci lama:

```
aws kms put-key-policy --key-id old-key-ARN --policy-name default --policy file://
policy.json
```

Membantu melindungi data log sensitif dengan masking

Anda dapat membantu melindungi data sensitif yang dicerna oleh CloudWatch Log dengan menggunakan kebijakan perlindungan data grup log. Kebijakan ini memungkinkan Anda mengaudit dan menutupi data sensitif yang muncul di peristiwa log yang dicerna oleh grup log di akun Anda.

Saat Anda membuat kebijakan perlindungan data, maka secara default, data sensitif yang cocok dengan pengidentifikasi data yang Anda pilih akan disembunyikan di semua titik keluar, termasuk Wawasan CloudWatch Log, filter metrik, dan filter langganan. Hanya pengguna yang memiliki izin `Logs:Unmask IAM` yang dapat melihat data yang dibuka kedoknya.

Anda dapat membuat kebijakan perlindungan data untuk semua grup log di akun Anda, dan Anda juga dapat membuat kebijakan perlindungan data untuk grup log individual. Saat Anda membuat kebijakan untuk seluruh akun, kebijakan tersebut berlaku untuk grup log dan grup log yang sudah ada yang dibuat di masa mendatang.

Jika Anda membuat kebijakan perlindungan data untuk seluruh akun Anda dan Anda juga membuat kebijakan untuk satu grup log, kedua kebijakan tersebut berlaku untuk grup log tersebut. Semua pengidentifikasi data terkelola yang ditentukan dalam salah satu kebijakan diaudit dan disamarkan dalam grup log tersebut.

 Note

Menyembunyikan data sensitif didukung untuk grup log di kelas log Akses Standar dan Jarang. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

Setiap grup log hanya dapat memiliki satu kebijakan perlindungan data tingkat grup log, tetapi kebijakan tersebut dapat menentukan banyak pengidentifikasi data terkelola untuk diaudit dan disembunyikan. Batas untuk kebijakan perlindungan data adalah 30.720 karakter.

 Important

Data sensitif terdeteksi dan disamarkan saat tertelan ke dalam grup log. Saat Anda menetapkan kebijakan perlindungan data, peristiwa log yang dicerna ke grup log sebelum waktu tersebut tidak disamarkan.

CloudWatch Log mendukung banyak pengidentifikasi data terkelola, yang menawarkan tipe data yang telah dikonfigurasi sebelumnya yang dapat Anda pilih untuk melindungi data keuangan, informasi kesehatan pribadi (PHI), dan informasi identitas pribadi (PII). CloudWatch Perlindungan data log memungkinkan Anda memanfaatkan pencocokan pola dan model pembelajaran mesin untuk mendeteksi data sensitif. Untuk beberapa jenis pengidentifikasi data terkelola, deteksi tergantung pada juga menemukan kata kunci tertentu yang berdekatan dengan data sensitif. Anda juga dapat

menggunakan pengidentifikasi data khusus untuk membuat pengidentifikasi data yang disesuaikan dengan kasus penggunaan spesifik Anda.

Metrik dipancarkan CloudWatch saat data sensitif terdeteksi yang cocok dengan pengidentifikasi data yang Anda pilih. Ini adalah `LogEventsWithFindings` metrik dan dipancarkan di ruang nama `AWS/log`. Anda dapat menggunakan metrik ini untuk membuat CloudWatch alarm, dan Anda dapat memvisualisasikannya dalam grafik dan dasbor. Metrik yang dipancarkan oleh perlindungan data adalah metrik yang dijual dan tidak dikenai biaya. Untuk informasi selengkapnya tentang metrik yang dikirimkan oleh CloudWatch Log CloudWatch, lihat [Pemantauan dengan CloudWatch metrik](#).

Setiap pengidentifikasi data terkelola dirancang untuk mendeteksi jenis data sensitif tertentu, seperti nomor kartu kredit, kunci akses AWS rahasia, atau nomor paspor untuk negara atau wilayah tertentu. Saat membuat kebijakan perlindungan data, Anda dapat mengonfigurasinya untuk menggunakan pengidentifikasi ini untuk menganalisis log yang dicerna oleh grup log, dan mengambil tindakan saat terdeteksi.

CloudWatch Perlindungan data log dapat mendeteksi kategori data sensitif berikut dengan menggunakan pengidentifikasi data terkelola:

- Kredensial, seperti kunci pribadi atau kunci akses AWS rahasia
- Informasi keuangan, seperti nomor kartu kredit
- Informasi Identifikasi Pribadi (PII) seperti SIM atau nomor jaminan sosial
- Informasi Kesehatan yang Dilindungi (PHI) seperti asuransi kesehatan atau nomor identifikasi medis
- Pengidentifikasi perangkat, seperti alamat IP atau alamat MAC

Untuk detail tentang jenis data yang dapat Anda lindungi, lihat [Jenis data yang dapat Anda lindungi](#).

Daftar Isi

- [Memahami kebijakan perlindungan data](#)
 - [Apa itu kebijakan perlindungan data?](#)
 - [Bagaimana kebijakan perlindungan data terstruktur?](#)
 - [Properti JSON untuk kebijakan perlindungan data](#)
 - [Properti JSON untuk pernyataan kebijakan](#)
 - [Properti JSON untuk operasi pernyataan kebijakan](#)
- [Izin IAM diperlukan untuk membuat atau bekerja dengan kebijakan perlindungan data](#)

- [Izin diperlukan untuk kebijakan perlindungan data tingkat akun](#)
- [Izin yang diperlukan untuk kebijakan perlindungan data untuk satu grup log](#)
- [Contoh kebijakan perlindungan data](#)
- [Buat kebijakan perlindungan data seluruh akun](#)
 - [Konsol](#)
 - [AWS CLI](#)
 - [Sintaks kebijakan perlindungan data untuk AWS CLI atau operasi API](#)
- [Membuat kebijakan perlindungan data untuk satu grup log](#)
 - [Konsol](#)
 - [AWS CLI](#)
 - [Sintaks kebijakan perlindungan data untuk AWS CLI atau operasi API](#)
- [Lihat data yang dibuka kedoknya](#)
- [Laporan temuan audit](#)
 - [Kebijakan kunci yang diperlukan untuk mengirim temuan audit ke ember yang dilindungi oleh AWS KMS](#)
- [Jenis data yang dapat Anda lindungi](#)
 - [CloudWatch Pengidentifikasi data terkelola log untuk tipe data sensitif](#)
 - [Kredensial](#)
 - [Pengidentifikasi data ARNs untuk tipe data kredensial](#)
 - [Pengidentifikasi perangkat](#)
 - [Pengidentifikasi data ARNs untuk tipe data perangkat](#)
 - [Informasi keuangan](#)
 - [Pengidentifikasi data ARNs untuk tipe data keuangan](#)
 - [Informasi kesehatan yang dilindungi \(PHI\)](#)
 - [Pengidentifikasi data ARNs untuk tipe data informasi kesehatan yang dilindungi \(PHI\)](#)
 - [Informasi Identifikasi Pribadi \(PII\)](#)
 - [Kata kunci untuk nomor identifikasi surat izin mengemudi](#)
 - [Kata kunci untuk nomor induk kependudukan](#)
 - [Kata kunci untuk nomor paspor](#)
 - [Kata kunci untuk nomor pokok wajib pajak](#)

- [Pengidentifikasi data ARNs untuk informasi identitas pribadi \(PII\)](#)
- [Pengidentifikasi data khusus](#)
 - [Apa itu pengidentifikasi data khusus?](#)
 - [Kendala pengenalan data kustom](#)
 - [Menggunakan pengidentifikasi data khusus di konsol](#)
 - [Menggunakan pengidentifikasi data khusus dalam kebijakan perlindungan data Anda](#)

Memahami kebijakan perlindungan data

Topik

- [Apa itu kebijakan perlindungan data?](#)
- [Bagaimana kebijakan perlindungan data terstruktur?](#)

Apa itu kebijakan perlindungan data?

CloudWatch Log menggunakan kebijakan perlindungan data untuk memilih data sensitif yang ingin Anda pindai, dan tindakan yang ingin Anda ambil untuk melindungi data tersebut. Untuk memilih data sensitif yang menarik, Anda menggunakan [pengidentifikasi data](#). CloudWatch Perlindungan data log kemudian mendeteksi data sensitif dengan menggunakan pembelajaran mesin dan pencocokan pola. Untuk menindaklanjuti pengidentifikasi data yang ditemukan, Anda dapat menentukan operasi audit dan de-identifikasi. Operasi ini memungkinkan Anda mencatat data sensitif yang ditemukan (atau tidak ditemukan), dan untuk menutupi data sensitif saat peristiwa log dilihat.

Bagaimana kebijakan perlindungan data terstruktur?

Seperti yang diilustrasikan pada gambar berikut, dokumen kebijakan perlindungan data mencakup elemen-elemen berikut:

- Informasi opsional untuk seluruh kebijakan di bagian atas dokumen
- Satu pernyataan yang mendefinisikan tindakan audit dan de-identifikasi

Hanya satu kebijakan perlindungan data yang dapat ditentukan per grup CloudWatch log Log. Kebijakan perlindungan data dapat memiliki satu atau lebih pernyataan penolakan atau de-identifikasi, tetapi hanya satu pernyataan audit.

Properti JSON untuk kebijakan perlindungan data

Kebijakan perlindungan data memerlukan informasi kebijakan dasar berikut untuk identifikasi:

- Nama — Nama kebijakan.
- Deskripsi (Opsional) — Deskripsi kebijakan.
- Versi - Versi bahasa kebijakan. Versi saat ini adalah 2021-06-01.
- Pernyataan — Daftar pernyataan yang menentukan tindakan kebijakan perlindungan data.

```
{
  "Name": "CloudWatchLogs-PersonalInformation-Protection",
  "Description": "Protect basic types of sensitive data",
  "Version": "2021-06-01",
  "Statement": [
    ...
  ]
}
```

Properti JSON untuk pernyataan kebijakan

Pernyataan kebijakan menetapkan konteks deteksi untuk operasi perlindungan data.

- Sid (Opsional) - Pengidentifikasi pernyataan.
- DataIdentifier— Data sensitif yang harus dipindai oleh CloudWatch Log. Misalnya, nama, alamat, atau nomor telepon.
- Operasi — Tindakan tindak lanjut, baik Audit atau De-identifikasi. CloudWatch Log melakukan tindakan ini ketika menemukan data sensitif.

```
{
  ...
  "Statement": [
    {
      "Sid": "audit-policy",
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/Address"
      ],
      "Operation": {
        "Audit": {
```

```

    "FindingsDestination": {}
  }
},

```

Properti JSON untuk operasi pernyataan kebijakan

Pernyataan kebijakan menetapkan salah satu operasi perlindungan data berikut.

- **Audit** — Memancarkan laporan metrik dan temuan tanpa mengganggu pencatatan. String yang cocok menambah LogEventsWithFindingsmetrik yang diterbitkan CloudWatch Log ke namespace AWS/Log. CloudWatch Anda dapat menggunakan metrik ini untuk membuat alarm.

Untuk contoh laporan temuan, lihat [Laporan temuan audit](#).

Untuk informasi selengkapnya tentang metrik yang dikirimkan oleh CloudWatch Log CloudWatch, lihat [Pemantauan dengan CloudWatch metrik](#).

- **De-identifikasi**—Tutupi data sensitif tanpa mengganggu pencatatan.

Izin IAM diperlukan untuk membuat atau bekerja dengan kebijakan perlindungan data

Agar dapat bekerja dengan kebijakan perlindungan data untuk grup log, Anda harus memiliki izin tertentu seperti yang ditunjukkan pada tabel berikut. Izin berbeda untuk kebijakan perlindungan data di seluruh akun dan untuk kebijakan perlindungan data yang berlaku untuk satu grup log.

Izin diperlukan untuk kebijakan perlindungan data tingkat akun

Note

Jika Anda melakukan salah satu operasi ini di dalam fungsi Lambda, peran eksekusi Lambda dan batas izin juga harus menyertakan izin berikut.

Operasi	Izin IAM diperlukan	Sumber daya
Membuat kebijakan perlindungan data tanpa tujuan audit	logs:PutAccountPolicy	*

Operasi	Izin IAM diperlukan	Sumber daya
Membuat kebijakan perlindungan data dengan CloudWatch Log sebagai tujuan audit	logs:PutDataProtectionPolicy	*
	logs:PutAccountPolicy	*
	logs:PutDataProtectionPolicy	*
	logs:CreateLogDelivery	*
	logs:PutResourcePolicy	*
	logs:DescribeResourcePolicies	*
Membuat kebijakan perlindungan data dengan Firehose sebagai tujuan audit	logs:DescribeLogGroups	*
	logs:PutAccountPolicy	*
	logs:PutDataProtectionPolicy	*
	logs:CreateLogDelivery	*
Membuat kebijakan perlindungan data dengan Amazon S3 sebagai tujuan audit	firehose:TagDeliveryStream	arn:aws:logs:::deliverystream/ <i>YOUR_DELIVERY_STREAM</i>
	logs:PutAccountPolicy	*

Operasi	Izin IAM diperlukan	Sumber daya
	<code>logs:PutDataProtectionPolicy</code>	*
	<code>logs:CreateLogDelivery</code>	*
	<code>s3:GetBucketPolicy</code>	<code>arn:aws:s3::: <i>YOUR_BUCKET</i></code>
	<code>s3:PutBucketPolicy</code>	<code>arn:aws:s3::: <i>YOUR_BUCKET</i></code>
Buka kedok peristiwa log bertopeng dalam grup log tertentu	<code>logs:Unmask</code>	<code>arn:aws:logs::log-group:*</code>
Melihat kebijakan perlindungan data yang ada	<code>logs:GetDataProtectionPolicy</code>	*
Menghapus kebijakan perlindungan data	<code>logs>DeleteAccountPolicy</code>	*
	<code>logs>DeleteDataProtectionPolicy</code>	*

Jika ada log audit perlindungan data yang sudah dikirim ke tujuan, maka kebijakan lain yang mengirim log ke tujuan yang sama hanya memerlukan izin `logs:PutDataProtectionPolicy` dan `logs:CreateLogDelivery` izin.

Izin yang diperlukan untuk kebijakan perlindungan data untuk satu grup log

Note

Jika Anda melakukan salah satu operasi ini di dalam fungsi Lambda, peran eksekusi Lambda dan batas izin juga harus menyertakan izin berikut.

Operasi	Izin IAM diperlukan	Sumber daya
Membuat kebijakan perlindungan data tanpa tujuan audit	logs:PutDataProtectionPolicy	arn:aws:logs::log-group: <i>YOUR_LOG_GROUP</i> :*
Membuat kebijakan perlindungan data dengan CloudWatch Log sebagai tujuan audit	logs:PutDataProtectionPolicy logs:CreateLogDelivery logs:PutResourcePolicy logs:DescribeResourcePolicies logs:DescribeLogGroups	arn:aws:logs::log-group: <i>YOUR_LOG_GROUP</i> :* * * * *
Membuat kebijakan perlindungan data dengan Firehose sebagai tujuan audit	logs:PutDataProtectionPolicy logs:CreateLogDelivery firehose:TagDeliveryStream	arn:aws:logs::log-group: <i>YOUR_LOG_GROUP</i> :* * arn:aws:logs::deliverystream/ <i>YOUR_DELIVERY_STREAM</i>
Membuat kebijakan perlindungan data dengan Amazon S3 sebagai tujuan audit	logs:PutDataProtectionPolicy logs:CreateLogDelivery s3:GetBucketPolicy	arn:aws:logs::log-group: <i>YOUR_LOG_GROUP</i> :* * arn:aws:s3::: <i>YOUR_BUCKET</i>

Operasi	Izin IAM diperlukan	Sumber daya
	s3:PutBucketPolicy	arn:aws:s3::: <i>YOUR_BUCKET</i>
Buka kedok peristiwa log bertopeng	logs:Unmask	arn:aws:logs:::log-group: <i>YOUR_LOG_GROUP</i> :*
Melihat kebijakan perlindungan data yang ada	logs:GetDataProtectionPolicy	arn:aws:logs:::log-group: <i>YOUR_LOG_GROUP</i> :*
Menghapus kebijakan perlindungan data	logs>DeleteDataProtectionPolicy	arn:aws:logs:::log-group: <i>YOUR_LOG_GROUP</i> :*

Jika ada log audit perlindungan data yang sudah dikirim ke tujuan, maka kebijakan lain yang mengirim log ke tujuan yang sama hanya memerlukan izin logs:PutDataProtectionPolicy dan logs:CreateLogDelivery izin.

Contoh kebijakan perlindungan data

Contoh kebijakan berikut memungkinkan pengguna untuk membuat, melihat, dan menghapus kebijakan perlindungan data yang dapat mengirimkan temuan audit ke ketiga jenis tujuan audit. Itu tidak mengizinkan pengguna untuk melihat data yang dibuka kedoknya.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowLogDeliveryConfiguration",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogDelivery",
        "logs:PutResourcePolicy",
```

```

        "logs:DescribeLogGroups",
        "logs:DescribeResourcePolicies"
    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowDataProtectionAndBucketConfiguration",
    "Effect": "Allow",
    "Action": [
      "logs:GetDataProtectionPolicy",
      "logs>DeleteDataProtectionPolicy",
      "logs:PutDataProtectionPolicy",
      "s3:PutBucketPolicy",
      "firehose:TagDeliveryStream",
      "s3:GetBucketPolicy"
    ],
    "Resource": [
      "arn:aws:firehose:us-east-1:111122223333:deliverystream/delivery-stream-name",
      "arn:aws:s3:::amzn-s3-demo-destination-bucket",
      "arn:aws:logs:us-east-1:111122223333:log-group:log-group-name:"
    ]
  }
]
}

```

Buat kebijakan perlindungan data seluruh akun

Anda dapat menggunakan konsol CloudWatch Log atau AWS CLI perintah untuk membuat kebijakan perlindungan data guna menutupi data sensitif untuk semua grup log di akun Anda. Melakukannya memengaruhi grup log saat ini dan grup log yang Anda buat di masa mendatang.

Important

Data sensitif terdeteksi dan disamarkan saat tertelan ke dalam grup log. Saat Anda menetapkan kebijakan perlindungan data, peristiwa log yang dicerna ke grup log sebelum waktu tersebut tidak disamarkan.

Topik

- [Konsol](#)
- [AWS CLI](#)

Konsol

Untuk menggunakan konsol untuk membuat kebijakan perlindungan data seluruh akun

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, silakan pilih Pengaturan. Itu terletak di dekat bagian bawah daftar.
3. Pilih tab Log.
4. Pilih Konfigurasi
5. Untuk pengidentifikasi data terkelola, pilih jenis data yang ingin Anda audit dan tutupi untuk semua grup log Anda. Anda dapat menyetel kotak pilihan untuk menemukan pengidentifikasi yang Anda inginkan.

Kami menyarankan Anda hanya memilih pengenal data yang relevan untuk data log dan bisnis Anda. Memilih banyak jenis data dapat menyebabkan positif palsu.

Untuk detail tentang jenis data yang dapat Anda lindungi, lihat [Jenis data yang dapat Anda lindungi](#).

6. (Opsional) Jika Anda ingin mengaudit dan menutupi jenis data lain dengan menggunakan pengidentifikasi data khusus, pilih Tambahkan pengenal data khusus. Kemudian masukkan nama untuk tipe data dan ekspresi reguler yang akan digunakan untuk mencari jenis data tersebut dalam peristiwa log. Untuk informasi selengkapnya, lihat [Pengidentifikasi data khusus](#).

Kebijakan perlindungan data tunggal dapat mencakup hingga 10 pengidentifikasi data kustom. Setiap ekspresi reguler yang mendefinisikan pengenal data kustom harus 200 karakter atau kurang.

7. (Opsional) Pilih satu atau lebih layanan untuk mengirimkan temuan audit ke. Bahkan jika Anda memilih untuk tidak mengirim temuan audit ke salah satu layanan ini, tipe data sensitif yang Anda pilih akan tetap tertutup.
8. Pilih Aktifkan perlindungan data.

AWS CLI

Untuk menggunakan AWS CLI untuk membuat kebijakan perlindungan data

1. Gunakan editor teks untuk membuat file kebijakan bernama `DataProtectionPolicy.json`. Untuk informasi tentang sintaks kebijakan, lihat bagian berikut.
2. Masukkan perintah berikut:

```
aws logs put-account-policy \
--policy-name TEST_POLICY --policy-type "DATA_PROTECTION_POLICY" \
--policy-document file://policy.json \
--scope "ALL" \
--region us-west-2
```

Sintaks kebijakan perlindungan data untuk AWS CLI atau operasi API

Saat Anda membuat kebijakan perlindungan data JSON untuk digunakan dalam operasi AWS CLI perintah atau API, kebijakan tersebut harus menyertakan dua blok JSON:

- Blok pertama harus menyertakan `DataIdentifier` array dan `Operation` properti dengan `Audit` tindakan. `DataIdentifierArray` mencantumkan jenis data sensitif yang ingin Anda tutupi. Untuk informasi lebih lanjut tentang opsi yang tersedia, lihat [Jenis data yang dapat Anda lindungi](#).

`Operation` properti dengan `Audit` tindakan diperlukan untuk menemukan istilah data sensitif. `AuditTindakan` ini harus berisi `FindingsDestination` objek. Anda dapat menggunakan `FindingsDestination` objek tersebut secara opsional untuk mencantumkan satu atau beberapa tujuan untuk mengirim laporan temuan audit. Jika Anda menentukan tujuan seperti grup log, aliran Amazon Data Firehose, dan bucket S3, mereka harus sudah ada. Untuk contoh laporan audit findings, lihat [Laporan temuan audit](#)

- Blok kedua harus menyertakan `DataIdentifier` array dan `Operation` properti dengan `Deidentify` tindakan. `DataIdentifierArray` harus sama persis dengan `DataIdentifier` array di blok pertama kebijakan.

`Operation` properti dengan `Deidentify` tindakan adalah apa yang sebenarnya menutupi data, dan itu harus berisi `"MaskConfig": {}` objek. `"MaskConfig": {}` objek harus kosong.

Berikut ini adalah contoh kebijakan perlindungan data yang hanya menggunakan pengidentifikasi data terkelola. Kebijakan ini menutupi alamat email dan SIM Amerika Serikat.

Untuk informasi tentang kebijakan yang menentukan pengenalan data kustom, lihat [Menggunakan pengidentifikasi data khusus dalam kebijakan perlindungan data Anda](#).

```
{
  "Name": "data-protection-policy",
  "Description": "test description",
  "Version": "2021-06-01",
  "Statement": [{
    "Sid": "audit-policy",
    "DataIdentifier": [
      "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
      "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
    ],
    "Operation": {
      "Audit": {
        "FindingsDestination": {
          "CloudWatchLogs": {
            "LogGroup": "EXISTING_LOG_GROUP_IN_YOUR_ACCOUNT",
          },
          "Firehose": {
            "DeliveryStream": "EXISTING_STREAM_IN_YOUR_ACCOUNT"
          },
          "S3": {
            "Bucket": "EXISTING_BUCKET"
          }
        }
      }
    }
  },
  {
    "Sid": "redact-policy",
    "DataIdentifier": [
      "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
      "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
    ],
    "Operation": {
      "Deidentify": {
        "MaskConfig": {}
      }
    }
  }
]
```


Membuat kebijakan perlindungan data untuk satu grup log

Anda dapat menggunakan konsol CloudWatch Log atau AWS CLI perintah untuk membuat kebijakan perlindungan data untuk menutupi data sensitif.

Anda dapat menetapkan satu kebijakan perlindungan data untuk setiap grup log. Setiap kebijakan perlindungan data dapat mengaudit berbagai jenis informasi. Setiap kebijakan perlindungan data dapat mencakup satu pernyataan audit.

Topik

- [Konsol](#)
- [AWS CLI](#)

Konsol

Untuk menggunakan konsol untuk membuat kebijakan perlindungan data

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Pilih nama grup log.
4. Pilih Tindakan, Buat kebijakan perlindungan data.
5. Untuk pengidentifikasi data terkelola, pilih jenis data yang ingin Anda audit dan tutupi di grup log ini. Anda dapat mengetikkan kotak pilihan untuk menemukan pengidentifikasi yang Anda inginkan.

Kami menyarankan Anda hanya memilih pengenalan data yang relevan untuk data log dan bisnis Anda. Memilih banyak jenis data dapat menyebabkan positif palsu.

Untuk detail tentang jenis data yang dapat Anda lindungi dengan menggunakan pengenalan data terkelola, lihat [Jenis data yang dapat Anda lindungi](#).

6. (Opsional) Jika Anda ingin mengaudit dan menutupi jenis data lain dengan menggunakan pengidentifikasi data khusus, pilih Tambahkan pengenalan data khusus. Kemudian masukkan nama untuk tipe data dan ekspresi reguler yang akan digunakan untuk mencari jenis data tersebut dalam peristiwa log. Untuk informasi selengkapnya, lihat [Pengidentifikasi data khusus](#).

Kebijakan perlindungan data tunggal dapat mencakup hingga 10 pengidentifikasi data kustom. Setiap ekspresi reguler yang mendefinisikan pengenalan data kustom harus 200 karakter atau kurang.

7. (Opsional) Pilih satu atau lebih layanan untuk mengirimkan temuan audit ke. Bahkan jika Anda memilih untuk tidak mengirim temuan audit ke salah satu layanan ini, tipe data sensitif yang Anda pilih akan tetap tertutup.
8. Pilih Aktifkan perlindungan data.

AWS CLI

Untuk menggunakan AWS CLI untuk membuat kebijakan perlindungan data

1. Gunakan editor teks untuk membuat file kebijakan bernama `DataProtectionPolicy.json`. Untuk informasi tentang sintaks kebijakan, lihat bagian berikut.
2. Masukkan perintah berikut:

```
aws logs put-data-protection-policy --log-group-identifier "my-log-group" --policy-document file:///Path/DataProtectionPolicy.json --region us-west-2
```

Sintaks kebijakan perlindungan data untuk AWS CLI atau operasi API

Saat Anda membuat kebijakan perlindungan data JSON untuk digunakan dalam operasi AWS CLI perintah atau API, kebijakan tersebut harus menyertakan dua blok JSON:

- Blok pertama harus menyertakan `DataIdentifier` array dan `Operation` properti dengan `Audit` tindakan. `DataIdentifierArray` mencantumkan jenis data sensitif yang ingin Anda tutupi. Untuk informasi lebih lanjut tentang opsi yang tersedia, lihat [Jenis data yang dapat Anda lindungi](#).

`Operation` Properti dengan `Audit` tindakan diperlukan untuk menemukan istilah data sensitif. `AuditTindakan` ini harus berisi `FindingsDestination` objek. Anda dapat menggunakan `FindingsDestination` objek tersebut secara opsional untuk mencantumkan satu atau beberapa tujuan untuk mengirim laporan temuan audit. Jika Anda menentukan tujuan seperti grup log, aliran Amazon Data Firehose, dan bucket S3, mereka harus sudah ada. Untuk contoh laporan audit findings, lihat [Laporan temuan audit](#)

- Blok kedua harus menyertakan DataIdentifier array dan Operation properti dengan Deidentify tindakan. DataIdentifierArray harus sama persis dengan DataIdentifier array di blok pertama kebijakan.

OperationProperti dengan Deidentify tindakan adalah apa yang sebenarnya menutupi data, dan itu harus berisi "MaskConfig": {} objek. "MaskConfig": {}Objek harus kosong.

Berikut ini adalah contoh kebijakan perlindungan data yang menutupi alamat email dan SIM Amerika Serikat.

```
{
  "Name": "data-protection-policy",
  "Description": "test description",
  "Version": "2021-06-01",
  "Statement": [{
    "Sid": "audit-policy",
    "DataIdentifier": [
      "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
      "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
    ],
    "Operation": {
      "Audit": {
        "FindingsDestination": {
          "CloudWatchLogs": {
            "LogGroup": "EXISTING_LOG_GROUP_IN_YOUR_ACCOUNT",
          },
          "Firehose": {
            "DeliveryStream": "EXISTING_STREAM_IN_YOUR_ACCOUNT"
          },
          "S3": {
            "Bucket": "EXISTING_BUCKET"
          }
        }
      }
    }
  },
  {
    "Sid": "redact-policy",
    "DataIdentifier": [
      "arn:aws:dataprotection::aws:data-identifier/EmailAddress",
      "arn:aws:dataprotection::aws:data-identifier/DriversLicense-US"
    ],
  },
]
```

```

      "Operation": {
        "Deidentify": {
          "MaskConfig": {}
        }
      }
    }
  ]
}

```

Lihat data yang dibuka kedoknya

Untuk melihat data yang dibuka kedok, pengguna harus memiliki izin. `logs:Unmask` Pengguna dengan izin ini dapat melihat data yang dibuka kedoknya dengan cara berikut:

- Saat melihat peristiwa dalam aliran log, pilih Tampilan, Buka Kedok.
- Gunakan kueri CloudWatch Logs Insights yang menyertakan perintah `unmask(@message)`. Contoh query berikut menampilkan 20 peristiwa log terbaru dalam aliran, membuka kedoknya:

```

fields @timestamp, @message, unmask(@message)
| sort @timestamp desc
| limit 20

```

Untuk informasi selengkapnya tentang perintah Wawasan CloudWatch Log, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

- Gunakan [GetLogEvents](#) atau [FilterLogEvents](#) operasi dengan `unmask` parameter.

`CloudWatchLogsFullAccess` Kebijakan tersebut termasuk `logs:Unmask` izin. Untuk memberikan `logs:Unmask` kepada pengguna yang tidak memiliki `CloudWatchLogsFullAccess`, Anda dapat melampirkan kebijakan IAM kustom ke pengguna tersebut. Untuk informasi selengkapnya, lihat [Menambahkan izin ke pengguna \(konsol\)](#).

Laporan temuan audit

Jika Anda menyiapkan kebijakan audit perlindungan data CloudWatch Log untuk menulis laporan audit ke CloudWatch Log, Amazon S3, atau Firehose, laporan temuan ini serupa dengan contoh berikut. CloudWatch Log menulis satu laporan temuan untuk setiap peristiwa log yang berisi data sensitif.

```

{

```

```

    "auditTimestamp": "2023-01-23T21:11:20Z",
    "resourceArn": "arn:aws:logs:us-west-2:111122223333:log-group:/aws/lambda/
MyLogGroup:*",
    "dataIdentifiers": [
      {
        "name": "EmailAddress",
        "count": 2,
        "detections": [
          {
            "start": 13,
            "end": 26
          },
          {
            "start": 30,
            "end": 43
          }
        ]
      }
    ]
  }
}

```

Bidang dalam laporan adalah sebagai berikut:

- `resourceArnBidang` menampilkan grup log tempat data sensitif ditemukan.
- `dataIdentifiersObjek` menampilkan informasi tentang temuan untuk satu jenis data sensitif yang Anda audit.
- `nameBidang` mengidentifikasi jenis data sensitif yang dilaporkan bagian ini.
- `countBidang` menampilkan berapa kali jenis data sensitif ini muncul dalam peristiwa log.
- `endBidang` `start` dan menunjukkan di mana dalam peristiwa log, berdasarkan jumlah karakter, setiap kemunculan data sensitif muncul.

Contoh sebelumnya menunjukkan laporan menemukan dua alamat email dalam satu peristiwa log. Alamat email pertama dimulai pada karakter ke-13 dari peristiwa log dan berakhir pada karakter ke-26. Alamat email kedua berjalan dari karakter ke-30 ke karakter ke-43. Meskipun peristiwa log ini memiliki dua alamat email, nilai `LogEventsWithFindings` metrik hanya bertambah satu, karena metrik tersebut menghitung jumlah peristiwa log yang berisi data sensitif, bukan jumlah kejadian data sensitif.

Kebijakan kunci yang diperlukan untuk mengirim temuan audit ke ember yang dilindungi oleh AWS KMS

Anda dapat melindungi data dalam bucket Amazon S3 dengan mengaktifkan Enkripsi Sisi Server dengan Amazon S3-Managed Keys (SSE-S3) atau Enkripsi Sisi Server dengan Kunci KMS (SSE-KMS). Untuk informasi selengkapnya, lihat [Melindungi data menggunakan enkripsi sisi server di Panduan Pengguna Amazon S3](#).

Jika Anda mengirim temuan audit ke bucket yang dilindungi dengan SSE-S3, konfigurasi tambahan tidak diperlukan. Amazon S3 menangani kunci enkripsi.

Jika Anda mengirim temuan audit ke bucket yang dilindungi oleh SSE-KMS, Anda harus memperbarui kebijakan kunci untuk kunci KMS Anda sehingga akun pengiriman log dapat menulis ke bucket S3 Anda. Untuk informasi selengkapnya tentang kebijakan kunci yang diperlukan untuk digunakan dengan SSE-KMS, lihat [Amazon S3](#) di Panduan Pengguna Amazon CloudWatch Logs.

Jenis data yang dapat Anda lindungi

Bagian ini berisi informasi tentang jenis data yang dapat Anda lindungi dalam kebijakan perlindungan data CloudWatch Log. CloudWatch Pengidentifikasi data terkelola log menawarkan tipe data yang telah dikonfigurasi sebelumnya untuk melindungi data keuangan, informasi kesehatan pribadi (PHI), dan informasi identitas pribadi (PII). Anda juga dapat menggunakan pengidentifikasi data khusus untuk membuat pengidentifikasi data yang disesuaikan dengan kasus penggunaan spesifik Anda.

Daftar Isi

- [CloudWatch Pengidentifikasi data terkelola log untuk tipe data sensitif](#)
 - [Kredensial](#)
 - [Pengidentifikasi data ARNs untuk tipe data kredensial](#)
 - [Pengidentifikasi perangkat](#)
 - [Pengidentifikasi data ARNs untuk tipe data perangkat](#)
 - [Informasi keuangan](#)
 - [Pengidentifikasi data ARNs untuk tipe data keuangan](#)
 - [Informasi kesehatan yang dilindungi \(PHI\)](#)
 - [Pengidentifikasi data ARNs untuk tipe data informasi kesehatan yang dilindungi \(PHI\)](#)
 - [Informasi Identifikasi Pribadi \(PII\)](#)
 - [Kata kunci untuk nomor identifikasi surat izin mengemudi](#)

- [Kata kunci untuk nomor induk kependudukan](#)
- [Kata kunci untuk nomor paspor](#)
- [Kata kunci untuk nomor pokok wajib pajak](#)
- [Pengidentifikasi data ARNs untuk informasi identitas pribadi \(PII\)](#)
- [Pengidentifikasi data khusus](#)
 - [Apa itu pengidentifikasi data khusus?](#)
 - [Kendala pengenalan data kustom](#)
 - [Menggunakan pengidentifikasi data khusus di konsol](#)
 - [Menggunakan pengidentifikasi data khusus dalam kebijakan perlindungan data Anda](#)

CloudWatch Pengidentifikasi data terkelola log untuk tipe data sensitif

Bagian ini berisi informasi tentang jenis data yang dapat Anda lindungi menggunakan pengidentifikasi data terkelola, dan negara dan wilayah mana yang relevan untuk masing-masing jenis data tersebut.

Untuk beberapa jenis data sensitif, perlindungan data CloudWatch Log memindai kata kunci di dekat data, dan menemukan kecocokan hanya jika menemukan kata kunci itu. Jika kata kunci harus berada di dekat tipe data tertentu, kata kunci biasanya harus berada dalam 30 karakter (inklusif) dari data tersebut.

Jika kata kunci berisi spasi, perlindungan data CloudWatch Log secara otomatis cocok dengan variasi kata kunci yang kehilangan ruang atau yang berisi garis bawah (_) atau tanda hubung (-) alih-alih spasi. Dalam beberapa kasus, CloudWatch Log juga memperluas atau menyingkat kata kunci untuk mengatasi variasi umum kata kunci.

Tabel berikut mencantumkan jenis informasi kredensial, perangkat, keuangan, medis, dan kesehatan yang dilindungi (PHI) yang dapat dideteksi oleh CloudWatch Log menggunakan pengidentifikasi data terkelola. Tabel ini merupakan tambahan untuk tipe data tertentu yang mungkin juga memenuhi syarat sebagai informasi pengenalan pribadi (PII).

Pengidentifikasi yang didukung yang independen bahasa dan wilayah

Pengidentifikasi	Kategori
Address	Personal

Pengidentifikasi	Kategori
AwsSecretKey	Kredensial
CreditCardExpiration	Keuangan
CreditCardNumber	Keuangan
CreditCardSecurityCode	Keuangan
EmailAddress	Personal
IpAddress	Personal
LatLong	Personal
Name	Personal
OpenSshPrivateKey	Kredensial
PgpPrivateKey	Kredensial
PkcsPrivateKey	Kredensial
PuttyPrivateKey	Kredensial
VehicleIdentificationNumber	Personal

Pengidentifikasi data yang bergantung pada wilayah harus menyertakan nama pengenalan, lalu tanda hubung, dan kemudian kode dua huruf (ISO 3166-1 alpha-2). Misalnya, `DriversLicense-US`.

Pengidentifikasi yang didukung yang harus menyertakan kode negara atau wilayah dua huruf

Pengidentifikasi	Kategori	Negara dan bahasa
BankAccountNumber	Keuangan	DE, ES, FR, GB, ITU, KITA
CepCode	Personal	BR
Cnpj	Personal	BR

Pengidentifikasi	Kategori	Negara dan bahasa
CpfCode	Personal	BR
DriversLicense	Personal	DI, AU, BE, BG, CA, CY, CZ, DE, DK, E, ES, FI, FR, GB, GR, HR, HU, YAITU, ITU, LT, LU, LV, MT, NL, PL, PT, RO, SE, SI, SK, US
DrugEnforcementAgencyNumber	Health	AS
ElectoralRollNumber	Personal	GB
HealthInsuranceCardNumber	Health	EU
HealthInsuranceClaimNumber	Health	AS
HealthInsuranceNumber	Health	FR
HealthcareProcedureCode	Health	AS
IndividualTaxIdentificationNumber	Personal	AS
InseeCode	Personal	FR
MedicareBeneficiaryNumber	Health	AS
NationalDrugCode	Health	AS
NationalIdentificationNumber	Personal	DE, ES, ITU
NationalInsuranceNumber	Personal	GB
NationalProviderId	Health	AS
NhsNumber	Health	GB
NieNumber	Personal	ES

Pengidentifikasi	Kategori	Negara dan bahasa
NifNumber	Personal	ES
PassportNumber	Personal	CA, DE, ES, FR, GB, ITU, KAMI
PermanentResidenceNumber	Personal	CA
PersonalHealthNumber	Health	CA
PhoneNumber	Personal	BR, DE, ES, FR, GB, ITU, KAMI
PostalCode	Personal	CA
RgNumber	Personal	BR
SocialInsuranceNumber	Personal	CA
Ssn	Personal	ES, KITA
TaxId	Personal	DE, ES, FR, GB
ZipCode	Personal	AS

Kredensial

CloudWatch Perlindungan data log dapat menemukan jenis kredensial berikut.

Jenis data	ID pengenal data	Diperlukan kata kunci	Negara dan wilayah
AWS kunci akses rahasia	AwsSecretKey	aws_secret_access_key , credentials , secret access key, secret key, set-awscr edential	Semua

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah
Kunci pribadi OpenSSH	OpenSSHPrivateKey	Tidak ada	Semua
Kunci pribadi PGP	PgpPrivateKey	Tidak ada	Semua
Kunci Pribadi PKCS	PkcsPrivateKey	Tidak ada	Semua
Kunci pribadi PuTTY	PuttyPrivateKey	Tidak ada	Semua

Pengidentifikasi data ARNs untuk tipe data kredensial

Berikut ini mencantumkan Nama Sumber Daya Amazon (ARNs) untuk pengidentifikasi data yang dapat Anda tambahkan ke kebijakan perlindungan data Anda.

Pengidentifikasi data kredensi ARNs

```
arn:aws:dataprotection::aws:data-identifier/AwsSecretKey
```

```
arn:aws:dataprotection::aws:data-identifier/OpenSshPrivateKey
```

```
arn:aws:dataprotection::aws:data-identifier/PgpPrivateKey
```

```
arn:aws:dataprotection::aws:data-identifier/PkcsPrivateKey
```

```
arn:aws:dataprotection::aws:data-identifier/PuttyPrivateKey
```

Pengidentifikasi perangkat

CloudWatch Perlindungan data log dapat menemukan jenis pengidentifikasi perangkat berikut.

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah
Alamat IP	IpAddress	Tidak ada	Semua

Pengidentifikasi data ARNs untuk tipe data perangkat

Berikut ini mencantumkan Nama Sumber Daya Amazon (ARNs) untuk pengidentifikasi data yang dapat Anda tambahkan ke kebijakan perlindungan data Anda.

Pengenalan data perangkat ARN

```
arn:aws:dataprotection::aws:data-identifier/IpAddress
```

Informasi keuangan

CloudWatch Perlindungan data log dapat menemukan jenis informasi keuangan berikut.

Jika Anda menetapkan kebijakan perlindungan data, CloudWatch Log akan memindai pengenalan data yang Anda tentukan, apa pun geolokasi grup log tersebut berada. Informasi di kolom Negara dan wilayah dalam tabel ini menunjukkan apakah kode negara dua huruf harus ditambahkan ke pengenalan data untuk mendeteksi kata kunci yang sesuai untuk negara dan wilayah tersebut.

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor rekening bank	BankAccountNumber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel Kata kunci untuk nomor rekening bank nanti di bagian ini.	Prancis, Jerman, Italia, Spanyol, Inggris Raya, Amerika Serikat	Termasuk Nomor Rekening Bank Internasional (IBANs) yang

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
				terdiri dari hingga 34 karakter alfanumerik, termasuk elemen seperti kode negara.
Tanggal kedaluwarsa kartu kredit	CreditCardExpiration	exp d, exp m, exp y, expiration , expiry	Semua	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor kartu kredit	CreditCardNumber	account number, american express, amex, bank card, card, card number, card num, cc #, ccn, check card, credit, credit card#, dankort, debit, debit card, diners club, discover, electron, japanese card bureau, jcb, mastercard , mc, pan, payment account number, payment card number, pcn, union pay, visa	Semua	Deteksi mengharuskan data menjadi urutan 13-19 digit yang mematuhi rumus pemeriksaan Luhn, dan menggunakan awalan nomor kartu standar untuk salah satu jenis kartu kredit berikut: American Express, Dankort, Diner's

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
				Club, Discover, Electron, Japanese Card Bureau (JCB), Mastercard, dan Visa. UnionPay
Kode verifikasi kartu kredit	CreditCardSecurity Code	card id, card identification code, card identification number , card security code, card validation code , card validation number , card verification data , card verification value, cvc, cvc2, cvv, cvv2, elo verification code	Semua	

Kata kunci untuk nomor rekening bank

Gunakan kata kunci berikut untuk nomor rekening bank. Ini termasuk Nomor Rekening Bank Internasional (IBANs) yang terdiri dari hingga 34 karakter alfanumerik, termasuk elemen seperti kode negara.

Negara	Kata kunci
France	account code, account number, accountno# , accountnumber# , bban, code bancaire, compte bancaire, customer account id, customer account number, customer bank account id, iban, numéro de compte
Germany	account code, account number, accountno# , accountnumber# , bankleitzahl , bban, customer account id, customer account number, customer bank account id, geheimzahl , iban, kartennummer , kontonummer , kreditkartennummer , sepa
Italy	account code, account number, accountno# , accountnumber# , bban, codice bancario, conto bancario, customer account id, customer account number, customer bank account id, iban, numero di conto
Spain	account code, account number, accountno# , accountnumber# , bban, código cuenta, código cuenta bancaria, cuenta cliente id, customer account ID, customer account number, customer bank account id, iban, número cuenta bancaria cliente, número cuenta cliente
Britania Raya	account code, account number, accountno# , accountnumber# , bban, customer account ID, customer account number, customer bank account id, iban, sepa
Amerika Serikat	bank account, bank acct, checking account, checking acct, deposit account, deposit acct, savings account, savings acct, chequing account, chequing acct

CloudWatch Log tidak melaporkan kejadian urutan berikut, yang telah disediakan oleh penerbit kartu kredit untuk pengujian publik.

```
1220000000000003, 2222405343248877, 2222990905257051, 2223007648726984,
2223577120017656,
30569309025904, 34343434343434, 3528000700000000, 3530111333300000, 3566002020360505,
36148900647913,
```



```
36700102000000, 371449635398431, 378282246310005, 378734493671000, 38520000023237,
4012888888881881,
4111111111111111, 4222222222222, 4444333322221111, 4462030000000000, 4484070000000000,
49118300000000,
4917300800000000, 4917610000000000, 4917610000000000003, 5019717010103742,
5105105105105100,
5111010030175156, 5185540810000019, 5200828282828210, 5204230080000017,
5204740009900014, 5420923878724339,
5454545454545454, 5455330760000018, 55069004900000436, 55069004900000444,
55069005100000234, 5506920809243667,
5506922400634930, 5506927427317625, 5553042241984105, 5555553753048194,
555555555554444, 5610591081018250,
6011000990139424, 6011000400000000, 6011111111111117, 630490017740292441,
630495060000000000,
6331101999990016, 6759649826438453, 6799990100000000019, and 76009244561.
```

Pengidentifikasi data ARNs untuk tipe data keuangan

Berikut ini mencantumkan Nama Sumber Daya Amazon (ARNs) untuk pengidentifikasi data yang dapat Anda tambahkan ke kebijakan perlindungan data Anda.

Pengidentifikasi data keuangan ARNs

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-DE
```

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-FR
```

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-GB
```

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-IT
```

```
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-US
```

```
arn:aws:dataprotection::aws:data-identifier/CreditCardExpiration
```

```
arn:aws:dataprotection::aws:data-identifier/CreditCardNumber
```

```
arn:aws:dataprotection::aws:data-identifier/CreditCardSecurityCode
```

Informasi kesehatan yang dilindungi (PHI)

CloudWatch Perlindungan data log dapat menemukan jenis informasi kesehatan yang dilindungi (PHI) berikut.

Jika Anda menetapkan kebijakan perlindungan data, CloudWatch Log akan memindai pengenal data yang Anda tentukan, apa pun geolokasi grup log tersebut berada. Informasi di kolom Negara dan wilayah dalam tabel ini menunjukkan apakah kode negara dua huruf harus ditambahkan ke pengenal data untuk mendeteksi kata kunci yang sesuai untuk negara dan wilayah tersebut.

Jenis data	ID pengenal data	Diperlukan kata kunci	Negara dan wilayah
Nomor registrasi Badan Penegakan Narkoba (DEA)	DrugEnforcementAgencyNumber	dea number, dea registration	Amerika Serikat
Nomor Kartu Asuransi Kesehatan (EHIC)	HealthInsuranceCardNumber	assicurazione sanitaria numero, carta assicurazione numero, carte d'assurance maladie , carte européenne d'assurance maladie , ceam, ehic, ehic#, finlandehicnumber# , gesundheitskarte , hälsokort , health card, health card number, health insurance card, health insurance number, insurance card number, krankenversicherungskarte , krankenversicherungsnnummer , medical account number, numero conto medico,	Uni Eropa

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah
		numéro d'assurance maladie , numéro de carte d'assurance , numéro de compte medical, número de cuenta médica, número de seguro de salud, número de tarjeta de seguro, sairaanho itokortin , sairausva kuutuskortti , sairausvakuutusnumero , sjukförsäkringsnummer, sjukförsäkringskort , suomi ehic-numero , tarjeta de salud, terveystkortti , tessera sanitaria assicurazione numero , versicherungsnummer	
Nomor Klaim Asuransi Kesehatan (HICN)	HealthInsuranceClaimNumber	health insurance claim number, hic no, hic no., hic number, hic#, hicn, hicn#, hicno#	Amerika Serikat
Nomor asuransi atau identifikasi medis	HealthInsuranceNumber	carte d'assuré social, carte vitale, insurance card	France

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah
Kode Sistem Pengkodean Prosedur Umum Pemeliharaan Kesehatan (HCPCS)	HealthcareProcedureCode	current procedural terminology , hcpcs, healthcare common procedure coding system	Amerika Serikat
Nomor Penerima Medicare (MBN)	MedicareBeneficiaryNumber	mbi, medicare beneficiary	Amerika Serikat
Kode Obat Nasional (NDC)	NationalDrugCode	national drug code, ndc	Amerika Serikat
Pengidentifikasi Penyedia Nasional (NPI)	NationalProviderId	hipaa, n.p.i., national provider, npi	Amerika Serikat
Nomor Layanan Kesehatan Nasional (NHS)	NhsNumber	national health service, NHS	Inggris Raya
Nomor Kesehatan Pribadi	PersonalHealthNumber	canada healthcare number, msp number, care number, phn, soins de santé	Kanada

Pengidentifikasi data ARNs untuk tipe data informasi kesehatan yang dilindungi (PHI)

Berikut ini mencantumkan pengenalan data Amazon Resource Names (ARNs) yang dapat digunakan dalam kebijakan perlindungan data informasi kesehatan yang dilindungi (PHI).

Pengidentifikasi data PHI ARNs

```
arn:aws:dataprotection::aws:data-identifier/DrugEnforcementAgencyNumber-US
```

Pengidentifikasi data PHI ARNs

```
arn:aws:dataprotection::aws:data-identifier/HealthcareProcedure
Code-US
```

```
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceCard
Number-EU
```

```
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceClai
mNumber-US
```

```
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceNumb
er-FR
```

```
arn:aws:dataprotection::aws:data-identifier/MedicareBeneficiary
Number-US
```

```
arn:aws:dataprotection::aws:data-identifier/NationalDrugCode-US
```

```
arn:aws:dataprotection::aws:data-identifier/NationalInsuranceNu
mber-GB
```

```
arn:aws:dataprotection::aws:data-identifier/NationalProviderId-US
```

```
arn:aws:dataprotection::aws:data-identifier/NhsNumber-GB
```

```
arn:aws:dataprotection::aws:data-identifier/PersonalHealthNumber-
CA
```

Informasi Identifikasi Pribadi (PII)

CloudWatch Perlindungan data log dapat menemukan jenis informasi identitas pribadi (PII) berikut.

Jika Anda menetapkan kebijakan perlindungan data, CloudWatch Log akan memindai pengenalan data yang Anda tentukan, apa pun geolokasi grup log tersebut berada. Informasi di kolom Negara dan wilayah dalam tabel ini menunjukkan apakah kode negara dua huruf harus ditambahkan ke pengenalan data untuk mendeteksi kata kunci yang sesuai untuk negara dan wilayah tersebut.

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Tanggal lahir	DateOfBirth	dob, date of birth, birthdate, birth date, birthday, b-day, bday	Setiap	Support mencakup sebagian besar format tanggal, seperti semua digit dan kombinasi digit dan nama bulan. Komponen tanggal dapat dipisahkan oleh spasi, garis miring (/), atau tanda hubung (-).
Kode Pos Endereçamento (CEP)	CepCode	cep, código de endereçamento postal, código de	Brazil	

Jenis data	ID pengenal data	Diperlukan kata kunci	Negara dan wilayah	Catatan
		endereço postal		
Kadastro Nacional da Pessoa Jurídica (CNPJ)	Cnpj	cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj	Brazil	
Kadaster Pessoas Físicas (CPF)	CpfCode	Cadastro de pessoas físicas, cadastro de pessoas físicas, cadastro de pessoa física, cadastro de pessoa fisica, cpf	Brazil	
Nomor identifikasi lisensi	DriversLicense	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi SIM nanti di bagian ini.	Banyak negara. Untuk detailnya, lihat tabel nomor identifikasi SIM.	
Nomor Roll Pemilu	Electoral RollNumber	electoral #, electoral number, electoral roll #, electoral roll no., electoral roll number, electoral rollno	Britania Raya	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Identifikasi wajib pajak perorangan	Individua lTaxIdent icationNu mber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi wajib pajak perorangan nanti di bagian ini.	Brasil, Prancis, Jerman, Spanyol, Britania Raya	
Institut Nasional untuk Statistik dan Studi Ekonomi (INSEE)	InseeCode	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel Kata kunci untuk nomor identifikasi nasional nanti di bagian ini.	France	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor Identifikasi Nasional	NationalIdentificationNumber	Ya. Untuk detailnya, lihat tabel Kata kunci untuk nomor identifikasi nasional nanti di bagian ini.	Jerman, Italia, Spanyol	Ini termasuk pengidentifikasi Documento Nacional de Identidad (DNI) (Spanyol), kode fiskal Codice (Italia), dan nomor Kartu Identitas Nasional (Jerman).
Nomor Asuransi Nasional (NINO)	NationalInsuranceNumber	insurance no., insurance number, insurance# , national insurance number, nationalinsurance# , nationalinsurancenumber , nin, nino	Britania Raya	–

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nama Identidad Extranjero (NIE)	NieNumber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi wajib pajak perorangan nanti di bagian ini.	Spain	
Nomor Identifikasi Fiskal (NIF)	NifNumber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi wajib pajak perorangan nanti di bagian ini.	Spain	
Nomor paspor	PassportNumber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel Kata kunci untuk nomor paspor nanti di bagian ini.	Kanada, Prancis, Jerman, Italia, Spanyol, Inggris Raya, Amerika Serikat	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor tempat tinggal permanen	Permanent Residence Number	carte résident permanent , numéro carte résident permanent , numéro résident permanent , permanent resident card, permanent resident card number, permanent resident no, permanent resident no., permanent resident number, pr no, pr no., pr non, pr number, résident permanent no., résident permanent non	Kanada	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor telepon	PhoneNumber	Brasil: kata kunci juga meliputi <code>cel:celular,fone, residencial ,numero residencial , telefone</code> Lainnya: <code>cell,contact,fax,number,,mobile,phone,phonenumber,tel,telephone , telephone number</code>	Brasil, Kanada, Prancis, Jerman, Italia, Spanyol, Inggris Raya, Amerika Serikat	Ini termasuk nomor bebas pulsa di Amerika Serikat dan nomor faks. Jika kata kunci berada di dekat data, nomor tersebut tidak harus menyertakan kode negara. Jika kata kunci tidak dekat dengan

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
				data, nomor tersebut harus menyertakan kode negara.
Kode Pos	PostalCode	Tidak ada	Kanada	
Registrasi Geral (RG)	RgNumber	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi wajib pajak perorangan nanti di bagian ini.	Brazil	
Nomor Pokok Wajib Pajak (SIN)	SocialInsuranceNumber	canadian id, numéro d'assurance sociale, social insurance number, sin	Kanada	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor Jaminan Sosial (SSN)	Ssn	Spanyol número de la seguridad social—social security no.,social security no., número de la seguridad social,social security number,socialsecurityno# ,ssn, ssn# Amerika Serikat social security—ss#,, ssn	Spanyol, Amerika Serikat	
Nomor identifikasi wajib pajak atau referensi	TaxId	Ya. Kata kunci yang berbeda berlaku untuk negara yang berbeda. Untuk detailnya, lihat tabel nomor identifikasi wajib pajak perorangan nanti di bagian ini. .	Prancis, Jerman, Spanyol, Britania Raya	Ini termasuk TIN (Prancis) ; Steueridentifikationsnummer (Jerman); CIF (Spanyol) ; dan TRN, UTR (Britania Raya).

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Kode Pos	ZipCode	zip code, zip+4	Amerika Serikat	Kode pos Amerika Serikat.
Alamat surat-menyurat	Address	Tidak ada	Australia, Kanada, Prancis, Jerman, Italia, Spanyol, Inggris Raya, Amerika Serikat	Meskipun kata kunci tidak diperlukan, deteksi memerlukan alamat untuk menyertakan nama kota atau tempat dan kode pos atau kode pos.
Alamat surat elektronik	EmailAddress	Tidak ada	Setiap	

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Koordinat Global Positioning System (GPS)	LatLong	<code>coordinate</code> , <code>coordinates</code> , <code>lat</code> <code>long</code> , <code>latitude</code> <code>longitude</code> , <code>location</code> , <code>position</code>	Setiap	CloudWatch Log dapat mendeteksi koordinat GPS jika koordinat lintang dan bujur disimpan sebagai pasangan dan mereka dalam format Derajat Desimal (DD), misalnya, 41.948614 , -87.655311. Support tidak menyertakan koordinat

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
				dalam format Degrees Decimal Minutes (DDM), misalnya format 41° 56.9168'N 87° 39.3187'W , atau Derajat, Menit, Detik (DMS), misalnya 41° 56'55.010 4 "N 87° 39'19.119 6"W.

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nama lengkap	Name	Tidak ada	Setiap	CloudWatch Log hanya dapat mendeteksi nama lengkap. Dukungan terbatas pada set karakter Latin.

Jenis data	ID pengenalan data	Diperlukan kata kunci	Negara dan wilayah	Catatan
Nomor Identifikasi Kendaraan (VIN)	VehicleIdentificationNumber	Fahrgestellnummer , niv, numarul de identificare , numarul seriei de sasiu, serie sasiu, numer VIN, Número de Identificação do Veículo, Número de Identificación de Automóviles , numéro d'identification du véhicule, vehicle identification number, vin, VIN numeris	Setiap	CloudWatch Log dapat mendeteksi VINs yang terdiri dari urutan 17 karakter dan mematuhi standar ISO 3779 dan 3780. Standar ini dirancang untuk penggunaan di seluruh dunia.

Kata kunci untuk nomor identifikasi surat izin mengemudi

Untuk mendeteksi berbagai jenis nomor identifikasi SIM, CloudWatch Log membutuhkan kata kunci untuk berada di dekat nomor. Tabel berikut mencantumkan kata kunci yang dikenali CloudWatch Log untuk negara dan wilayah tertentu.

Negara atau wilayah	Kata kunci
Australia	dl# dl:, dlno# driver licence, driver license, driver permit, drivers lic., drivers licence, driver's licence, drivers license, driver's license, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit
Austria	führerschein, fuhrerschein, führerschein republik österreich, fuhrerschein republik osterreich
Belgium	fuehrerschein, fuehrerschein- nr, fuehrersc heinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrersch einnummer, führerscheinnummer, numéro permis conduire, permis de conduire, rijbewijs, rijbewijsnummer
Bulgaria	превозно средство, свидетелство за управление на моторно, свидетелство за управление на мпс, сумпс, шофьорска книжка
Canada	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit, permis de conduire
Croatia	vozačka dozvola
Cyprus	άδεια οδήγησης

Negara atau wilayah	Kata kunci
Czech Republic	číslo licence, číslo licence řidiče, číslo řidičského o průkazu, ovladače lic., povolení k jízdě, povolení řidiče, řidiči povolení, řidičský průkaz, řidičský průkaz
Denmark	kørekort, kørekortnummer
Estonia	juhi litsentsi number, juhiloa number, juhiluba, juhiluba number
Finland	ajokortin numero, ajokortti, förare lic., körkort, körkort nummer, kuljettaja lic., permis de conduire
France	permis de conduire
Germany	fuehrerschein, fuehrerschein- nr, fuehrerscheinnummer, fuhrerschein, fuhrerschein, fuhrerschein- nr, fuhrerschein- nr, fuhrerscheinnummer, fuhrerscheinnummer
Greece	δεία οδήγησης, adeia odigisis
Hungary	illesztőprogramok lic, jogosítvány, jogsi, licencszám, vezető engedély, vezetői engedély
Ireland	ceadúnas tiomána
Italy	patente di guida, patente di guida numero, patente guida, patente guida numero
Latvia	autovadītāja apliecība, licences numurs, vadītāja apliecība, vadītāja apliecības numurs, vadītāja atļauja, vadītāja licences numurs, vadītāji lic.
Lithuania	vairuotojo pažymėjimas

Negara atau wilayah	Kata kunci
Luxembourg	fahrerlaubnis, fñhrerschäin
Malta	liċenzja tas-sewqan
Netherlands	permis de conduire, rijbewijs, rijbewijsnummer
Poland	numer licencyjny, prawo jazdy, zezwolenie na prowadzenie
Portugal	carta de condução, carteira de habilitação, carteira de motorist, carteira habilitação, carteira motorist, licença condução, licença de condução, número de licença, número licença, permissão condução, permissão de condução
Romania	numărul permisului de conducere, permis de conducere
Slovakia	číslo licencie, číslo vodičského preukazu, ovládače lic., povolenia vodičov, povolenie jazdu, povolenie na jazdu, povolenie vodiča, vodičský preukaz
Slovenia	vozniško dovoljenje
Spain	carnet conducir, el carnet de conducir, licencia conducir, licencia de manejo, número carnet conducir, número de carnet de conducir, número de permiso conducir, número de permiso de conducir, número licencia conducir, número permiso conducir, permiso conducción, permiso conducir, permiso de conducción
Sweden	ajokortin numero, dlno# ajokortti, drivere lic., förare lic., körkort, körkort nummer, körkortsn ummer, kuljettajat lic.

Negara atau wilayah	Kata kunci
Britania Raya	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit
Amerika Serikat	dl#, dl:, dlno#, driver licence, driver licences, driver license, driver licenses, driver permit, drivers lic., drivers licence, driver's licence, drivers licences, driver's licences, drivers license, driver's license, drivers licenses, driver's licenses, drivers permit, driver's permit, drivers permit number, driving licence, driving license, driving permit

Kata kunci untuk nomor induk kependudukan

Untuk mendeteksi berbagai jenis nomor identifikasi nasional, CloudWatch Log membutuhkan kata kunci untuk berada di dekat nomor. Hal ini termasuk pengenalan Documento Nacional de Identidad (DNI) (Spain), kode French National Institute for Statistics and Economic Studies (INSEE), nomor German National Identity Card, dan nomor Registro Geral (RG) (Brazil).

Tabel berikut mencantumkan kata kunci yang dikenali CloudWatch Log untuk negara dan wilayah tertentu.

Negara atau wilayah	Kata kunci
Brazil	registro geral, rg
France	assurance sociale, carte nationale d'identité, cni, code sécurité sociale, French social security number, fssn#, insee, insurance

Negara atau wilayah	Kata kunci
	number, national id number, nationalid#, numéro d'assurance, sécurité sociale, sécurité sociale non., sécurité sociale numéro, social, social security, social security number, socialsecuritynumber, ss#, ssn, ssn#
Germany	ausweisnummer, id number, identification number, identity number, insurance number, personal id, personalausweis
Italy	codice fiscal, dati anagrafici, ehic, health card, health insurance card, p. iva, partita i.v.a., personal data, tax code, tessera sanitaria
Spain	dni, dni#, dninúmero#, documento nacional de identidad, identidad único, identidadúnico#, insurance number, national identification number, national identity, nationalid#, nationali dno#, número nacional identidad, personal identification number, personal identity no, unique identity number, uniqueid#

Kata kunci untuk nomor paspor

Untuk mendeteksi berbagai jenis nomor paspor, CloudWatch Log membutuhkan kata kunci untuk berada di dekat nomor. Tabel berikut mencantumkan kata kunci yang dikenali CloudWatch Log untuk negara dan wilayah tertentu.

Negara atau wilayah	Kata kunci
Canada	passeport, passeport#, passport, passport#, passportno, passportno#
France	numéro de passeport, passeport, passeport #, passeport #, passeportn °, passeport n °, passeportNon, passeport non

Negara atau wilayah	Kata kunci
Germany	ausstellungsdatum, ausstellungsort, geburtsdatum, passport, passports, reisepass, reisepassnr, reisepassnummer
Italy	italian passport number, numéro passeport, numéro passeport italien, passaporto, passaporto italiana, passaporto numero, passport number, repubblica italiana passaporto
Spain	españa pasaporte, libreta pasaporte, número pasaporte, pasaporte, passport, passport book, passport no, passport number, spain passport
Britania Raya	passeport #, passeport n °, passeportNon, passeport non, passeportn °, passport #, passport no, passport number, passport#, passportid
Amerika Serikat	passport, travel document

Kata kunci untuk nomor pokok wajib pajak

Untuk mendeteksi berbagai jenis identifikasi wajib pajak dan nomor referensi, CloudWatch Log membutuhkan kata kunci untuk berada di dekat angka-angka tersebut. Tabel berikut mencantumkan kata kunci yang dikenali CloudWatch Log untuk negara dan wilayah tertentu.

Negara atau wilayah	Kata kunci
Brazil	cadastro de pessoa física, cadastro de pessoa física, cadastro de pessoas físicas, cadastro de pessoas físicas, cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj, cpf

Negara atau wilayah	Kata kunci
France	numéro d'identification fiscale, tax id, tax identification number, tax number, tin, tin#
Germany	identifikationsnummer, steuer id, steueride ntifikationsnummer, steuernummer, tax id, tax identification number, tax number
Spain	cif, cif número, cifnúmero#, nie, nif, número de contribuyente, número de identidad de extranjero, número de identificación fiscal, número de impuesto corporativo, personal tax number, tax id, tax identification number, tax number, tin, tin#
Britania Raya	paye, tax id, tax id no., tax id number, tax identification, tax identification#, tax no., tax number, tax reference, tax#, taxid#, temporary reference number, tin, trn, unique tax reference, unique taxpayer reference, utr
Amerika Serikat	nomor identifikasi wajib pajak individu, itin, i.t.i.n.

Pengidentifikasi data ARNs untuk informasi identitas pribadi (PII)

Tabel berikut mencantumkan Nama Sumber Daya Amazon (ARNs) untuk pengidentifikasi data informasi identitas pribadi (PII) yang dapat Anda tambahkan ke kebijakan perlindungan data Anda.

Pengidentifikasi data PII ARNs

```
arn:aws:dataprotection::aws:data-identifier/Address
```

```
arn:aws:dataprotection::aws:data-identifier/CepCode-BR
```

```
arn:aws:dataprotection::aws:data-identifier/Cnpj-BR
```

Pengidentifikasi data PII ARNs

arn:aws:dataprotection::aws:data-identifier/CpfCode-BR

arn:aws:dataprotection::aws:data-identifier/DriversLicense-AT

arn:aws:dataprotection::aws:data-identifier/DriversLicense-AU

arn:aws:dataprotection::aws:data-identifier/DriversLicense-BE

arn:aws:dataprotection::aws:data-identifier/DriversLicense-BG

arn:aws:dataprotection::aws:data-identifier/DriversLicense-CA

arn:aws:dataprotection::aws:data-identifier/DriversLicense-CY

arn:aws:dataprotection::aws:data-identifier/DriversLicense-CZ

arn:aws:dataprotection::aws:data-identifier/DriversLicense-DE

arn:aws:dataprotection::aws:data-identifier/DriversLicense-DK

arn:aws:dataprotection::aws:data-identifier/DriversLicense-EE

arn:aws:dataprotection::aws:data-identifier/DriversLicense-ES

arn:aws:dataprotection::aws:data-identifier/DriversLicense-FI

arn:aws:dataprotection::aws:data-identifier/DriversLicense-FR

arn:aws:dataprotection::aws:data-identifier/DriversLicense-GB

arn:aws:dataprotection::aws:data-identifier/DriversLicense-GR

arn:aws:dataprotection::aws:data-identifier/DriversLicense-HR

arn:aws:dataprotection::aws:data-identifier/DriversLicense-HU

arn:aws:dataprotection::aws:data-identifier/DriversLicense-IE

arn:aws:dataprotection::aws:data-identifier/DriversLicense-IT

Pengidentifikasi data PII ARNs

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LT
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LU
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-LV
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-MT
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-NL
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-PL
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-PT
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-R0
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SE
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SI
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SK
```

```
arn:aws:dataprotection::aws:data-identifier/DriversLicense-US
```

```
arn:aws:dataprotection::aws:data-identifier/ElectoralRollNumber-GB
```

```
arn:aws:dataprotection::aws:data-identifier/EmailAddress
```

```
arn:aws:dataprotection::aws:data-identifier/IndividualTaxIdentificationNumber-US
```

```
arn:aws:dataprotection::aws:data-identifier/InseeCode-FR
```

```
arn:aws:dataprotection::aws:data-identifier/LatLong
```

```
arn:aws:dataprotection::aws:data-identifier/Name
```

Pengidentifikasi data PII ARNs

```
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-DE
```

```
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/NationalIdentificationNumber-IT
```

```
arn:aws:dataprotection::aws:data-identifier/NieNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/NifNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-CA
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-DE
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-FR
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-GB
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-IT
```

```
arn:aws:dataprotection::aws:data-identifier/PassportNumber-US
```

```
arn:aws:dataprotection::aws:data-identifier/PermanentResidenceNumber-CA
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-BR
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-DE
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-ES
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-FR
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-GB
```

Pengidentifikasi data PII ARNs

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-IT
```

```
arn:aws:dataprotection::aws:data-identifier/PhoneNumber-US
```

```
arn:aws:dataprotection::aws:data-identifier/PostalCode-CA
```

```
arn:aws:dataprotection::aws:data-identifier/RgNumber-BR
```

```
arn:aws:dataprotection::aws:data-identifier/SocialInsuranceNumber-CA
```

```
arn:aws:dataprotection::aws:data-identifier/Ssn-ES
```

```
arn:aws:dataprotection::aws:data-identifier/Ssn-US
```

```
arn:aws:dataprotection::aws:data-identifier/TaxId-DE
```

```
arn:aws:dataprotection::aws:data-identifier/TaxId-ES
```

```
arn:aws:dataprotection::aws:data-identifier/TaxId-FR
```

```
arn:aws:dataprotection::aws:data-identifier/TaxId-GB
```

```
arn:aws:dataprotection::aws:data-identifier/VehicleIdentificationNumber
```

```
arn:aws:dataprotection::aws:data-identifier/ZipCode-US
```

Pengidentifikasi data khusus

Topik

- [Apa itu pengidentifikasi data khusus?](#)
- [Kendala pengenalan data kustom](#)
- [Menggunakan pengidentifikasi data khusus di konsol](#)
- [Menggunakan pengidentifikasi data khusus dalam kebijakan perlindungan data Anda](#)

Apa itu pengidentifikasi data khusus?

Pengidentifikasi data kustom (CDIs) memungkinkan Anda menentukan ekspresi reguler kustom Anda sendiri yang dapat digunakan dalam kebijakan perlindungan data Anda. Dengan menggunakan pengidentifikasi data khusus, Anda dapat menargetkan kasus penggunaan informasi identitas pribadi (PII) khusus bisnis yang tidak dapat diberikan oleh pengidentifikasi data [terkelola](#). Misalnya, Anda dapat menggunakan pengenalan data khusus untuk mencari karyawan khusus perusahaan. IDs Pengidentifikasi data khusus dapat digunakan bersama dengan pengidentifikasi data terkelola.

Kendala pengenalan data kustom

CloudWatch Log pengidentifikasi data kustom memiliki batasan berikut:

- Maksimal 10 pengidentifikasi data kustom didukung untuk setiap kebijakan perlindungan data.
- Nama pengidentifikasi data kustom memiliki panjang maksimum 128 karakter. Karakter berikut didukung:
 - Alfanumerik: (A-Za-Z0-9)
 - Simbol: ('_' | '-')
- RegEx memiliki panjang maksimum 200 karakter. Karakter berikut didukung:
 - Alfanumerik: (A-Za-Z0-9)
 - Simbol: ('_' | '#' | '=' | '@' | '/' | ';' | ',' | '-' | '"')
 - RegEx karakter yang dipesan: ('^' | '\$' | '?' | '[' | ']' | '{' | '}' | '\\' | '*' | '+' | '.')
- Pengidentifikasi data kustom tidak dapat berbagi nama yang sama dengan pengenalan data terkelola.
- Pengidentifikasi data khusus dapat ditentukan dalam kebijakan perlindungan data tingkat akun atau dalam kebijakan perlindungan data tingkat grup log. Mirip dengan pengidentifikasi data terkelola, pengidentifikasi data kustom yang ditentukan dalam kebijakan tingkat akun bekerja dalam kombinasi dengan pengidentifikasi data kustom yang ditentukan dalam kebijakan tingkat grup log.

Menggunakan pengidentifikasi data khusus di konsol

Saat Anda menggunakan CloudWatch konsol untuk membuat atau mengedit kebijakan perlindungan data, untuk menentukan pengenalan data kustom, Anda cukup memasukkan nama dan ekspresi reguler untuk pengenalan data. Misalnya, Anda mungkin memasukkan **Employee_ID** nama dan **EmployeeID-\d{9}** sebagai ekspresi reguler. Ekspresi reguler ini akan mendeteksi dan menutupi peristiwa log dengan sembilan angka setelahnya **EmployeeID-**. Sebagai contoh, **EmployeeID-123456789**.

Menggunakan pengidentifikasi data khusus dalam kebijakan perlindungan data Anda

Jika Anda menggunakan AWS API atau AWS CLI untuk menentukan pengenalan data kustom, Anda harus menyertakan nama pengenalan data dan ekspresi reguler dalam kebijakan JSON yang digunakan untuk menentukan kebijakan perlindungan data. Kebijakan perlindungan data berikut mendeteksi dan menutupi peristiwa log yang membawa karyawan khusus perusahaan. IDs

1. Buat Configuration blok dalam kebijakan perlindungan data Anda.
2. Masukkan a Name untuk pengenalan data kustom Anda. Misalnya, **EmployeeId**.
3. Masukkan a Regex untuk pengenalan data kustom Anda. Misalnya, **EmployeeID-\d{9}**. Ekspresi reguler ini akan cocok dengan peristiwa log EmployeeID- yang berisi sembilan digit EmployeeID- setelahnya. Sebagai contoh, EmployeeID-123456789.
4. Lihat pengenalan data kustom berikut dalam pernyataan kebijakan.

```
{
  "Name": "example_data_protection_policy",
  "Description": "Example data protection policy with custom data identifiers",
  "Version": "2021-06-01",
  "Configuration": {
    "CustomDataIdentifier": [
      {
        "Name": "EmployeeId",
        "Regex": "EmployeeId-\\d{9}"
      }
    ]
  },
  "Statement": [
    {
      "Sid": "audit-policy",
      "DataIdentifier": [
        "EmployeeId"
      ],
      "Operation": {
        "Audit": {
          "FindingsDestination": {
            "S3": {
              "Bucket": "EXISTING_BUCKET"
            }
          }
        }
      }
    },
    {
      "Sid": "redact-policy",
```



```
    "DataIdentifier": [
      "EmployeeId"
    ],
    "Operation": {
      "Deidentify": {
        "MaskConfig": {
          }
        }
      }
    }
  ]
}
```

5. (Opsional) Lanjutkan untuk menambahkan pengidentifikasi data khusus tambahan ke Configuration blok sesuai kebutuhan. Kebijakan perlindungan data saat ini mendukung maksimal 10 pengidentifikasi data kustom.

Ubah log selama konsumsi

Dengan transformasi dan pengayaan log, Anda dapat menormalkan semua log Anda dalam format yang konsisten dan kaya konteks pada saat masuk ke dalam Log. CloudWatch [Anda dapat menambahkan struktur ke log Anda dengan menggunakan templat yang telah dikonfigurasi sebelumnya untuk AWS layanan umum seperti AWS WAF dan Amazon Route 53, atau membuat transformator khusus dengan parser asli seperti Grok.](#) Anda juga dapat mengganti nama atribut yang ada dan menambahkan metadata tambahan ke log Anda seperti ID akun, dan Wilayah.

Transformasi log membantu menyederhanakan dan mempersingkat kueri log Anda di seluruh aplikasi Anda, dan membantu menyederhanakan pembuatan peringatan di log Anda. Fitur ini menyediakan transformasi untuk jenis log umum dengan templat out-of-the-box transformasi untuk sumber AWS log utama seperti log VPC Flow, Route 53, dan. Amazon RDS for PostgreSQL Anda dapat menggunakan templat transformasi yang telah dikonfigurasi sebelumnya atau membuat transformer khusus yang sesuai dengan kebutuhan Anda.

Transformasi log membantu Anda mengelola log yang dipancarkan dari berbagai sumber yang sangat bervariasi dalam format dan nama atribut.

Setelah Anda membuat transformator, peristiwa log yang dicerna dikonversi dan disimpan dalam format standar. Anda dapat memanfaatkan log yang diubah ini untuk mempercepat pengalaman analitik Anda dengan fitur-fitur berikut:

- [Indeks lapangan](#)
- [CloudWatch Logs Insights menemukan bidang](#)
- [Fleksibilitas dalam mengkhawatirkan menggunakan filter metrik](#)
- [Penerusan melalui filter berlangganan](#)
- Membuat data metrik dari peristiwa log dengan [Contributor Insights](#), di mana Anda dapat memilih agar aturan Contributor Insights mengevaluasi peristiwa log baik sebelum atau sesudah diubah.

Transformasi hanya terjadi selama konsumsi log. Anda tidak dapat mengubah peristiwa log yang telah dicerna. Transformasi tidak reversibel. Log asli dan yang diubah disimpan di CloudWatch Log dengan kebijakan retensi yang sama. Transformasi log dan kemampuan pengayaan termasuk dalam harga konsumsi kelas log Standar yang ada. Biaya penyimpanan log akan didasarkan pada ukuran log setelah transformasi, yang mungkin melebihi volume log asli.

⚠ Important

Setelah peristiwa log diubah, Anda harus menggunakan kueri Wawasan CloudWatch Log untuk melihat versi log yang diubah. [FilterLogEvents](#) Tindakan [GetLogEvents](#) dan hanya mengembalikan versi asli dari peristiwa log, sebelum diubah.

⚠ Important

Meskipun satu peristiwa log PutLogEvents memungkinkan hingga 1MB, transformasi log hanya dapat menangani peristiwa log dengan ukuran kurang dari 512kb. Setiap peristiwa log yang lebih besar dari 512kb akan gagal dalam transformasi dan memancarkan kesalahan. Ukuran total masih PutLogEvents bisa melebihi 512kb.

Selain mengubah ke format yang berbeda, Anda juga dapat memperkaya log Anda dengan konteks tambahan, seperti ID akun, Wilayah, dan kata kunci. Ini diekstraksi dari nama grup log dan dari kata kunci statis.

Transformasi log membantu Anda dengan log yang dipancarkan dari berbagai sumber yang sangat bervariasi dalam format dan nama atribut.

Transformasi log dan pengayaan hanya didukung untuk grup log di kelas log Standar.

Anda dapat membuat transformer untuk grup log individual, dan Anda juga dapat membuat transformator tingkat akun yang berlaku untuk semua atau banyak grup log di akun Anda. Jika grup log memiliki transformator tingkat grup log, transformator tersebut akan mengganti transformator tingkat akun apa pun yang akan berlaku untuk grup log tersebut.

Topik

- [Buat dan kelola trafo log](#)
- [Prosesor tipe parser yang dapat dikonfigurasi](#)
- [Prosesor bawaan untuk AWS log penjual otomatis](#)
- [Prosesor mutasi string](#)
- [JSON bermutasi prosesor](#)
- [Prosesor konverter tipe data](#)

- [Metrik dan kesalahan transformasi](#)

Buat dan kelola trafo log

Trafo log mencakup satu atau lebih prosesor yang berada dalam pipa logis bersama-sama. Setiap prosesor diterapkan ke peristiwa log, satu demi satu dalam urutan yang tercantum dalam konfigurasi transformator.

Beberapa prosesor adalah tipe parser. Setiap transformator harus memiliki setidaknya satu parser, dan prosesor pertama dalam transformator harus berupa parser.

Beberapa parser adalah parser bawaan yang dikonfigurasi untuk jenis log AWS vended tertentu.

Jenis prosesor lainnya adalah mutator string, mutator JSON, dan prosesor data.

Anda dapat membuat transformer untuk grup log individual, dan Anda juga dapat membuat transformator tingkat akun yang berlaku untuk semua atau banyak grup log di akun Anda. Jika grup log memiliki transformator tingkat grup log, transformator tersebut akan mengganti transformator tingkat akun apa pun yang akan berlaku untuk grup log tersebut. Anda dapat memiliki sebanyak 20 trafo tingkat akun di Wilayah di akun Anda.

Anda harus mengikuti panduan ini saat membuat transformator:

- Jika Anda menyertakan parser pra-konfigurasi untuk jenis log AWS vended, itu harus menjadi prosesor pertama yang tercantum dalam transformator. Anda hanya dapat memasukkan satu prosesor seperti itu ke dalam transformator.
- Anda hanya dapat memasukkan satu `grok` prosesor dalam transformator.
- Anda harus memiliki setidaknya satu prosesor tipe parser dalam transformator. Anda dapat menyertakan sebanyak lima prosesor tipe parser. Batas lima ini mencakup parser bawaan dan parser yang dapat dikonfigurasi.
- Anda dapat memiliki sebanyak 20 prosesor dalam transformator.
- Anda hanya dapat menyertakan satu prosesor `AddKeys` dalam transformator.
- Anda hanya dapat menyertakan satu prosesor `CopyValue` dalam transformator.
- Setiap transformator dapat mengekstrak hingga 200 bidang dari peristiwa log.
- Setiap peristiwa log HARUS di bawah 512KB. Ukuran total peristiwa log masih bisa melebihi 512KB.

Topik

- [Buat kebijakan transformator tingkat akun](#)
- [Mengedit atau menghapus kebijakan transformator tingkat akun](#)
- [Buat transformator log-group-level log dari awal](#)
- [Buat log-group-level transformator dengan menyalin yang sudah ada](#)
- [Edit log-group-level transformator](#)
- [Hapus log-group-level transformator](#)

Buat kebijakan transformator tingkat akun

Gunakan langkah-langkah di bagian ini untuk membuat kebijakan transformator yang berlaku untuk semua grup log di akun, atau ke beberapa grup log yang memiliki nama grup log yang dimulai dengan string (awalan) yang sama. Anda dapat memiliki sebanyak 20 kebijakan transformator tingkat akun di suatu Wilayah.

Anda tidak dapat membuat dua kebijakan transformator di Wilayah yang sama yang menggunakan awalan yang sama atau memiliki satu awalan di dalam yang lain. Misalnya, jika Anda membuat satu kebijakan transformator untuk awalan string `/aws/lambda`, Anda tidak dapat membuat yang lain dengan awalan `/aws`. Tapi Anda bisa memiliki satu transformator untuk `/aws/lambda` dan satu lagi untuk `/aws/waf`.

Untuk membuat kebijakan transformator tingkat akun

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Pengaturan dan kemudian pilih tab Log.
3. Di bagian Kebijakan Transformer untuk akun, pilih Buat kebijakan transformator.
4. Untuk nama kebijakan Transformer, masukkan nama untuk poiiicy baru Anda.
5. Untuk Pilih grup log, lakukan salah satu hal berikut:
 - Pilih Semua grup log standar agar kebijakan transformator berlaku untuk semua grup log Kelas Standar di akun.
 - pilih Grup log menurut awalan yang cocok untuk menerapkan kebijakan ke subset grup log yang semuanya memiliki nama yang dimulai dengan string yang sama. Kemudian, masukkan awalan untuk grup log ini di Kriteria pemilihan.
6. Di area Select parsers, gunakan Parser untuk memilih parser yang akan disertakan dalam transformator Anda.

Jika ini adalah parser pra-konfigurasi untuk jenis log AWS vended, Anda tidak perlu menentukan konfigurasi apa pun untuknya.

Jika itu adalah parser yang berbeda, Anda perlu menentukan konfigurasinya. Untuk informasi selengkapnya, lihat informasi untuk prosesor tersebut di [Prosesor tipe parser yang dapat dikonfigurasi](#).

7. Untuk menambahkan prosesor lain, pilih Pilih prosesor. Kemudian pilih prosesor yang Anda inginkan di kotak Prosesor, dan isi parameter konfigurasi.

Ingat bahwa prosesor beroperasi pada peristiwa log dalam urutan yang Anda tambahkan ke transformator.

8. (Opsional) Untuk menambahkan prosesor tambahan, pilih + Prosesor dan ulangi langkah sebelumnya.
9. (Opsional) Kapan saja, Anda dapat menguji transformator yang telah Anda buat sejauh ini pada peristiwa log sampel. Untuk melakukannya, lakukan salah satu hal berikut di bagian pratinjau Transformer:
 - Pilih sebanyak lima grup log di Pilih grup log dan kemudian pilih Muat peristiwa log terbaru. Kemudian pilih Test transformator.
 - Salin peristiwa log langsung ke peristiwa log Contoh dan kemudian pilih Transformator uji.

Versi log yang diubah kemudian muncul.

10. Setelah selesai menambahkan prosesor dan puas dengan pengujian pada log sampel, pilih Simpan.
11. Setelah selesai, pilih Buat.

Mengedit atau menghapus kebijakan transformator tingkat akun

Gunakan langkah-langkah di bagian ini untuk mengedit atau menghapus kebijakan transformator tingkat akun.

Untuk mengedit atau menghapus kebijakan transformator tingkat akun

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Pengaturan dan kemudian pilih tab Log.
3. Di bagian Kebijakan akun Transformer, pilih Kelola.

4. Pilih tombol berdasarkan kebijakan transformator yang ingin Anda kelola, lalu pilih Edit atau Hapus.

Jika Anda mengedit kebijakan, lihat langkah 5-11 [Prosesor tipe parser yang dapat dikonfigurasi](#) untuk melihat opsi Anda.

Buat transformator log-group-level log dari awal

Gunakan langkah-langkah ini untuk membuat log-group-level transformator dari awal.

Untuk menggunakan konsol untuk membuat transformator log untuk grup log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Pilih grup log yang ingin Anda buat transformator.
4. Pilih tab Transformer. Anda mungkin harus menggulir daftar tab ke kanan untuk melihatnya.
5. Pilih Buat transformator.
6. Dalam kotak Pilih parser, pilih parser untuk disertakan dalam transformator Anda.

Jika ini adalah parser pra-konfigurasi untuk jenis log AWS vended, Anda tidak perlu menentukan konfigurasi apa pun untuknya.

Jika itu adalah parser yang berbeda, Anda perlu menentukan konfigurasinya. Untuk informasi selengkapnya, lihat informasi untuk prosesor tersebut di [Prosesor tipe parser yang dapat dikonfigurasi](#).

7. Untuk menambahkan prosesor lain, pilih + Tambahkan prosesor. Kemudian pilih prosesor yang Anda inginkan di kotak Pilih prosesor, dan isi parameter konfigurasi.

Ingat bahwa prosesor beroperasi pada peristiwa log dalam urutan yang Anda tambahkan ke transformator.

8. (Opsional) Kapan saja, Anda dapat menguji transformator yang telah Anda buat sejauh ini pada peristiwa log sampel. Untuk melakukan hal itu, lakukan hal berikut:
 - Di bagian pratinjau Transformasi, pilih Muat log sampel untuk memuat peristiwa log sampel dari grup log tempat transformator ini digunakan, atau tempelkan peristiwa log ke dalam kotak teks.

Pilih Trafo uji. Versi log yang diubah muncul

9. Setelah selesai menambahkan prosesor dan puas dengan pengujian pada log sampel, pilih Simpan.

Untuk menggunakan AWS CLI untuk membuat transformator log dari awal

- Gunakan perintah `aws logs put-transformer`. Saat menggunakan `parseJSON` sebagai prosesor pertama, Anda harus mengurai seluruh peristiwa log menggunakan `@message` sebagai bidang sumber. Setelah penguraian JSON awal, Anda kemudian dapat memanipulasi bidang tertentu di prosesor berikutnya. Berikut ini adalah contoh yang menciptakan transformator yang mencakup `parseJSON` dan `addKeys` prosesor:

```
aws logs put-transformer \
  --transformer-config '[{"parseJSON":{"source":"@message"}}, {"addKeys":
{"entries":[{"key":"metadata.transformed_in","value":"CloudWatchLogs"},
{"key":"feature","value":"Transformation"}]}], {"trimString":{"withKeys":
["status"]}]}' \
  --log-group-identifier my-log-group-name
```

Buat log-group-level transformator dengan menyalin yang sudah ada

Anda dapat menggunakan konsol untuk menyalin konfigurasi JSON dari transformator yang ada. Anda kemudian dapat menggunakan kode itu untuk membuat transformator identik dengan menggunakan AWS CLI, atau Anda dapat memodifikasi konfigurasi terlebih dahulu.

Untuk membuat transformator log dengan menyalin yang sudah ada

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Pilih grup log yang memiliki transformator yang ingin Anda salin.
4. Pilih tab Transformasi. Anda mungkin harus menggulir daftar tab ke kanan untuk melihatnya.
5. Pilih Kelola transformator.
6. Pilih Salin transformator. Ini menyalin transformator JSON ke clipboard Anda.
7. Buat file dan tempel dalam konfigurasi transformator. Dalam contoh ini kita akan memanggil file `CopiedTransformer.json`
8. Gunakan AWS CLI untuk membuat transformator baru dengan konfigurasi itu.


```
aws logs put-transformer --log-group-identifier my-log-group-name \  
--transformer-config file://CopiedTransformer.json
```

Edit log-group-level transformator

Gunakan langkah-langkah ini untuk mengedit transformator log yang ada.

Untuk mengedit transformator log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Pilih grup log yang memiliki transformator yang ingin Anda edit.
4. Pilih tab Transformasi. Anda mungkin harus menggulir daftar tab ke kanan untuk melihatnya.
5. Pilih Kelola transformator.
6. Di bagian Parser dan Prosesor, buat perubahan Anda.
7. Untuk menambahkan prosesor lain, pilih + Tambah Prosesor. Kemudian pilih prosesor yang Anda inginkan di kotak Prosesor, dan isi parameter konfigurasi.

Ingat bahwa prosesor beroperasi pada peristiwa log dalam urutan yang Anda tambahkan ke transformator.

8. (Opsional) Kapan saja, Anda dapat menguji transformator yang telah Anda buat sejauh ini pada peristiwa log sampel. Untuk melakukan hal itu, lakukan hal berikut:
 - Di bagian Pratinjau Transformasi, pilih Muat Log Sampel untuk memuat peristiwa log sampel dari grup log tempat transformator ini digunakan, atau tempelkan peristiwa log ke dalam kotak teks.

Pilih Test Transformation. Versi log yang diubah muncul

9. Setelah selesai menambahkan prosesor dan puas dengan pengujian pada log sampel, pilih Simpan.

Hapus log-group-level transformator

Gunakan langkah-langkah ini untuk menghapus transformator log.

Untuk menghapus transformator log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih Log, Grup log.
3. Pilih grup log yang memiliki transformator yang ingin Anda edit.
4. Pilih tab Transformasi. Anda mungkin harus menggulir daftar tab ke kanan untuk melihatnya.
5. Pilih Hapus.
6. Di kotak konfirmasi, pilih Hapus Kebijakan.

Prosesor tipe parser yang dapat dikonfigurasi

Bagian ini berisi informasi tentang prosesor parser data yang dapat dikonfigurasi yang dapat Anda gunakan dalam transformator peristiwa log.

Daftar Isi

- [parseJSON](#)
- [grok](#)
 - [Contoh Grok](#)
 - [Contoh 1: Gunakan grok untuk mengekstrak bidang dari log yang tidak terstruktur](#)
 - [Contoh 2: Gunakan grok dalam kombinasi dengan ParseJson untuk mengekstrak bidang dari peristiwa log JSON](#)
 - [Contoh 3: Pola Grok dengan anotasi bertitik di FIELD_NAME](#)
 - [Pola grok yang didukung](#)
 - [Contoh format log umum](#)
 - [Contoh log Apache](#)
 - [Contoh log NGINX](#)
 - [Contoh log Protokol Syslog \(RFC 5424\)](#)
- [csv](#)
- [parseKeyValue](#)

parseJSON

Prosesor ParseJSON mem-parsing peristiwa log JSON dan menyisipkan pasangan nilai kunci JSON yang diekstraksi di bawah tujuan. Jika Anda tidak menentukan tujuan, prosesor menempatkan pasangan kunci-nilai di bawah simpul akar. Saat menggunakan parseJSON sebagai prosesor pertama, Anda harus mengurai seluruh peristiwa log menggunakan @message sebagai bidang sumber. Setelah penguraian JSON awal, Anda kemudian dapat memanipulasi bidang tertentu di prosesor berikutnya.

@messageKonten asli tidak diubah, kunci baru ditambahkan ke pesan.

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Jalur ke bidang dalam peristiwa log yang akan diuraikan. Gunakan notasi titik untuk mengakses bidang anak. Sebagai contoh, <code>store.book</code> .	Tidak	@message	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
tujuan	Bidang tujuan dari JSON yang diuraikan	Tidak	Parent JSON node	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3

Contoh

Misalkan peristiwa log tertelan terlihat seperti ini:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Kemudian jika kita memiliki prosesor ParseJson ini:

```
[
  {
    "parseJSON": {
      "destination": "new_key"
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "new_key": {
    "outer_key": {
      "inner_key": "inner_value"
    }
  }
}
```

grok

Gunakan prosesor grok untuk mengurai dan menyusun data yang tidak terstruktur menggunakan pencocokan pola. Prosesor ini juga dapat mengekstrak bidang dari pesan log.

Bidang	Deskripsi	Diperlukan?	Default	Batas	Catatan
sumber	Jalur bidang untuk menerapkan pencocokan Grok pada	Tidak	@message	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3	
match	Pola grok untuk mencocokkan dengan peristiwa log	Ya		Panjang maksimal: 512 Pola grok maksimum: 20	Lihat semua pola Grok yang didukung

Bidang	Deskripsi	Diperlukan?	Default	Batas	Catatan
				Beberapa jenis pola grok memiliki batas penggunaan individual. Setiap kombinasi dari pola berikut dapat digunakan sebanyak lima kali: {URI, URIPARAM, URIPATHPARAM, SPACE, DATA, GREEDYDATA, GREEDYDATA_MULTILINE} Pola Grok tidak mendukung konversi tipe. Untuk pola format log umum (APACHE_ACCESS_LOG, NGINX_ACCESS_LOG, SYSLOG5424), hanya pola DATA, GREEDYDATA, atau GREEDYDATA_MULTILINE yang didukung untuk disertakan setelah pola log umum.	

Struktur Pola Grok

Ini adalah struktur pola grok yang didukung:

```
%{PATTERN_NAME:FIELD_NAME}
```

- **PATTERN_NAME:** Mengacu pada ekspresi reguler yang telah ditentukan sebelumnya untuk mencocokkan jenis data tertentu. Hanya [pola grok](#) yang telah ditentukan yang didukung. Membuat pola kustom tidak diperbolehkan.
- **FIELD_NAME:** Menetapkan nama ke nilai yang diekstraksi. **FIELD_NAME** bersifat opsional, tetapi jika Anda tidak menentukan nilai ini maka data yang diekstraksi akan dihapus dari peristiwa log yang diubah. Jika **FIELD_NAME** menggunakan notasi putus-putus (misalnya, "parent.child"), itu dianggap sebagai jalur JSON.
- **Jenis Konversi:** Konversi tipe eksplisit tidak didukung. Gunakan [TypeConverter prosesor](#) untuk mengonversi tipe data dari nilai apa pun yang diekstraksi oleh grok.

Untuk membuat ekspresi pencocokan yang lebih kompleks, Anda dapat menggabungkan beberapa pola grok. Sebanyak 20 pola grok dapat digabungkan untuk mencocokkan peristiwa log. Misalnya, kombinasi pola ini `%{NUMBER:timestamp} [%{NUMBER:db} %{IP:client_ip}: %{NUMBER:client_port}] %{GREEDYDATA:data}` dapat digunakan untuk mengekstrak bidang dari entri log lambat Redis seperti ini:

```
1629860738.123456 [0 127.0.0.1:6379] "SET" "key1" "value1"
```

Contoh Grok

Contoh 1: Gunakan grok untuk mengekstrak bidang dari log yang tidak terstruktur

Log sampel:

```
293750 server-01.internal-network.local OK "[Thread-000] token generated"
```

Transformator yang digunakan:

```
[
  {
    "grok": {
      "match": "%{NUMBER:version} %{HOSTNAME:hostname} %{NOTSPACE:status}
%{QUOTEDSTRING:logMsg}"
```

```
    }  
  }  
]
```

Output:

```
{  
  "version": "293750",  
  "hostname": "server-01.internal-network.local",  
  "status": "OK",  
  "logMsg": "[Thread-000] token generated"  
}
```

Log sampel:

```
23/Nov/2024:10:25:15 -0900 172.16.0.1 200
```

Transformator yang digunakan:

```
[  
  {  
    "grok": {  
      "match": "%{HTTPDATE:timestamp} %{IPORHOST:clientip}  
%{NUMBER:response_status}"  
    }  
  }  
]
```

Output:

```
{  
  "timestamp": "23/Nov/2024:10:25:15 -0900",  
  "clientip": "172.16.0.1",  
  "response_status": "200"  
}
```

Contoh 2: Gunakan grok dalam kombinasi dengan ParseJson untuk mengekstrak bidang dari peristiwa log JSON

Contoh log:

```
{
  "timestamp": "2024-11-23T16:03:12Z",
  "level": "ERROR",
  "logMsg": "GET /page.html HTTP/1.1"
}
```

Transformator yang digunakan:

```
[
  {
    "parseJSON": {}
  },
  {
    "grok": {
      "source": "logMsg",
      "match": "%{WORD:http_method} %{NOTSPACE:request} HTTP/"
    }
  }
]
```

Output:

```
{
  "timestamp": "2024-11-23T16:03:12Z",
  "level": "ERROR",
  "logMsg": "GET /page.html HTTP/1.1",
  "http_method": "GET",
  "request": "/page.html",
  "http_version": "1.1"
}
```

Contoh 3: Pola Grok dengan anotasi bertitik di FIELD_NAME

Contoh log:

```
192.168.1.1 GET /index.html?param=value 200 1234
```

Transformator yang digunakan:

```
[
```



```
{
  "grok": {
    "match": "%{IP:client.ip} %{WORD:method} %{URIPATHPARAM:request.uri}
%{NUMBER:response.status} %{NUMBER:response.bytes}"
  }
}
```

Output:

```
{
  "client": {
    "ip": "192.168.1.1"
  },
  "method": "GET",
  "request": {
    "uri": "/index.html?param=value"
  },
  "response": {
    "status": "200",
    "bytes": "1234"
  }
}
```

Pola grok yang didukung

Tabel berikut mencantumkan pola yang didukung oleh grok prosesor.

Pola grok umum

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
USERNAME atau USER	Cocokkan satu atau lebih karakter yang dapat mencakup huruf kecil (a-z), huruf besar (A-Z), digit (0-9), titik (.), garis bawah (_), atau tanda hubung (-)	20	Masukan: user123.name-TEST Pola: %{USERNAME:name} Keluaran: {"name": "user123.name-TEST"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
INT	Cocokkan tanda plus atau minus opsional diikuti oleh satu atau lebih digit.	20	Masukan: -456 Pola: <code>%{INT:num}</code> Keluaran: <code>{"num": "-456"}</code>
BASE10JUM LAH	Cocokkan bilangan bulat atau angka floating-point dengan tanda opsional dan titik desimal	20	Masukan: -0.67 Pola: <code>%{BASE10NUM:num}</code> Keluaran: <code>{"num": "-0.67"}</code>
BASE16JUM LAH	Cocokkan angka desimal dan heksadesimal dengan tanda opsional (+ atau -) dan awalan 0x opsional	20	Masukan: +0xA1B2 Pola: <code>%{BASE16NUM:num}</code> Keluaran: <code>{"num": "+0xA1B2"}</code>
POSINT	Cocokkan seluruh bilangan bulat positif tanpa nol di depan, terdiri dari satu atau lebih digit (1-9 diikuti oleh 0-9)	20	Masukan: 123 Pola: <code>%{POSINT:num}</code> Keluaran: <code>{"num": "123"}</code>
NONNEGINT	Cocokkan setiap bilangan bulat (terdiri dari satu atau lebih digit 0-9) termasuk nol dan angka dengan angka nol di depan.	20	Masukan: 007 Pola: <code>%{NONNEGINT:num}</code> Keluaran: <code>{"num": "007"}</code>
KATA	Cocokkan seluruh kata yang terdiri dari satu atau lebih karakter kata (<code>\w</code>), termasuk huruf, digit, dan garis bawah	20	Masukan: user_123 Pola: <code>%{WORD:user}</code> Keluaran: <code>{"user": "user_123"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
NOTSPACE	Cocokkan satu atau lebih karakter non-spasi.	5	Masukan: hello_world123 Pola: %{NOTSPACE:msg} Keluaran: {"msg": "hello_world123"}
LUAR ANGKASA	Cocokkan nol atau lebih karakter spasi putih.	5	Masukan: " " Pola: %{SPACE:extra} Keluaran: {"extra": " "}
DATA	Cocokkan karakter apa pun (kecuali baris baru) nol kali atau lebih, tidak serakah.	5	Masukan: abc def ghi Pola: %{DATA:x} %{DATA:y} Keluaran: {"x": "abc", "y": "def ghi"}
DATA KESERAKAHAN	Cocokkan karakter apa pun (kecuali baris baru) nol atau lebih kali, serakah.	5	Masukan: abc def ghi Pola: %{GREEDYDATA:x} %{GREEDYDATA:y} Keluaran: {"x": "abc def", "y": "ghi"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
GREEDYDATA_MULTILINE	Cocokkan karakter apa pun (termasuk baris baru) nol kali atau lebih, serakah.	1	Masukan: <pre>abc def ghi</pre> Pola: <code>%{GREEDYDATA_MULTILINE:data}</code> Keluaran: <code>{"data": "abc\ndef\nghi"}</code>
STRING_KUTIPAN	Cocokkan string yang dikutip (tanda kutip tunggal atau ganda) dengan karakter yang lolos.	20	Masukan: "Hello, world!" Pola: <code>%{QUOTEDSTRING:msg}</code> Keluaran: <code>{"msg": "Hello, world!"}</code>
UUID	Cocokkan format UUID standar: 8 karakter heksadesimal, diikuti oleh tiga kelompok 4 karakter heksadesimal, dan diakhiri dengan 12 karakter heksadesimal, semuanya dipisahkan oleh tanda hubung.	20	Masukan: 550e8400-e29b-41d4-a716-446655440000 Pola: <code>%{UUID:id}</code> Keluaran: <code>{"id": "550e8400-e29b-41d4-a716-446655440000"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
GUCI	Mencocokkan sintaks URN (Uniform Resource Name).	20	Masukan: urn:isbn:0451450523 Pola: %{URN:urn} Keluaran: {"urn": "urn:isbn:0451450523"}

AWS pola grok

Pola	Deskripsi	Batas pola maksimum	Contoh
ARN	Mencocokkan AWS Amazon Resource Names (ARNs), menangkap partisi (aws,aws-cn, atauaws-us-gov), layanan, Wilayah, ID akun, dan hingga 5 pengidentifikasi sumber daya hierarkis yang dipisahkan oleh garis miring. Itu tidak akan cocok dengan informasi ARNs yang hilang di antara titik dua.	5	Masukan: arn:aws:iam:us-east-1:123456789012:user/johndoe Pola: %{ARN:arn} Keluaran: {"arn": "arn:aws:iam:us-east-1:123456789012:user/johndoe"}

Pola grok jaringan

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
CISKOMAK	Cocokkan alamat MAC dalam format heksadesimal 4-4-4.	20	Masukan: 0123.4567.89AB

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
			Pola: <code>%{CISCOMAC:MacAddress}</code> Keluaran: <code>{"MacAddress": "0123.4567.89AB"}</code>
WINDOWSMAC	Cocokkan alamat MAC dalam format heksadesimal dengan tanda hubung	20	Masukan: 01-23-45-67-89-AB Pola: <code>%{WINDOWSMAC:MacAddress}</code> Keluaran: <code>{"MacAddress": "01-23-45-67-89-AB"}</code>
COMMONMAC	Cocokkan alamat MAC dalam format heksadesimal dengan titik dua.	20	Masukan: 01:23:45:67:89:AB Pola: <code>%{COMMONMAC:MacAddress}</code> Keluaran: <code>{"MacAddress": "01:23:45:67:89:AB"}</code>
MAC	Cocokkan dengan salah satu pola grok CISCOMAC, WINDOWSMAC atau COMMONMAC	20	Masukan: 01:23:45:67:89:AB Pola: <code>%{MAC:m1}</code> Keluaran: <code>{"m1": "01:23:45:67:89:AB"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
IPV6	Mencocokkan IPv6 alamat, termasuk formulir terkompresi dan alamat yang IPv4 dipetakan IPv6 .	5	Masukan: 2001:db8:3333:4444:5555:6666:7777:8888 Pola: %{IPV6:ip} Keluaran: {"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}
IPV4	Cocokkan IPv4 alamat.	20	Masukan: 192.168.0.1 Pola: %{IPV4:ip} Keluaran: {"ip": "192.168.0.1"}
IP	Cocokkan IPv6 alamat yang didukung oleh% {IPv6} atau IPv4 alamat yang didukung oleh% {IPv4}	5	Masukan: 192.168.0.1 Pola: %{IP:ip} Keluaran: {"ip": "192.168.0.1"}
HOSTNAME atau HOST	Cocokkan nama domain, termasuk subdomain	5	Masukan: server-01.internal-network.local Pola: %{HOST:host} Keluaran: {"host": "server-01.internal-network.local"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
IPORHOST	Cocokkan nama host atau alamat IP	5	Masukan: 2001:db8:3333:4444:5555:6666:7777:8888 Pola: %{IPORHOST:ip} Keluaran: {"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}
HOSTPORT	Cocokkan alamat IP atau nama host yang didukung oleh pola% {IPORHOST} diikuti oleh titik dua dan nomor port, menangkap port sebagai "PORT" dalam output.	5	Masukan: 192.168.1.1:8080 Pola: %{HOSTPORT:ip} Keluaran: {"ip": "192.168.1.1:8080", "PORT": "8080"}
URIHOST	Cocokkan alamat IP atau nama host yang didukung oleh pola% {IPORHOST}, secara opsional diikuti oleh titik dua dan nomor port, menangkap port sebagai "port" jika ada.	5	Masukan: example.com:443 10.0.0.1 Pola: %{URIHOST:host} %{URIHOST:ip} Keluaran: {"host": "example.com:443", "port": "443", "ip": "10.0.0.1"}

Pola grok jalur

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
UNIXPATH	Cocokkan jalur URL, berpotensi termasuk parameter kueri.	20	Masukan: /search?q=regex Pola: %{UNIXPATH:path} Keluaran: {"path":"/search?q=regex"}
WINPATH	Cocokkan jalur file Windows.	5	Masukan: C:\Users\John\Documents\file.txt Pola: %{WINPATH:path} Keluaran: {"path": "C:\\Users\\John\\Documents\\file.txt"}
PATH	Cocokkan dengan URL atau jalur file Windows	5	Masukan: /search?q=regex Pola: %{PATH:path} Keluaran: {"path":"/search?q=regex"}
TTY	Cocokkan jalur perangkat Unix untuk terminal dan terminal semu.	20	Masukan: /dev/tty1 Pola: %{TTY:path} Keluaran: {"path":"/dev/tty1"}
URIPROTO	Mencocokkan huruf, secara opsional diikuti oleh karakter plus (+) dan huruf tambahan atau karakter plus (+)	20	Masukan: web+transformer

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
			Pola: <code>%{URIPROTO:protocol}</code> Keluaran: <code>{"protocol": "web+transformer"}</code>
URIPATH	Cocokkan komponen jalur URI	20	Masukan: <code>/category/sub-category/product_name</code> Pola: <code>%{URIPATH:path}</code> Keluaran: <code>{"path": "/category/sub-category/product_name"}</code>
URIPARAM	Cocokkan parameter kueri URL	5	Masukan: <code>?param1=value1&param2=value2</code> Pola: <code>%{URIPARAM:url}</code> Keluaran: <code>{"url": "?param1=value1&param2=value2"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
URIPATHPARAM	Cocokkan jalur URI secara opsional diikuti oleh parameter kueri	5	Masukan: /category/sub-category/product?id=12345&color=red Pola: %{URIPATHPARAM:path} Keluaran: {"path":"/category/sub-category/product?id=12345&color=red"}
URI	Cocokkan URI lengkap	5	Masukan: https://user:password@example.com/path/to/resource?param1=value1&param2=value2 Pola: %{URI:uri} Keluaran: {"path":"https://user:password@example.com/path/to/resource?param1=value1&param2=value2"}

Pola grok tanggal dan waktu

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
MONTH	Cocokkan nama bulan bahasa Inggris lengkap	20	Masukan: Jan Pola: %{MONTH:month}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
	atau disingkat sebagai kata utuh		Keluaran: {"month": "Jan"} Masukan: January Pola: %{MONTH:month} Keluaran: {"month": "January"}
BULAN	Mencocokkan angka bulan dari 1 hingga 12, dengan nol opsional untuk bulan satu digit.	20	Masukan: 5 Pola: %{MONTHNUM:month} Keluaran: {"month": "5"} Masukan: 05 Pola: %{MONTHNUM:M:month} Keluaran: {"month": "05"}
MONTHNUM	Cocokkan angka bulan dua digit dari 01 hingga 12.	20	Masukan: 05 Pola: %{MONTHNUM:M2:month} Keluaran: {"month": "05"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
BULANHARI	Pertandingan hari dalam sebulan dari 1 hingga 31, dengan nol opsional.	20	Masukan: 31 Pola: <code>%{MONTHDAY:monthDay}</code> Keluaran: <code>{"monthDay": "31"}</code>
YEAR	Pertandingan tahun dalam dua atau empat digit	20	Masukan: 2024 Pola: <code>%{YEAR:year}</code> Keluaran: <code>{"year": "2024"}</code> Masukan: 24 Pola: <code>%{YEAR:year}</code> Keluaran: <code>{"year": "24"}</code>
DAY	Cocokkan nama hari penuh atau disingkat.	20	Masukan: Tuesday Pola: <code>%{DAY:day}</code> Keluaran: <code>{"day": "Tuesday"}</code>
HOURL	Mencocokkan jam dalam format 24 jam dengan opsional nol (0) 0-23.	20	Masukan: 22 Pola: <code>%{HOUR:hour}</code> Keluaran: <code>{"hour": "22"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
MINUTE	Menit pertandingan (00-59).	20	Masukan: 59 Pola: <code>%{MINUTE:min}</code> Keluaran: <code>{"min": "59"}</code>
DETIK	Cocokkan angka yang mewakili detik (0) 0-60, secara opsional diikuti oleh titik desimal atau titik dua dan satu atau lebih digit untuk menit pecahan	20	Masukan: 3 Pola: <code>%{SECOND:second}</code> Keluaran: <code>{"second": "3"}</code> Masukan: 30.5 Pola: <code>%{SECOND:minSec}</code> Keluaran: <code>{"minSec": "30.5"}</code> Masukan: 30:5 Pola: <code>%{SECOND:minSec}</code> Keluaran: <code>{"minSec": "30:5"}</code>
TIME	Cocokkan format waktu dengan jam, menit, dan detik dalam format (H) H: mm: (s) s. Detik termasuk detik kabisat (0) 0-60.	20	Masukan: 09:45:32 Pola: <code>%{TIME:time}</code> Keluaran: <code>{"time": "09:45:32"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
DATE_US	Cocokkan tanggal dalam format (M) M/ (d) d/ (yy) yy atau (M) M- (d) d- (yy) yy.	20	Masukan: 11/23/2024 Pola: %{DATE_US :date} Keluaran: {"date": "11/23/2024"} Masukan: 1-01-24 Pola: %{DATE_US :date} Keluaran: {"date": "1-01-24"}
DATE_EU	Tanggal pertandingan dalam format (d) d/ (M) M/ (yy) yy, (d) d- (M) M- (yy) yy, atau (d) d. (M) M. (yy) yy.	20	Masukan: 23/11/2024 Pola: %{DATE_EU :date} Keluaran: {"date": "23/11/2024"} Masukan: 1.01.24 Pola: %{DATE_EU :date} Keluaran: {"date": "1.01.24"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
ISO8601_Z ONA WAKTU	Cocokkan offset UTC 'Z' atau offset zona waktu dengan titik dua opsional dalam format [+ -] (H) H (:) mm.	20	Masukan: +05:30 Pola: %{ISO8601_TIMEZONE:tz} Keluaran: {"tz":"+05:30"} Masukan: -530 Pola: %{ISO8601_TIMEZONE:tz} Keluaran: {"tz":"-530"} Masukan: Z Pola: %{ISO8601_TIMEZONE:tz} Keluaran: {"tz":"Z"}
ISO8601_D ETIK	Cocokkan angka yang mewakili detik (0) 0-60, secara opsional diikuti oleh titik desimal atau titik dua dan satu atau lebih digit untuk detik pecahan	20	Masukan: 60 Pola: %{ISO8601_SECOND:second} Keluaran: {"second":"60"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
STEMPEL WAKTU_ ISO8601	Cocokkan format ISO8601 datetime (yy) yy- (M) M- (d) dT (H) H: mm: (s) s) (Z [+ -] (H) H: mm) dengan detik opsional dan zona waktu.	20	Masukan: 2023-05-1 5T14:30:00+05:30 Pola: %{TIMESTA MP_ISO860 1:timestamp} Keluaran: {"timesta mp": "2023 -05-15T14 :30:00+05:30"} Masukan: 23-5-1T1: 25+5:30 Pola: %{TIMESTA MP_ISO860 1:timestamp} Keluaran: {"timesta mp": "23-5 -1T1:25+5:30"} Masukan: 23-5-1T1: 25Z Pola: %{TIMESTA MP_ISO860 1:timestamp} Keluaran: {"timesta mp": "23-5 -1T1:25Z"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
DATE	Cocokkan tanggal dalam format AS menggunakan <code>% {DATE_US}</code> atau dalam format UE menggunakan <code>% {DATE_EU}</code>	20	Masukan: 11/29/2024 Pola: <code>%{DATE:date}</code> Keluaran: <code>{"date": "11/29/2024"}</code> Masukan: 29.11.2024 Pola: <code>%{DATE:date}</code> Keluaran: <code>{"date": "29.11.2024"}</code>
DATESTAMP	Cocokkan <code>% {DATE}</code> diikuti oleh pola <code>% {TIME}</code> , dipisahkan oleh spasi atau tanda hubung.	20	Masukan: 29-11-2024 14:30:00 Pola: <code>%{DATESTAMP:dateTime}</code> Keluaran: <code>{"dateTime": "29-11-2024 14:30:00"}</code>
TZ	Cocokkan singkatan zona waktu umum (PST, PDT, MST, MDT, CST CDT, EST, EDT, UTC).	20	Masukan: PDT Pola: <code>%{TZ:tz}</code> Keluaran: <code>{"tz": "PDT"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
DATESTAMP_RFC822	Tanggal dan waktu pertandingan dalam format: Hari MonthName (D) D (YY) YY (H) H:mm : (s) s Zona Waktu	20	Masukan: Monday Jan 5 23 1:30:00 CDT Pola: %{DATESTAMP_RFC822:dateTime} Keluaran: {"dateTime":"Monday Jan 5 23 1:30:00 CDT"} Masukan: Mon January 15 2023 14:30:00 PST Pola: %{DATESTAMP_RFC822:dateTime} Keluaran: {"dateTime":"Mon January 15 2023 14:30:00 PST"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
DATESTAMP_RFC2822	Format RFC2822 tanggal-waktu yang cocok: Hari, (d) d (yy) yy MonthName (H) H: mm : (s) s Z [+ -] (H) H: mm	20	Masukan: Mon, 15 May 2023 14:30:00 +0530 Pola: %{DATESTAMP_RFC2822:dateTime} Keluaran: {"dateTime": "Mon, 15 May 2023 14:30:00 +0530"} Masukan: Monday, 15 Jan 23 14:30:00 Z Pola: %{DATESTAMP_RFC2822:dateTime} Keluaran: {"dateTime": "Monday, 15 Jan 23 14:30:00 Z"}
DATESTAMP_OTHER	Tanggal dan waktu pertandingan dalam format: Hari MonthName (d) d (H) H:mm : (s) s Zona Waktu (yy) yy	20	Masukan: Mon May 15 14:30:00 PST 2023 Pola: %{DATESTAMP_OTHER:dateTime} Keluaran: {"dateTime": "Mon May 15 14:30:00 PST 2023"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
DATESTAMP_EVENTLOG	Cocokkan format datetime ringkas tanpa pemisah: (yy) YYmm (d) d (H) Hmm (s) s	20	Masukan: 20230515143000 Pola: %{DATESTAMP_EVENTLOG:dateTime} Keluaran: {"dateTime": "20230515143000"}

Pola log grok

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
LOGLEVEL	Cocokkan tingkat log standar dalam kapitalisasi dan singkatan yang berbeda, termasuk yang berikut ini: Alert/ALERT, Trace/TRACE, Debug/DEBUG, Notice/NOTICE, Info/INFO, Warn/Warning/WARN/WARNING, Err/Error/ERR/ERROR, Crit/Critical/CRITICAL, Fatal/FATAL, Severe/SEVERE, Emerg/Emergency/EMERG/EMERGENCY	20	Masukan: INFO Pola: %{LOGLEVEL:logLevel} Keluaran: {"logLevel": "INFO"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
HTTPDATE	Cocokkan format tanggal dan waktu yang sering digunakan dalam file log. Format: (d) d/MonthName/(yy) yy :(H) H: mm : (s) s Zona Waktu: Cocokkan nama bulan bahasa Inggris lengkap atau disingkat (Contoh MonthName: "Jan" atau "Januari") Zona Waktu: Cocokkan% {INT} pola grok	20	Masukan: 23/Nov/20 24:14:30:00 +0640 Pola: %{HTTPDATE:date} Keluaran: {"date": 23/Nov/20 24:14:30:00 +0640"}
SYSLOGTIMESTAMP	Mencocokkan format tanggal dengan MonthName (d) d (H) H: mm :(s) s MonthName : Cocokkan nama bulan Inggris penuh atau disingkat (Contoh: "Jan" atau "Januari")	20	Masukan: Nov 29 14:30:00 Pola: %{SYSLOGTIMESTAMP:dateTime} Keluaran: {"dateTime": "Nov 29 14:30:00"}
PROG	Cocokkan nama program yang terdiri dari string huruf, digit, titik, garis bawah, garis miring ke depan, tanda persen, dan karakter tanda hubung.	20	Masukan: user.profile/settings-page Pola: %{PROGRAM:program} Keluaran: {"program": "user.profile/ settings-page"}

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
SYSLOGPROGRAM	Cocokkan pola grok PROGRAM secara opsional diikuti oleh ID proses dalam tanda kurung siku.	20	Masukan: <code>user.profile/settings-page[1234]</code> Pola: <code>%{SYSLOGPROGRAM:programWithId}</code> Keluaran: <code>{"programWithId": "user.profile/settings-page[1234]", "program": "user.profile/settings-page", "pid": "1234"}</code>
SYSLOGHOST	Cocokkan dengan pola <code>%{HOST}</code> atau <code>%{IP}</code>	5	Masukan: <code>2001:db8:3333:4444:5555:6666:7777:8888</code> Pola: <code>%{SYSLOGHOST:ip}</code> Keluaran: <code>{"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}</code>

Pola Grok	Deskripsi	Batas pola maksimum	Contoh
FASILITAS SYSLOG	Cocokkan prioritas syslog dalam format desimal. Nilai harus diapit dalam tanda kurung sudut (<>).	20	Masukan: <13.6> Pola: %{SYSLOGFACILITY:syslog} Keluaran: {"syslog": "<13.6>", "facility": "13", "priority": "6"}

Pola grok log umum

Anda dapat menggunakan pola grok kustom yang telah ditentukan sebelumnya untuk mencocokkan format log Apache, NGINX dan Syslog Protocol (RFC 5424). Saat Anda menggunakan pola khusus ini, mereka harus menjadi pola pertama dalam konfigurasi pencocokan Anda, dan tidak ada pola lain yang dapat mendahuluinya. Juga, Anda dapat mengikuti mereka hanya dengan tepat satu DATA.GREEDYDATA atau pola GREEDYDATA_MULTILINE.

Pola Grok	Deskripsi	Batas pola maksimum
APACHE_ACCESS_LOG	Cocokkan log akses Apache	1
NGINX_ACCESS_LOG	Cocokkan log akses NGINX	1
SYSLOG5424	Pertandingan Syslog Protocol (RFC 5424) log	1

Berikut ini menunjukkan contoh yang valid dan tidak valid untuk menggunakan pola format log umum ini.


```
"%{NGINX_ACCESS_LOG} %{DATA}" // Valid
"%{SYSLOG5424}%{DATA:logMsg}" // Valid
"%{APACHE_ACCESS_LOG} %{GREEDYDATA:logMsg}" // Valid
"%{APACHE_ACCESS_LOG} %{SYSLOG5424}" // Invalid (multiple common log patterns used)
"%{NGINX_ACCESS_LOG} %{NUMBER:num}" // Invalid (Only GREEDYDATA and DATA patterns are supported with common log patterns)
"%{GREEDYDATA:logMsg} %{SYSLOG5424}" // Invalid (GREEDYDATA and DATA patterns are supported only after common log patterns)
```

Contoh format log umum

Contoh log Apache

Contoh log:

```
127.0.0.1 - - [03/Aug/2023:12:34:56 +0000] "GET /page.html HTTP/1.1" 200 1234
```

Transformator:

```
[
  {
    "grok": {
      "match": "%{APACHE_ACCESS_LOG}"
    }
  }
]
```

Output:

```
{
  "request": "/page.html",
  "http_method": "GET",
  "status_code": 200,
  "http_version": "1.1",
  "response_size": 1234,
  "remote_host": "127.0.0.1",
  "timestamp": "2023-08-03T12:34:56Z"
}
```

Contoh log NGINX

Contoh log:

```
192.168.1.100 - Foo [03/Aug/2023:12:34:56 +0000] "GET /account/login.html HTTP/1.1"
200 42 "https://www.amazon.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36"
```

Transformer:

```
[
  {
    "grok": {
      "match": "%{NGINX_ACCESS_LOG}"
    }
  }
]
```

Output:

```
{
  "request": "/account/login.html",
  "referrer": "https://www.amazon.com/",
  "agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/92.0.4515.131 Safari/537.36",
  "http_method": "GET",
  "status_code": 200,
  "auth_user": "Foo",
  "http_version": "1.1",
  "response_size": 42,
  "remote_host": "192.168.1.100",
  "timestamp": "2023-08-03T12:34:56Z"
}
```

Contoh log Protokol Syslog (RFC 5424)

Contoh log:

```
<165>1 2003-10-11T22:14:15.003Z mymachine.example.com evntslog - ID47
[exampleSDID@32473 iut="3" eventSource= "Application" eventID="1011"]
[examplePriority@32473 class="high"]
```

Transformer:

```
[
```

```
{
  "grok": {
    "match": "%{SYSLOG5424}"
  }
}
```

Output:

```
{
  "pri": 165,
  "version": 1,
  "timestamp": "2003-10-11T22:14:15.003Z",
  "hostname": "mymachine.example.com",
  "app": "evntslog",
  "msg_id": "ID47",
  "structured_data": "exampleSDID@32473 iut=\"3\" eventSource= \"Application\" eventID= \"1011\"",
  "message": "[examplePriority@32473 class=\"high\"]"
```

CSV

Prosesor csv mem-parsing nilai yang dipisahkan koma (CSV) dari peristiwa log ke dalam kolom.

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Jalur ke bidang dalam peristiwa log yang akan diuraikan	Tidak	@message	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
pembatas	Karakter yang digunakan untuk memisahkan setiap kolom dalam peristiwa log nilai dipisahkan koma asli	Tidak	,	Panjang maksimum: 1 kecuali nilainya adalah \t atau \s

Bidang	Deskripsi	Diperlukan?	Default	Batas
Quotecharacter	Karakter yang digunakan sebagai kualifikasi teks untuk satu kolom data	Tidak	"	Panjang maksimal: 1
kolom	Daftar nama yang akan digunakan untuk kolom dalam peristiwa log yang diubah.	Tidak	[column_1, column_2]	Kolom CSV maksimum: 100 Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
tujuan	Bidang induk untuk menempatkan pasangan nilai kunci yang diubah di bawah	Tidak	Root node	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3

Pengaturan `delimiter` untuk `\t` akan memisahkan setiap kolom pada karakter tab, dan `\t` akan memisahkan setiap kolom pada karakter spasi tunggal.

Contoh

Misalkan bagian dari peristiwa log yang dicerna terlihat seperti ini:

```
'Akua Mansa':28:'New York: USA'
```

Misalkan kita hanya menggunakan prosesor csv:

```
[
  {
    "csv": {
      "delimiter": ":",
```

```

        "quoteCharacter": ""
    }
}
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "column_1": "Akua Mansa",
  "column_2": "28",
  "column_3": "New York: USA"
}
```

Contoh 2

Misalkan peristiwa log tertelan terlihat seperti ini:

```

{
  "timestamp": "2024-11-23T16:03:12Z",
  "type": "user_data",
  "logMsg": "'Akua Mansa':28:'New York: USA'"
}
```

Misalkan kita mengurai acara sebagai JSON, mereka mengurai bidang JSON dengan prosesor csv, menentukan nama kolom dan tujuan:

```

[
  {
    "parseJSON": {}
  },
  {
    "csv": {
      "source": "logMsg",
      "delimiter": ":",
      "quoteCharacter": "",
      "columns": ["name", "age", "location"],
      "destination": "msg"
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "timestamp": "2024-11-23T16:03:12Z",
  "logMsg": "'Akua Mansa':28:'New York: USA'",
  "type": "user_data",
  "msg": {
    "name": "Akua Mansa",
    "age": "28",
    "location": "New York: USA"
  }
}
```

parseKeyValue

Gunakan `parseKeyValue` prosesor untuk mengurai bidang tertentu menjadi pasangan kunci-nilai. Anda dapat menyesuaikan prosesor untuk mengurai informasi bidang dengan opsi berikut.

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Jalur ke bidang dalam peristiwa log yang akan diuraikan	Tidak	@message	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
tujuan	Bidang tujuan untuk menempatkan pasangan kunci-nilai yang diekstraksi ke dalam	Tidak		Panjang maksimum: 128
Fielddelimiter	String pembatas bidang yang digunakan antara pasangan kunci-nilai dalam peristiwa log asli	Tidak	&	Panjang maksimum: 128
keyValueDelimiter	String pembatas untuk digunakan antara kunci dan nilai di setiap pasangan dalam peristiwa log yang diubah	Tidak	=	Panjang maksimum: 128

Bidang	Deskripsi	Diperlukan?	Default	Batas
nonMatchValue	Nilai untuk dimasukkan ke dalam bidang nilai dalam hasil, ketika pasangan kunci-nilai tidak berhasil dibagi.	Tidak		Panjang maksimum: 128
KeyPrefix	Jika Anda ingin menambahkan awalan ke semua kunci yang diubah, tentukan di sini.	Tidak		Panjang maksimum: 128
overwriteIfExists	Apakah akan menimpa nilai jika kunci tujuan sudah ada	Tidak	false	

Contoh

Ambil contoh peristiwa log berikut:

```
key1:value1!key2:value2!key3:value3!key4
```

Misalkan kita menggunakan konfigurasi prosesor berikut:

```
[
  {
    "parseKeyValue": {
      "destination": "new_key",
      "fieldDelimiter": "!",
      "keyValueDelimiter": ":",
      "nonMatchValue": "defaultValue",
      "keyPrefix": "parsed_"
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "new_key": {
```

```
"parsed_key1": "value1",  
"parsed_key2": "value2",  
"parsed_key3": "value3",  
"parsed_key4": "defaultValue"  
}  
}
```

Prosesor bawaan untuk AWS log penjual otomatis

Bagian ini berisi informasi tentang prosesor bawaan yang dapat Anda gunakan dengan AWS layanan yang menjual log.

Daftar Isi

- [ParseWAF](#)
- [ParsePostgres](#)
- [ParseCloudFront](#)
- [ParseRoute53](#)
- [ParseVPC](#)
- [ParseTooCSF](#)

ParseWAF

Gunakan prosesor ini untuk mengurai log AWS WAF vended, Dibutuhkan isi `httpRequest.headers` dan membuat kunci JSON dari setiap nama header, dengan nilai yang sesuai. Itu juga melakukan hal yang sama untuk `labels`. Transformasi ini dapat membuat query AWS WAF log jauh lebih mudah. Untuk informasi selengkapnya tentang format AWS WAF log, lihat [Contoh log untuk lalu lintas ACL web](#).

Prosesor ini hanya menerima `@message` sebagai input.

Important

Jika Anda menggunakan prosesor ini, itu harus menjadi prosesor pertama di transformator Anda.

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "timestamp": 1576280412771,
  "formatVersion": 1,
  "webaclId": "arn:aws:wafv2:ap-southeast-2:111122223333:regional/webacl/
STMTest/1EXAMPLE-2ARN-3ARN-4ARN-123456EXAMPLE",
  "terminatingRuleId": "STMTest_SQLi_XSS",
  "terminatingRuleType": "REGULAR",
  "action": "BLOCK",
  "terminatingRuleMatchDetails": [
    {
      "conditionType": "SQL_INJECTION",
      "sensitivityLevel": "HIGH",
      "location": "HEADER",
      "matchedData": ["10", "AND", "1"]
    }
  ],
  "httpSourceName": "-",
  "httpSourceId": "-",
  "ruleGroupList": [],
  "rateBasedRuleList": [],
  "nonTerminatingMatchingRules": [],
  "httpRequest": {
    "clientIp": "1.1.1.1",
    "country": "AU",
    "headers": [
      { "name": "Host", "value": "localhost:1989" },
      { "name": "User-Agent", "value": "curl/7.61.1" },
      { "name": "Accept", "value": "*/*" },
      { "name": "x-stm-test", "value": "10 AND 1=1" }
    ],
    "uri": "/myUri",
    "args": "",
    "httpVersion": "HTTP/1.1",
    "httpMethod": "GET",
    "requestId": "rid"
  },
  "labels": [{ "name": "value" }]
}
```

Konfigurasi prosesor adalah ini:

```
[
  {
    "parseWAF": {}
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "httpRequest": {
    "headers": {
      "Host": "localhost:1989",
      "User-Agent": "curl/7.61.1",
      "Accept": "/*/*",
      "x-stm-test": "10 AND 1=1"
    },
    "clientIp": "1.1.1.1",
    "country": "AU",
    "uri": "/myUri",
    "args": "",
    "httpVersion": "HTTP/1.1",
    "httpMethod": "GET",
    "requestId": "rid"
  },
  "labels": { "name": "value" },
  "timestamp": 1576280412771,
  "formatVersion": 1,
  "webaclId": "arn:aws:wafv2:ap-southeast-2:111122223333:regional/webacl/
STMTest/1EXAMPLE-2ARN-3ARN-4ARN-123456EXAMPLE",
  "terminatingRuleId": "STMTest_SQLi_XSS",
  "terminatingRuleType": "REGULAR",
  "action": "BLOCK",
  "terminatingRuleMatchDetails": [
    {
      "conditionType": "SQL_INJECTION",
      "sensitivityLevel": "HIGH",
      "location": "HEADER",
      "matchedData": ["10", "AND", "1"]
    }
  ],
  "httpSourceName": "-",
  "httpSourceId": "-",
  "ruleGroupList": [],
}
```

```
"rateBasedRuleList": [],  
"nonTerminatingMatchingRules": []  
}
```

ParsePostgres

Gunakan prosesor ini untuk mengurai log Amazon RDS for PostgreSQL penjual, mengekstrak bidang, dan mengonversinya ke format JSON. Untuk informasi selengkapnya tentang RDS untuk format log PostgreSQL, lihat [RDS](#) untuk file log database PostgreSQL.

Prosesor ini hanya menerima @message sebagai input.

Important

Jika Anda menggunakan prosesor ini, itu harus menjadi prosesor pertama di transformator Anda.

Contoh

Ambil contoh peristiwa log berikut:

```
2019-03-10 03:54:59 UTC:10.0.0.123(52834):postgres@logtestdb:[20175]:ERROR: column  
"wrong_column_name" does not exist at character 8
```

Konfigurasi prosesor adalah ini:

```
[  
  {  
    "parsePostgres": {}  
  }  
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{  
  "logTime": "2019-03-10 03:54:59 UTC",  
  "srcIp": "10.0.0.123(52834)",  
  "userName": "postgres",  
  "dbName": "logtestdb",  
  "processId": "20175",  
}
```

```
"logLevel": "ERROR"
}
```

ParseCloudFront

Gunakan prosesor ini untuk mengurai log Amazon CloudFront vended, mengekstrak bidang, dan mengubahnya menjadi format JSON. Nilai bidang yang dikodekan diterjemahkan. Nilai yang merupakan bilangan bulat dan ganda diperlakukan seperti itu. Untuk informasi selengkapnya tentang format Amazon CloudFront log, lihat [Mengkonfigurasi dan menggunakan log standar \(log akses\)](#).

Prosesor ini hanya menerima @message sebagai input.

Important

Jika Anda menggunakan prosesor ini, itu harus menjadi prosesor pertama di transformator Anda.

Contoh

Ambil contoh peristiwa log berikut:

```
2019-12-04 21:02:31 LAX1 392 192.0.2.24 GET
d111111abcdef8.cloudfront.net /index.html 200 - Mozilla/5.0%20(Windows
%20NT%2010.0;%20Win64;%20x64)%20AppleWebKit/537.36%20(KHTML,
%20like%20Gecko)%20Chrome/78.0.3904.108%20Safari/537.36 - - Hit
SOX4xwn4XV6Q4rgb7XiVG0Hms_BG1TAC4KyHmureZmBNrjGdRLiNIQ==
d111111abcdef8.cloudfront.net https 23 0.001 - TLSv1.2 ECDHE-RSA-AES128-GCM-
SHA256 Hit HTTP/2.0 - - 11040 0.001 Hit text/html 78 - -
```

Konfigurasi prosesor adalah ini:

```
[
  {
    "parseCloudfront": {}
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
```

```

"date": "2019-12-04",
"time": "21:02:31",
"x-edge-location": "LAX1",
"sc-bytes": 392,
"c-ip": "192.0.2.24",
"cs-method": "GET",
"cs(Host)": "d111111abcdef8.cloudfront.net",
"cs-uri-stem": "/index.html",
"sc-status": 200,
"cs(Referer)": "-",
"cs(User-Agent)": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36",
"cs-uri-query": "-",
"cs(Cookie)": "-",
"x-edge-result-type": "Hit",
"x-edge-request-id": "S0X4xwn4XV6Q4rgb7XiVG0Hms_BG1TAC4KyHmureZmBNrjGdRLiNIQ==",
"x-host-header": "d111111abcdef8.cloudfront.net",
"cs-protocol": "https",
"cs-bytes": 23,
"time-taken": 0.001,
"x-forwarded-for": "-",
"ssl-protocol": "TLSv1.2",
"ssl-cipher": "ECDHE-RSA-AES128-GCM-SHA256",
"x-edge-response-result-type": "Hit",
"cs-protocol-version": "HTTP/2.0",
"fle-status": "-",
"fle-encrypted-fields": "-",
"c-port": 11040,
"time-to-first-byte": 0.001,
"x-edge-detailed-result-type": "Hit",
"sc-content-type": "text/html",
"sc-content-len": 78,
"sc-range-start": "-",
"sc-range-end": "-"
}

```

ParseRoute53

Gunakan prosesor ini untuk mengurai log Amazon Route 53 Public Data Plane vendid, mengekstrak bidang, dan mengubahnya menjadi format JSON. Nilai bidang yang dikodekan diterjemahkan. Prosesor ini tidak mendukung Amazon Route 53 Resolver log.

Prosesor ini hanya menerima @message sebagai input.

⚠ Important

Jika Anda menggunakan prosesor ini, itu harus menjadi prosesor pertama di transformator Anda.

Contoh

Ambil contoh peristiwa log berikut:

```
1.0 2017-12-13T08:15:50.235Z Z123412341234 example.com AAAA NOERROR TCP IAD12 192.0.2.0
198.51.100.0/24
```

Konfigurasi prosesor adalah ini:

```
[
  {
    "parseRoute53": {}
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "version": 1.0,
  "queryTimestamp": "2017-12-13T08:15:50.235Z",
  "hostZoneId": "Z123412341234",
  "queryName": "example.com",
  "queryType": "AAAA",
  "responseCode": "NOERROR",
  "protocol": "TCP",
  "edgeLocation": "IAD12",
  "resolverIp": "192.0.2.0",
  "ednsClientSubnet": "198.51.100.0/24"
}
```

ParseVPC

Gunakan prosesor ini untuk mengurai log penjual Amazon VPC, mengekstrak bidang, dan mengubahnya menjadi format JSON. Nilai bidang yang dikodekan diterjemahkan.

Prosesor ini hanya menerima @message sebagai input.

 Important

Jika Anda menggunakan prosesor ini, itu harus menjadi prosesor pertama di transformator Anda.

Contoh

Ambil contoh peristiwa log berikut:

```
2 123456789010 eni-abc123de 192.0.2.0 192.0.2.24 20641 22 6 20 4249 1418530010
1418530070 ACCEPT OK
```

Konfigurasi prosesor adalah ini:

```
[
  {
    "parseVPC": {}
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "version": 2,
  "accountId": "123456789010",
  "interfaceId": "eni-abc123de",
  "srcAddr": "192.0.2.0",
  "dstAddr": "192.0.2.24",
  "srcPort": 20641,
  "dstPort": 22,
  "protocol": 6,
  "packets": 20,
  "bytes": 4249,
  "start": 1418530010,
  "end": 1418530070,
  "action": "ACCEPT",
  "logStatus": "OK"
}
```

ParseTooCSF

parseToOCSFProsesor mengubah log menjadi peristiwa Open Cybersecurity Schema Framework (OCSF). OCSF adalah standar terbuka yang menyediakan skema umum untuk data keamanan, memungkinkan interoperabilitas dan analisis yang lebih baik di berbagai alat dan platform keamanan.

Prosesor ini sangat berguna untuk alur kerja analitik keamanan di mana Anda perlu menstandarisasi format log dari berbagai AWS layanan ke dalam skema yang konsisten untuk analisis hilir.

Parameter

eventSource(Diperlukan)

Menentukan AWS layanan atau proses yang menghasilkan peristiwa log yang akan dikonversi. Nilai yang valid adalah:

- CloudTrail- CloudTrail log
- Route53Resolver- Log Route 53 Resolver
- VPCFlow- Log Aliran VPC Amazon
- EKSAudit- Log audit Amazon EKS
- AWSWAF- AWS WAF log

ocsfVersion(Diperlukan)

Menentukan versi skema OCSF yang akan digunakan untuk peristiwa log diubah. Versi yang saat ini didukung: V1.1, V1.5

mappingVersion (opsional)

Menentukan versi pemetaan transformasi OCSF. Mengontrol logika transformasi mana yang diterapkan saat mengonversi log ke format OCSF. Jika tidak ditentukan, gunakan versi terbaru yang tersedia pada waktu pembuatan kebijakan. Kebijakan yang ada tidak otomatis ditingkatkan saat versi pemetaan baru dirilis. Versi terbaru saat ini: v1.5.0.

Catatan: Tidak ocsfVersion didukung kapan V1.1.

source (opsional)

Jalur ke bidang dalam peristiwa log yang ingin Anda uraikan. Jika dihilangkan, seluruh pesan log diuraikan.

Contoh

Contoh berikut menunjukkan bagaimana menggunakan `parseToOCSF` untuk mengkonversi VPC Flow Logs ke format OCSF:

```
{
  "parseToOCSF": {
    "eventSource": "VPCFlow",
    "ocsfVersion": "V1.1"
  }
}
```

Contoh berikut menunjukkan cara menentukan versi pemetaan tertentu untuk perilaku transformasi yang konsisten:

```
{
  "parseToOCSF": {
    "eventSource": "CloudTrail",
    "ocsfVersion": "V1.5",
    "mappingVersion": "v1.5.0"
  }
}
```

Prosesor mutasi string

Bagian ini berisi informasi tentang prosesor mutasi string yang dapat Anda gunakan dengan transformator peristiwa log.

Daftar Isi

- [lowerCaseString](#)
- [upperCaseString](#)
- [SplitString](#)
- [SubstituteString](#)
- [TrimString](#)

lowerCaseString

`lowerCaseString` prosesor mengkonversi string ke versi huruf kecil.

Bidang	Deskripsi	Diperlukan?	Default	Batas
denganKeys	Daftar kunci untuk dikonversi ke huruf kecil	Ya		Entri maksimum: 10

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "INNER_VALUE"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `toLowerCaseString` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "toLowerCaseString": {
      "withKeys": ["outer_key.inner_key"]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

toUpperCaseString

`toUpperCaseStringProsesor` mengkonversi string ke versi huruf besar.

Bidang	Deskripsi	Diperlukan?	Default	Batas
denganKeys	Daftar kunci untuk dikonversi ke huruf besar	Ya		Entri maksimum: 10

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `upperCaseString` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "upperCaseString": {
      "withKeys": ["outer_key.inner_key"]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": {
    "inner_key": "INNER_VALUE"
  }
}
```

SplitString

`splitString` prosesor adalah jenis prosesor mutasi string yang membagi bidang menjadi array menggunakan karakter pembatas.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array harus berisi <code>source</code> dan <code>delimiter</code> bidang.	Ya		Entri maksimum: 10
sumber	Kunci dari nilai bidang untuk dibagi	Ya		Panjang maksimum: 128
pembatas	String pembatas untuk membagi nilai bidang pada	Ya		Panjang maksimum: 128

Contoh 1

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `splitString` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "splitString": {
      "entries": [
        {
          "source": "outer_key.inner_key",
          "delimiter": "_"
        }
      ]
    }
  }
]
```

```

    }
  ]
}
]

```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": {
    "inner_key": [
      "inner",
      "value"
    ]
  }
}

```

Contoh 2

Pembatas untuk membagi string bisa menjadi beberapa karakter panjang.

Ambil contoh peristiwa log berikut:

```

{
  "outer_key": {
    "inner_key": "item1, item2, item3"
  }
}

```

Konfigurasi transformator adalah sebagai berikut:

```

[
  {
    "parseJSON": {}
  },
  {
    "splitString": {
      "entries": [
        {
          "source": "outer_key.inner_key",
          "delimiter": ", "
        }
      ]
    }
  }
]

```

```

    ]
  }
}
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": {
    "inner_key": [
      "item1",
      "item2",
      "item3"
    ]
  }
}
```

SubstituteString

`substituteStringProsesor` adalah jenis prosesor mutasi string yang cocok dengan nilai kunci terhadap ekspresi reguler dan menggantikan semua kecocokan dengan string pengganti.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array harus beris <code>source</code> , <code>from</code> , dan <code>to</code> bidang.	Ya		Entri maksimum: 10
sumber	Kunci bidang untuk memodifikasi	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
From	String ekspresi reguler yang akan diganti. Karakter regex khusus seperti <code>[dan]</code> harus diloloskan menggunakan <code>\\</code> saat menggunak	Ya		Panjang maksimum: 128

Bidang	Deskripsi	Diperlukan?	Default	Batas
	an tanda kutip ganda dan dengan \ saat menggunakan tanda kutip tunggal atau ketika dikonfigurasi dari. Konsol Manajemen AWS Untuk informasi selengkapnya, lihat Pola Kelas di situs web Oracle. Anda dapat membungkus pola (. . .) untuk membuat grup penangkap bernomor dan membuat grup penangkap (?P<group_name> . . .) bernama yang dapat direferensikan di bidang. to			
kepada	String yang akan diganti untuk setiap kecocokan from Backreferences untuk menangkap grup dapat digunakan. Gunakan formulir \$n untuk grup bernomor seperti \$1 dan gunakan \${group_name} untuk grup bernama seperti \$. {my_group} >	Ya		Panjang maksimum: 128 Jumlah backreferensi-referensi maksimum: 10 Jumlah maksimum referensi balik duplikat: 2

Contoh 1

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key1": "[]",
    "inner_key2": "123-345-567",
    "inner_key3": "A cat takes a catnap."
```

```

    }
  }
}

```

Konfigurasi transformator adalah ini, menggunakan `substituteString` dengan `parseJSON`:

```

[
  {
    "parseJSON": {}
  },
  {
    "substituteString": {
      "entries": [
        {
          "source": "outer_key.inner_key1",
          "from": "\\[\\]",
          "to": "value1"
        },
        {
          "source": "outer_key.inner_key2",
          "from": "[0-9]{3}-[0-9]{3}-[0-9]{3}",
          "to": "xxx-xxx-xxx"
        },
        {
          "source": "outer_key.inner_key3",
          "from": "cat",
          "to": "dog"
        }
      ]
    }
  }
]

```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": {
    "inner_key1": "value1",
    "inner_key2": "xxx-xxx-xxx",
    "inner_key3": "A dog takes a dognap."
  }
}

```

Contoh 2

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key1": "Tom, Dick, and Harry",
    "inner_key2": "arn:aws:sts::123456789012:assumed-role/MyImportantRole/MySession"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `substituteString` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "substituteString": {
      "entries": [
        {
          "source": "outer_key.inner_key1",
          "from": "(\\w+), (\\w+), and (\\w+)",
          "to": "$1 and $3"
        },
        {
          "source": "outer_key.inner_key2",
          "from": "^arn:aws:sts::(?P<account_id>\\d{12}):assumed-role/(?P<role_name>[\\w+=,.-]+)/(?P<role_session_name>[\\w+=,.-]+)$",
          "to": "${account_id}:${role_name}:${role_session_name}"
        }
      ]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": {
    "inner_key1": "Tom and Harry",
    "inner_key2": "123456789012:MyImportantRole:MySession"
  }
}
```

```
}

```

TrimString

`trimString`Prosesor menghapus spasi putih dari awal dan akhir kunci.

Bidang	Deskripsi	Diperlukan?	Default	Batas
denganKey	Daftar kunci untuk memangkas	Ya		Entri maksimum: 10
s				

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "  inner_value  "
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `trimString` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "trimString": {
      "withKeys": ["outer_key.inner_key"]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

```
}
}
```

JSON bermutasi prosesor

Bagian ini berisi informasi tentang prosesor mutasi JSON yang dapat Anda gunakan dengan transformator peristiwa log.

Daftar Isi

- [AddKeys](#)
- [DeleteKeys](#)
- [MoveKeys](#)
- [RenameKeys](#)
- [CopyValue](#)
- [listToMap](#)

AddKeys

Gunakan addKeys prosesor untuk menambahkan pasangan kunci-nilai baru ke peristiwa log.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array dapat berisi <code>key</code> , <code>value</code> , dan <code>overwriteIfExists</code> bidang.	Ya		Entri maksimal: 5
kunci	Kunci entri baru yang akan ditambahkan	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
nilai	Nilai entri baru yang akan ditambahkan	Ya		Panjang maksimal: 256

Bidang	Deskripsi	Diperlukan?	Default	Batas
overwriteIfExists	Jika Anda menyetel ini ke true, nilai yang ada akan ditimpa jika key sudah ada dalam acara tersebut. Nilai default-nya adalah false.	Tidak	false	Tidak ada batas

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan addKeys dengan parseJSON:

```
[
  {
    "parseJSON": {}
  },
  {
    "addKeys": {
      "entries": [
        {
          "key": "outer_key.new_key",
          "value": "new_value"
        }
      ]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
```

```
"outer_key": {
  "inner_key": "inner_value",
  "new_key": "new_value"
}
}
```

DeleteKeys

Gunakan `deleteKeys` prosesor untuk menghapus bidang dari peristiwa log. Bidang ini dapat mencakup pasangan kunci-nilai.

Bidang	Deskripsi	Diperlukan?	Default	Batas
denganKeys	Daftar kunci untuk dihapus.	Ya	Tidak ada batas	Entri maksimal: 5

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan `deleteKeys` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "deleteKeys": {
      "withKeys": ["outer_key.inner_key"]
    }
  }
]
```

]

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": {}
}
```

MoveKeys

Gunakan moveKeys prosesor untuk memindahkan kunci dari satu bidang ke bidang lainnya.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array dapat berisiproperty, target, dan overwriteIfExists bidang.	Ya		Entri maksimal: 5
sumber	Kunci untuk bergerak	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
target	Kunci untuk pindah ke	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
overwriteIfExists	Jika Anda menyetel ini ke true, nilai yang ada akan ditimpa jika key sudah ada dalam acara tersebut. Nilai default-nya adalah false.	Tidak	false	Tidak ada batas

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key1": {
    "inner_key1": "inner_value1"
  },
  "outer_key2": {
    "inner_key2": "inner_value2"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan moveKeys dengan parseJSON:

```
[
  {
    "parseJSON": {}
  },
  {
    "moveKeys": {
      "entries": [
        {
          "source": "outer_key1.inner_key1",
          "target": "outer_key2"
        }
      ]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key1": {},
  "outer_key2": {
    "inner_key2": "inner_value2",
    "inner_key1": "inner_value1"
  }
}
```

RenameKeys

Gunakan renameKeys prosesor untuk mengganti nama kunci dalam peristiwa log.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array dapat berisi key, target, dan overwriteIfExists bidang.	Ya	Tidak ada batas	Entri maksimal: 5
kunci	Kunci untuk mengganti nama	Ya	Tidak ada batas	Panjang maksimum: 128
target	Nama kunci baru	Ya	Tidak ada batas	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
overwriteIfExists	Jika Anda menyetel ini ke true, nilai yang ada akan ditimpa jika key sudah ada dalam acara tersebut. Nilai default-nya adalah false.	Tidak	false	Tidak ada batas

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```


Konfigurasi transformator adalah ini, menggunakan `renameKeys` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "renameKeys": {
      "entries": [
        {
          "key": "outer_key",
          "target": "new_key"
        }
      ]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "new_key": {
    "inner_key": "inner_value"
  }
}
```

CopyValue

Gunakan `copyValue` prosesor untuk menyalin nilai dalam peristiwa log. Anda juga dapat menggunakan prosesor ini untuk menambahkan metadata ke log peristiwa, dengan menyalin nilai kunci metadata berikut ke dalam peristiwa log:,,, @logGroupName @logGroupStream @accountId @regionName Ini diilustrasikan dalam contoh berikut.

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array dapat berisi <code>source</code> , <code>target</code> , dan <code>overwriteIfExists</code> bidang.	Ya		Entri maksimal: 5

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Kunci untuk menyalin	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
target	Kunci untuk menyalin nilai ke	Ya	Tidak ada batas	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
overwriteIfExists	Jika Anda menyetel ini ke true, nilai yang ada akan ditimpa jika key sudah ada dalam acara tersebut. Nilai default-nya adalah false.	Tidak	false	Tidak ada batas

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Konfigurasi transformator adalah ini, menggunakan copyValue dengan parseJSON:

```
[
  {
    "parseJSON": {}
  },
]
```

```

{
  "copyValue": {
    "entries": [
      {
        "key": "outer_key.new_key",
        "target": "new_key"
      },
      {
        "source": "@logGroupName",
        "target": "log_group_name"
      },
      {
        "source": "@logGroupStream",
        "target": "log_group_stream"
      },
      {
        "source": "@accountId",
        "target": "account_id"
      },
      {
        "source": "@regionName",
        "target": "region_name"
      }
    ]
  }
}

```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": {
    "inner_key": "inner_value"
  },
  "new_key": "inner_value",
  "log_group_name": "myLogGroupName",
  "log_group_stream": "myLogStreamName",
  "account_id": "012345678912",
  "region_name": "us-east-1"
}

```

listToMap

`listToMapProsesor` mengambil daftar objek yang berisi bidang kunci, dan mengubahnya menjadi peta kunci target.

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Kunci dalam <code>ProcessingEvent</code> dengan daftar objek yang akan dikonversi ke peta	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
kunci	Kunci bidang yang akan diekstraksi sebagai kunci di peta yang dihasilkan	Ya		Panjang maksimum: 128
ValueKey	Jika ini ditentukan, nilai yang Anda tentukan dalam parameter ini akan diekstraksi dari <code>source</code> objek dan dimasukkan ke dalam nilai peta yang dihasilkan. Jika tidak, objek asli dalam daftar sumber akan dimasukkan ke dalam nilai peta yang dihasilkan.	Tidak		Panjang maksimum: 128
target	Kunci bidang yang akan menampung peta yang dihasilkan	Tidak	Simpul akar	Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
meratakan	Nilai Boolean untuk menunjukkan apakah daftar akan diratakan menjadi item tunggal atau jika nilai	Tidak	false	

Bidang	Deskripsi	Diperlukan?	Default	Batas
	dalam peta yang dihasilkan akan menjadi daftar. Secara default nilai untuk kunci yang cocok akan direpresentasikan dalam array. Set <code>flatten true</code> untuk mengkonversi array ke nilai tunggal berdasarkan nilai <code>flattenedElement</code>.			
Flattened Element	Jika Anda mengatur <code>flatten</code> ke <code>true</code> , gunakan <code>flattenedElement</code> untuk menentukan elemen mana, <code>first</code> atau <code>last</code> , untuk disimpan.	Diperlukan saat <code>flatten</code> diatur ke <code>true</code>		Nilai hanya bisa <code>first</code> atau <code>last</code>

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "outer_key": [
    {
      "inner_key": "a",
      "inner_value": "val-a"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b1"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b2"
    },
    {
      "inner_key": "c",
      "inner_value": "val-c"
    }
  ]
}
```

```
    }
  ]
}
```

Transformator untuk kasus penggunaan 1: `flatten` adalah `false`

```
[
  {
    "parseJSON": {}
  },
  {
    "listToMap": {
      "source": "outer_key"
      "key": "inner_key",
      "valueKey": "inner_value",
      "flatten": false
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "outer_key": [
    {
      "inner_key": "a",
      "inner_value": "val-a"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b1"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b2"
    },
    {
      "inner_key": "c",
      "inner_value": "val-c"
    }
  ],
  "a": [
    "val-a"
```

```

    ],
    "b": [
        "val-b1",
        "val-b2"
    ],
    "c": [
        "val-c"
    ]
  }

```

Transformator untuk kasus penggunaan 2: `flatten` adalah `true` dan `flattenedElement` adalah `first`

```

[
  {
    "parseJSON": {}
  },
  {
    "listToMap": {
      "source": "outer_key"
      "key": "inner_key",
      "valueKey": "inner_value",
      "flatten": true,
      "flattenedElement": "first"
    }
  }
]

```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": [
    {
      "inner_key": "a",
      "inner_value": "val-a"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b1"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b2"
    }
  ]
}

```

```

    },
    {
      "inner_key": "c",
      "inner_value": "val-c"
    }
  ],
  "a": "val-a",
  "b": "val-b1",
  "c": "val-c"
}

```

Transformator untuk kasus penggunaan 3: `flatten` adalah `true` dan `flattenedElement` adalah `last`

```

[
  {
    "parseJSON": {}
  },
  {
    "listToMap": {
      "source": "outer_key"
      "key": "inner_key",
      "valueKey": "inner_value",
      "flatten": true,
      "flattenedElement": "last"
    }
  }
]

```

Peristiwa log yang diubah adalah sebagai berikut.

```

{
  "outer_key": [
    {
      "inner_key": "a",
      "inner_value": "val-a"
    },
    {
      "inner_key": "b",
      "inner_value": "val-b1"
    },
    {
      "inner_key": "b",

```



```

        "inner_value": "val-b2"
    },
    {
        "inner_key": "c",
        "inner_value": "val-c"
    }
],
"a": "val-a",
"b": "val-b2",
"c": "val-c"
}

```

Prosesor konverter tipe data

Bagian ini berisi informasi tentang prosesor konverter tipe data yang dapat Anda gunakan dengan transformator peristiwa log.

Daftar Isi

- [TypeConverter](#)
- [DateTimeConverter](#)

TypeConverter

Gunakan `typeConverter` prosesor untuk mengonversi jenis nilai yang terkait dengan kunci yang ditentukan ke tipe yang ditentukan. Ini adalah prosesor casting yang mengubah jenis bidang yang ditentukan. Nilai dapat diubah menjadi salah satu tipe data berikut: `integer`, `double` dan `string` `boolean`

Bidang	Deskripsi	Diperlukan?	Default	Batas
entri	Array entri. Setiap item dalam array harus berisi key dan type bidang.	Ya		Entri maksimum: 10
kunci	Kunci dengan nilai yang akan dikonversi ke tipe yang berbeda	Ya		Panjang maksimum: 128

Bidang	Deskripsi	Diperlukan?	Default	Batas
				Kedalaman kunci bersarang maksimum: 3
jenis	Jenis untuk dikonversi ke. Nilai yang valid adalah <code>integer</code> , <code>double</code> , <code>string</code> dan <code>boolean</code> .	Ya		

Contoh

Ambil contoh peristiwa log berikut:

```
{
  "name": "value",
  "status": "200"
}
```

Konfigurasi transformator adalah ini, menggunakan `typeConverter` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "typeConverter": {
      "entries": [
        {
          "key": "status",
          "type": "integer"
        }
      ]
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "name": "value",
  "status": 200
}
```

DateTimeConverter

Gunakan `datetimeConverter` prosesor untuk mengonversi string datetime menjadi format yang Anda tentukan.

Bidang	Deskripsi	Diperlukan?	Default	Batas
sumber	Kunci untuk menerapkan konversi tanggal ke.	Ya		Entri maksimum: 10
MatchPatterns	Daftar pola yang cocok dengan source bidang	Ya		Entri maksimal: 5
target	Bidang JSON untuk menyimpan hasilnya.	Ya		Panjang maksimum: 128 Kedalaman kunci bersarang maksimum: 3
TargetFormat	Format datetime yang akan digunakan untuk data yang dikonversi di bidang target.	Tidak	yyyy-MM-dd'T'HH:mm:ss.SSS'Z'	Panjang maksimum: 64
SourceTimezone	Zona waktu bidang sumber. Untuk daftar nilai yang mungkin, lihat Id dan Offset Zona yang Didukung Java .	Tidak	UTC	Panjang minimum: 1

Bidang	Deskripsi	Diperlukan?	Default	Batas
TargetTimezone	Zona waktu bidang target. Untuk daftar nilai yang mungkin, lihat Id dan Offset Zona yang Didukung Java .	Tidak	UTC	Panjang minimum:1
tempat	Lokal bidang sumber. Untuk daftar nilai yang mungkin, lihat Metode Locale getAvailableLocales () di Java dengan Contoh .	Ya		Panjang minimum:1

Contoh

Ambil contoh peristiwa log berikut:

```
{"german_datetime": "Samstag 05. Dezember 1998 11:00:00"}
```

Konfigurasi transformator adalah ini, menggunakan `dateTimeConverter` dengan `parseJSON`:

```
[
  {
    "parseJSON": {}
  },
  {
    "dateTimeConverter": {
      "source": "german_datetime",
      "target": "target_1",
      "locale": "de",
      "matchPatterns": ["EEEE dd. MMMM yyyy HH:mm:ss"],
      "sourceTimezone": "Europe/Berlin",
      "targetTimezone": "America/New_York",
      "targetFormat": "yyyy-MM-dd'T'HH:mm:ss z"
    }
  }
]
```

Peristiwa log yang diubah adalah sebagai berikut.

```
{
  "german_datetime": "Samstag 05. Dezember 1998 11:00:00",
  "target_1": "1998-12-05T17:00:00 MEZ"
}
```

Metrik dan kesalahan transformasi

CloudWatch Log menerbitkan metrik transformasi ke CloudWatch Metrik ini termasuk `TransformedLogEvents`, `TransformedBytes`, dan `TransformationErrors`. Untuk informasi selengkapnya, lihat [Metrik dan dimensi transformator log](#).

Setiap kali CloudWatch Log mencoba dan gagal mengubah peristiwa log, ia menambahkan bidang `@transformationError` sistem ke peristiwa log itu. Saat menjalankan kueri Wawasan CloudWatch Log, Anda akan melihat bidang ini di semua peristiwa log yang mengalami kegagalan transformasi. Anda dapat menanyakan bidang ini dengan kueri seperti `filter ispresent(@transformationError)` untuk menemukan semua peristiwa transformasi yang gagal.

Analisis dengan Amazon OpenSearch Service

CloudWatch Log terintegrasi Amazon OpenSearch Service untuk memungkinkan Anda membuat dasbor terkurasi otomatis yang menampilkan metrik kunci yang diperoleh OpenSearch Layanan dari log yang dijual dari layanan. AWS Dasbor berikut tersedia:

- Dasbor log aliran VPC Amazon menangkap data aliran jaringan untuk Amazon VPC. Ini membantu Anda menganalisis lalu lintas jaringan, mendeteksi pola yang tidak biasa, dan memantau penggunaan sumber daya. Metrik utama yang ditampilkan meliputi:
 - Arus total dan penerimaan serta penolakan arus ini
 - Pola lalu lintas dari waktu ke waktu
 - Diagram Sankey yang menggambarkan aliran data antara sumber dan tujuan IPs (pembicara teratas)
 - Teratas IPs dengan byte dan paket yang ditransfer

Note

Saat ini hanya format bidang VPC versi 2 yang didukung.

- Dasbor AWS WAF log memberikan wawasan tentang lalu lintas web yang dipantau oleh. AWS WAF Dasbor ini membantu Anda mengidentifikasi pola lalu lintas, permintaan yang diblokir, dan potensi ancaman dari wilayah tertentu atau IPs. Metrik utama yang ditampilkan meliputi:
 - Total permintaan, termasuk dengan jumlah “ALLOW” dan “BLOCK”.
 - Riwayat permintaan dari waktu ke waktu, menampilkan permintaan yang diizinkan dan diblokir.
 - Rincian permintaan berdasarkan nama Web ACL, permintaan yang diblokir dengan mengakhiri aturan, dan sumber. IPs
 - Distribusi geografis asal permintaan.
 - Klien teratas IPs dan aturan penghentian berdasarkan jumlah permintaan.
- Dasbor CloudTrail log memberikan ikhtisar aktivitas API di AWS lingkungan Anda menggunakan CloudTrail log. Ini berguna untuk memantau aktivitas API, tindakan audit, dan mengidentifikasi potensi masalah keamanan atau kepatuhan. Metrik utama yang ditampilkan meliputi:
 - Total jumlah acara dan riwayat acara dari waktu ke waktu
 - Rincian peristiwa berdasarkan akun IDs, kategori, dan Wilayah.
 - Top APIs, layanan, dan sumber IPs yang terlibat dalam menghasilkan acara.

- Tabel pengguna teratas yang menghasilkan peristiwa, merinci informasi akun pengguna, dan jumlah acara.
- AWS Network FirewallDasbor memberikan visibilitas yang ditingkatkan ke lalu lintas jaringan, menawarkan wawasan berharga untuk pemantauan dan analisis keamanan. Dasbor ini menawarkan pandangan komprehensif dari berbagai metrik dan pola jaringan, untuk dengan cepat mengidentifikasi potensi masalah keamanan dan mengoptimalkan konfigurasi jaringan. Metrik utama yang ditampilkan meliputi:
 - Pembicara dan protokol teratas
 - Wawasan tentang titik akhir PrivateLink
 - Lalu lintas Indikasi Nama Server TLS yang diizinkan dan diblokir

Metrik yang ditampilkan di dasbor yang dikurasi ini berasal dari analitik. Amazon OpenSearch Service

Sebelum Anda dapat melihat dasbor ini, Anda harus membuat peran IAM dan melakukan integrasi satu kali Log dengan CloudWatch . Amazon OpenSearch Service Integrasi satu kali ini mengonfigurasi Amazon OpenSearch Service sumber daya yang diperlukan untuk membuat dan merender dasbor. Anda akan dikenakan biaya untuk OpenSearch layanan yang digunakan. Untuk informasi selengkapnya, lihat [CloudWatch Harga Amazon](#).

Anda dapat membuat dasbor yang dikurasi ini hanya untuk grup log di Kelas Log Standar.

 Important

Jangan gunakan [transformator log](#) untuk grup log apa pun yang ingin Anda buat dasbor log penjual. Mengubah peristiwa log akan menyebabkan dasbor memiliki data kosong.

Topik

- [Langkah 1: Buat integrasi dengan OpenSearch Layanan](#)
- [Langkah 2: Buat dasbor log vendes](#)
- [Melihat, mengedit, atau menghapus dasbor log terjual](#)
- [Kebijakan IAM untuk pengguna](#)
- [Izin yang dibutuhkan integrasi](#)

Langkah 1: Buat integrasi dengan OpenSearch Layanan

Langkah pertama adalah membuat integrasi dengan OpenSearch Layanan, yang perlu Anda lakukan hanya sekali. Membuat integrasi akan membuat sumber daya berikut di akun Anda.

- [Koleksi deret OpenSearch Service waktu](#) tanpa ketersediaan tinggi.

Koleksi adalah sekumpulan indeks OpenSearch Layanan yang bekerja sama untuk mendukung beban kerja.

- Dua kebijakan keamanan untuk koleksi. Satu mendefinisikan jenis enkripsi, yang baik dengan kunci yang dikelola pelanggan atau AWS KMS kunci yang dimiliki layanan. Kebijakan lain mendefinisikan akses jaringan, memungkinkan aplikasi OpenSearch Layanan untuk mengakses koleksi. Untuk informasi selengkapnya, lihat [Enkripsi data saat istirahat untuk OpenSearch Layanan Amazon](#).
- [Kebijakan akses data OpenSearch Layanan](#) yang menentukan siapa yang dapat mengakses data dalam koleksi.
- [Sumber data kueri langsung OpenSearch Layanan](#) dengan CloudWatch Log didefinisikan sebagai sumber.
- [Aplikasi OpenSearch Layanan](#) dengan nama `aws-analytics`. Aplikasi akan dikonfigurasi untuk memungkinkan pembuatan ruang kerja. Jika aplikasi bernama `aws-analytics` sudah ada, itu akan diperbarui untuk menambahkan koleksi ini sebagai sumber data.
- [Ruang kerja OpenSearch Layanan yang](#) akan meng-host dasbor dan memungkinkan semua orang yang telah diberikan akses untuk membaca dari ruang kerja.

Topik

- [Izin yang diperlukan](#)
- [Buat integrasi](#)

Izin yang diperlukan

Untuk membuat integrasi, Anda harus masuk ke akun yang memiliki kebijakan IAM `CloudWatchOpenSearchDashboardsFullAccess`terkelola atau izin yang setara, ditampilkan di sini. Anda juga harus memiliki izin ini untuk menghapus integrasi, membuat, mengedit, dan menghapus dasbor, dan menyegarkan dasbor secara manual.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "CloudWatchOpenSearchDashboardsIntegration",
    "Effect": "Allow",
    "Action": [
      "logs:ListIntegrations",
      "logs:GetIntegration",
      "logs>DeleteIntegration",
      "logs:PutIntegration",
      "logs:DescribeLogGroups",
      "opensearch:ApplicationAccessAll",
      "iam:ListRoles",
      "iam:ListUsers"
    ],
    "Resource": "*"
  },
  {
    "Sid": "CloudWatchLogsOpensearchReadAPIs",
    "Effect": "Allow",
    "Action": [
      "aoss:BatchGetCollection",
      "aoss:BatchGetLifecyclePolicy",
      "es:ListApplications"
    ],
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "aws:CalledViaFirst": "logs.amazonaws.com"
      }
    }
  },
  {
    "Sid": "CloudWatchLogsOpensearchCreateServiceLinkedAccess",
    "Effect": "Allow",
    "Action": [
      "iam:CreateServiceLinkedRole"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/
opensearchservice.amazonaws.com/AWSServiceRoleForAmazonOpenSearchService",
    "Condition": {
```

```

        "StringEquals": {
            "iam:AWSServiceName": "opensearchservice.amazonaws.com",
            "aws:CalledViaFirst": "logs.amazonaws.com"
        }
    },
    {
        "Sid": "CloudWatchLogsObservabilityCreateServiceLinkedAccess",
        "Effect": "Allow",
        "Action": [
            "iam:CreateServiceLinkedRole"
        ],
        "Resource": "arn:aws:iam::*:role/aws-service-role/observability.aoss.amazonaws.com/AWSServiceRoleForAmazonOpenSearchServerless",
        "Condition": {
            "StringEquals": {
                "iam:AWSServiceName": "observability.aoss.amazonaws.com",
                "aws:CalledViaFirst": "logs.amazonaws.com"
            }
        }
    },
    {
        "Sid": "CloudWatchLogsCollectionRequestAccess",
        "Effect": "Allow",
        "Action": [
            "aoss:CreateCollection"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:CalledViaFirst": "logs.amazonaws.com",
                "aws:RequestTag/CloudWatchOpenSearchIntegration": [
                    "Dashboards"
                ]
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": "CloudWatchOpenSearchIntegration"
            }
        }
    },
    {
        "Sid": "CloudWatchLogsApplicationRequestAccess",
        "Effect": "Allow",
        "Action": [

```

```

        "es:CreateApplication"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:RequestTag/OpenSearchIntegration": [
                "Dashboards"
            ]
        },
        "ForAllValues:StringEquals": {
            "aws:TagKeys": "OpenSearchIntegration"
        }
    }
},
{
    "Sid": "CloudWatchLogsCollectionResourceAccess",
    "Effect": "Allow",
    "Action": [
        "aoss:DeleteCollection"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                "Dashboards"
            ]
        }
    }
},
{
    "Sid": "CloudWatchLogsApplicationResourceAccess",
    "Effect": "Allow",
    "Action": [
        "es:UpdateApplication",
        "es:GetApplication"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:ResourceTag/OpenSearchIntegration": [
                "Dashboards"
            ]
        }
    }
}

```

```

        ]
      }
    },
    {
      "Sid": "CloudWatchLogsCollectionPolicyAccess",
      "Effect": "Allow",
      "Action": [
        "aoss:CreateSecurityPolicy",
        "aoss:CreateAccessPolicy",
        "aoss:DeleteAccessPolicy",
        "aoss:DeleteSecurityPolicy",
        "aoss:GetAccessPolicy",
        "aoss:GetSecurityPolicy"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aoss:collection": "cloudwatch-logs-*",
          "aws:CalledViaFirst": "logs.amazonaws.com"
        }
      }
    },
    {
      "Sid": "CloudWatchLogsAPIAccessAll",
      "Effect": "Allow",
      "Action": [
        "aoss:APIAccessAll"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aoss:collection": "cloudwatch-logs-*"
        }
      }
    },
    {
      "Sid": "CloudWatchLogsIndexPolicyAccess",
      "Effect": "Allow",
      "Action": [
        "aoss:CreateAccessPolicy",
        "aoss:DeleteAccessPolicy",
        "aoss:GetAccessPolicy",
        "aoss:CreateLifecyclePolicy",

```

```

        "aoss:DeleteLifecyclePolicy"
    ],
    "Resource": "*",
    "Condition": {
        "StringLike": {
            "aoss:index": "cloudwatch-logs-*",
            "aws:CalledViaFirst": "logs.amazonaws.com"
        }
    }
},
{
    "Sid": "CloudWatchLogsDQSRequestQueryAccess",
    "Effect": "Allow",
    "Action": [
        "es:AddDirectQueryDataSource"
    ],
    "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:RequestTag/CloudWatchOpenSearchIntegration": [
                "Dashboards"
            ]
        },
        "ForAllValues:StringEquals": {
            "aws:TagKeys": "CloudWatchOpenSearchIntegration"
        }
    }
},
{
    "Sid": "CloudWatchLogsStartDirectQueryAccess",
    "Effect": "Allow",
    "Action": [
        "opensearch:StartDirectQuery",
        "opensearch:GetDirectQuery"
    ],
    "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*"
},
{
    "Sid": "CloudWatchLogsDQSResourceQueryAccess",
    "Effect": "Allow",
    "Action": [
        "es:GetDirectQueryDataSource",
        "es>DeleteDirectQueryDataSource"
    ]
}

```

```

    ],
    "Resource": "arn:aws:opensearch:*:*:datasource/cloudwatch_logs_*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                "Dashboards"
            ]
        }
    }
},
{
    "Sid": "CloudWatchLogsPassRoleAccess",
    "Effect": "Allow",
    "Action": [
        "iam:PassRole"
    ],
    "Resource": "*",
    "Condition": {
        "StringLike": {
            "iam:PassedToService":
"directquery.opensearchservice.amazonaws.com",
            "aws:CalledViaFirst": "logs.amazonaws.com"
        }
    }
},
{
    "Sid": "CloudWatchLogsAossTagsAccess",
    "Effect": "Allow",
    "Action": [
        "aoss:TagResource"
    ],
    "Resource": "arn:aws:aoss:*:*:collection/*",
    "Condition": {
        "StringEquals": {
            "aws:CalledViaFirst": "logs.amazonaws.com",
            "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
                "Dashboards"
            ]
        },
        "ForAllValues:StringEquals": {
            "aws:TagKeys": "CloudWatchOpenSearchIntegration"
        }
    }
}

```

```

    },
    {
      "Sid": "CloudWatchLogsEsApplicationTagsAccess",
      "Effect": "Allow",
      "Action": [
        "es:AddTags"
      ],
      "Resource": "arn:aws:opensearch:*:*:application/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/OpenSearchIntegration": [
            "Dashboards"
          ],
          "aws:CalledViaFirst": "logs.amazonaws.com"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": "OpenSearchIntegration"
        }
      }
    },
  ],
  {
    "Sid": "CloudWatchLogsEsDataSourceTagsAccess",
    "Effect": "Allow",
    "Action": [
      "es:AddTags"
    ],
    "Resource": "arn:aws:opensearch:*:*:datasource/*",
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/CloudWatchOpenSearchIntegration": [
          "Dashboards"
        ],
        "aws:CalledViaFirst": "logs.amazonaws.com"
      },
      "ForAllValues:StringEquals": {
        "aws:TagKeys": "CloudWatchOpenSearchIntegration"
      }
    }
  }
]
}

```

Buat integrasi

Gunakan langkah-langkah ini untuk membuat integrasi.

Untuk mengintegrasikan CloudWatch Log dengan Amazon OpenSearch Service

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Wawasan Log dan kemudian pilih tab Analisis dengan OpenSearch.
3. Pilih Buat integrasi.
4. Untuk nama Integrasi, masukkan nama untuk integrasi.
5. (Opsional) Untuk mengenkripsi data yang ditulis ke OpenSearch Service Serverless, masukkan ARN kunci yang ingin Anda gunakan dalam ARN AWS KMS kunci KMS. Untuk informasi selengkapnya, lihat [Enkripsi saat istirahat](#) di Panduan Pengembang OpenSearch Layanan Amazon.
6. Untuk penyimpanan data, masukkan jumlah waktu yang Anda inginkan agar indeks data OpenSearch Layanan dipertahankan. Ini juga menentukan periode waktu maksimum di mana Anda dapat melihat data di dasbor. Memilih periode retensi data yang lebih lama akan menimbulkan biaya pencarian dan pengindeksan tambahan. Untuk informasi selengkapnya, lihat [OpenSearch Harga Tanpa Server Layanan](#).

Periode retensi maksimum adalah 30 hari.

Panjang retensi data juga akan digunakan untuk membuat kebijakan siklus hidup pengumpulan OpenSearch Layanan.

7. Untuk peran IAM untuk menulis ke OpenSearch koleksi, buat peran IAM baru atau pilih peran IAM yang ada yang akan digunakan untuk menulis ke koleksi Layanan. OpenSearch

Membuat peran baru adalah metode paling sederhana, dan peran akan dibuat dengan izin yang diperlukan.

Note

Jika Anda membuat peran, itu akan memiliki izin untuk membaca dari semua grup log di akun.

Jika Anda ingin memilih peran yang ada, itu harus memiliki izin yang tercantum [diizin yang dibutuhkan integrasi](#). Atau, Anda dapat memilih Gunakan peran yang ada dan kemudian di bagian Verifikasi izin akses dari peran yang dipilih, Anda dapat memilih Buat peran. Dengan cara ini Anda dapat menggunakan izin yang tercantum [Izin yang dibutuhkan integrasi](#) sebagai templat dan memodifikasinya. Misalnya, jika Anda ingin menentukan kontrol butir yang lebih baik dari grup log.

8. Untuk peran IAM dan pengguna yang dapat melihat dasbor, Anda memilih cara Anda ingin memberikan akses ke peran IAM dan pengguna IAM untuk akses dasbor log penjual otomatis:
 - Untuk membatasi akses dasbor ke beberapa pengguna saja, pilih Pilih peran IAM dan pengguna yang dapat melihat dasbor dan kemudian di kotak teks cari dan pilih peran IAM dan pengguna IAM yang ingin Anda berikan akses.
 - Untuk memberikan akses dasbor ke semua pengguna, pilih Izinkan semua peran dan pengguna di akun ini untuk melihat dasbor.

 Important

Memilih peran atau pengguna, atau memilih semua pengguna, hanya menambahkannya ke [kebijakan akses data](#) yang diperlukan untuk mengakses koleksi OpenSearch Layanan yang menyimpan data dasbor. Agar mereka dapat melihat dasbor log penjual otomatis, Anda juga harus memberikan peran tersebut dan pengguna kebijakan IAM [CloudWatchOpenSearchDashboardAccess](#) terkelola.

9. Pilih Buat integrasi

Membuat integrasi akan memakan waktu beberapa menit.

Langkah 2: Buat dasbor log vended

Setelah Anda membuat integrasi, Anda dapat membuat dasbor. Dasbor tersedia untuk log aliran, log CloudTrail, dan log Amazon VPC. AWS WAF

Untuk membuat dasbor log vended dengan metrik yang diturunkan oleh Service OpenSearch

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Wawasan Log dan kemudian pilih tab Analisis dengan OpenSearch.

3. Pilih Buat dasbor.
4. Pilih jenis log untuk membuat dasbor, AWS WAF, log aliran VPC Amazon, CloudTrail, atau. AWS Network Firewall
5. Masukkan nama untuk dasbor, dan secara opsional masukkan deskripsi.
6. Untuk frekuensi sinkronisasi data, masukkan seberapa sering Anda ingin OpenSearch Layanan melakukan kueri CloudWatch sehingga metrik dan indeks yang dibuat di OpenSearch Layanan dapat disinkronkan dan diperbarui dengan data baru. OpenSearch Layanan membuat metrik dan indeks pada log Anda untuk merender dasbor.

Memilih waktu yang lebih singkat membuat data lebih mutakhir dan menimbulkan biaya yang lebih tinggi.

7. Pilih grup log untuk mengumpulkan data dari dasbor ini. Pastikan untuk memilih grup log yang cocok dengan jenis dasbor yang Anda buat.

Anda dapat menggunakan tombol Jelajahi grup log dan opsi Lihat sampel log dari grup log yang dipilih saat Anda membuat pilihan ini, untuk memastikan bahwa Anda mendapatkan grup log yang Anda inginkan.

8. Pilih Buat dasbor.

Pada awalnya, dasbor muncul tanpa data apa pun. Setelah beberapa menit, data akan muncul di dasbor. Ketika data pertama kali muncul, itu akan menjadi entri log 15 menit terakhir.

Melihat, mengedit, atau menghapus dasbor log terjual

Lihat dasbor log yang dijual di CloudWatch Log atau Layanan OpenSearch

Untuk dapat melihat dasbor, Anda harus masuk ke kepala sekolah IAM yang memiliki kebijakan CloudWatchOpenSearchDashboardAccessIAM.

Untuk melihat dasbor log yang dijual

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Wawasan Log dan kemudian pilih tab Analisis dengan OpenSearch.
3. Pilih dasbor di kotak OpenSearch dasbor.
4. (Opsional) Di kanan atas, pilih Lihat di OpenSearch.

Konsol OpenSearch Layanan terbuka dan Anda melihat dasbor yang sama di sana. Di konsol OpenSearch Layanan, Anda dapat membuat perubahan pada dasbor dan widgetnya, dan perubahan ini juga akan terlihat saat Anda melihat dasbor di CloudWatch Log.

Berikan akses tampilan dasbor ke peran IAM tambahan atau pengguna IAM

Untuk memberikan akses ke prinsipal IAM tambahan setelah Anda membuat integrasi, lakukan langkah-langkah berikut.

Untuk memberikan akses dasbor log penjual otomatis ke peran atau pengguna IAM tambahan

1. Edit kebijakan akses data untuk koleksi untuk menambahkan peran atau pengguna ini. Untuk informasi selengkapnya, lihat [Kontrol akses data untuk Amazon OpenSearch Service Tanpa Server](#) di Panduan Pengembang OpenSearch Layanan.
2. Berikan CloudWatchOpenSearchDashboardAccess kepada pengguna ini. Untuk informasi selengkapnya tentang isi kebijakan ini, lihat [CloudWatchOpenSearchDashboardAccess](#).

Edit konfigurasi dasbor

Anda dapat mengedit nama, deskripsi, dan frekuensi sinkronisasi dasbor log penjual yang ada.

Untuk mengedit dasbor log penjual

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Wawasan Log dan kemudian pilih tab Analisis dengan OpenSearch.
3. Pilih dasbor di kotak OpenSearch dasbor.
4. Pilih Tindakan, Ubah detail dasbor.
5. Buat perubahan, lalu pilih Konfirmasi perubahan.

Hapus dasbor log penjual

Anda dapat menghapus dasbor log penjual. Jika Anda melakukannya, dasbor, metrik, dan indeks yang dibuat dalam koleksi OpenSearch Layanan semuanya akan dihapus.

 Note

Setelah Anda menghapus dasbor log penjual otomatis, tunggu setidaknya enam jam sebelum mencoba membuat ulang dasbor yang sama. Jika Anda tidak menunggu, dasbor yang dibuat ulang tidak akan berfungsi dengan benar.

Untuk menghapus dasbor log penjual

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi kiri, pilih Wawasan Log dan kemudian pilih tab Analisis dengan OpenSearch.
3. Pilih dasbor di kotak OpenSearch dasbor.
4. Pilih Tindakan, Hapus.
5. Konfirmasikan keputusan Anda dengan memasukkan **delete**, lalu pilih Hapus.

Hapus semua integrasi dasbor log penjual otomatis dengan Layanan OpenSearch

Anda dapat menghapus seluruh OpenSearch integrasi Anda. Jika Anda melakukannya, semua dasbor log vended dan data yang ditampilkan di dalamnya akan dihapus.

 Important

Untuk menghindari biaya yang berkelanjutan, kami sangat menyarankan Anda menghapus sumber daya berikut secara manual sebelum menghapus integrasi. Menghapus integrasi tidak secara otomatis menghapus sumber daya ini, dan setelah Anda menghapus integrasi, Anda tidak akan dapat mengakses sumber daya ini untuk menghapusnya. Untuk menemukan nama sumber daya yang akan dihapus, lihat prosedur berikut.

- [Sumber data](#)
- [Koleksinya](#)
- [Kebijakan akses data](#)
- [Kebijakan enkripsi](#)
- [Kebijakan jaringan](#)
- [Kebijakan siklus hidup](#)

Untuk menghapus seluruh integrasi dasbor log penjual Anda dengan Layanan OpenSearch

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi kiri, pilih Pengaturan.
3. Pilih tab Log.
4. Di bagian OpenSearch integrasi, pilih Hapus integrasi.

Layar berikutnya menampilkan nama-nama sumber daya OpenSearch Layanan yang harus Anda hapus sebelum menghapus integrasi.

5. Konfirmasikan keputusan Anda dengan memasukkan **delete**, lalu pilih Hapus integrasi.

Kebijakan IAM untuk pengguna

CloudWatch Log telah membuat dua kebijakan IAM, CloudWatchOpenSearchDashboardsFullAccess dan CloudWatchOpenSearchDashboardAccess. Tabel berikut mencantumkan tindakan yang diaktifkan oleh masing-masing kebijakan ini.

Tindakan	Kebijakan IAM	Izin tambahan diperlukan
Buat integrasi	CloudWatchOpenSearchDashboardsFullAccess	
Hapus integrasi	CloudWatchOpenSearchDashboardsFullAccess	
Buat dasbor	CloudWatchOpenSearchDashboardsFullAccess	
Edit dasbor	CloudWatchOpenSearchDashboardsFullAccess	
Hapus dasbor	CloudWatchOpenSearchDashboardsFullAccess	
Segarkan dasbor menggunakan Sinkronisasi sekarang	CloudWatchOpenSearchDashboardsFullAccess	

Tindakan	Kebijakan IAM	Izin tambahan diperlukan
Lihat integrasi di Pengaturan	CloudWatchOpenSearchDashboardAccess atau CloudWatchOpenSearchDashboardsFullAccess	
Lihat dasbor	CloudWatchOpenSearchDashboardAccess atau CloudWatchOpenSearchDashboardsFullAccess	Tentukan peran atau pengguna saat Anda membuat integrasi, atau edit kebijakan akses data untuk koleksi untuk menambahkan peran atau pengguna ini. Untuk informasi selengkapnya, lihat Kontrol akses data untuk Amazon OpenSearch Service Tanpa Server di Panduan Pengembang OpenSearch Layanan.
Lihat dasbor di konsol OpenSearch Layanan	CloudWatchOpenSearchDashboardAccess atau CloudWatchOpenSearchDashboardsFullAccess	Tentukan peran atau pengguna saat Anda membuat integrasi, atau edit kebijakan akses data untuk koleksi untuk menambahkan peran atau pengguna ini. Untuk informasi selengkapnya, lihat Kontrol akses data untuk Amazon OpenSearch Service Tanpa Server di Panduan Pengembang OpenSearch Layanan.

Izin yang dibutuhkan integrasi

Jika Anda membuat peran IAM agar integrasi dapat digunakan, alih-alih mengizinkan CloudWatch Log membuat peran, peran tersebut harus menyertakan izin dan kebijakan kepercayaan berikut. Untuk informasi selengkapnya tentang cara membuat peran IAM, lihat [Membuat peran untuk mendelegasikan izin ke layanan](#). AWS

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudWatchLogsAccess",
      "Effect": "Allow",
      "Action": [
        "logs:StartQuery",
        "logs:GetLogGroupFields",
        "logs:GetQueryResults"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "CloudWatchLogsDescribeLogGroupsAccess",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AmazonOpenSearchCollectionAccess",
      "Effect": "Allow",
      "Action": [
        "aoss:APIAccessAll"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aoss:collection": "cloudwatch-logs-*"
        }
      }
    }
  ]
}
```

```
}  
  }  
}  
]  
}
```

Note

Peran sebelumnya memberikan akses untuk membaca dari semua grup log di akun, untuk memungkinkan Anda membuat dasbor untuk akun log apa pun, termasuk grup log lintas akun. Jika Anda ingin membatasi akses ke grup log tertentu dan membuat dasbor hanya untuk grup log tersebut, Anda dapat memperbarui pernyataan pertama dalam kebijakan tersebut menjadi berikut:

```
{  
  "Sid": "CloudWatchLogsAccess",  
  "Effect": "Allow",  
  "Action": [  
    "logs:StartQuery",  
    "logs:GetLogGroupFields",  
    "logs:GetQueryResults"  
  ],  
  "Resource": [  
    "arn:aws:logs:us-east-1:123456789012:log-group:myLogGroup:*",  
    "arn:aws:logs:us-east-1:123456789012:log-group:myLogGroup"  
  ]  
}
```


Akses log dengan Integrasi Tabel S3

Integrasi Tabel S3 CloudWatch memungkinkan Anda mengakses data log yang tertelan CloudWatch menggunakan mesin analitik seperti Amazon Athena, Amazon Redshift, dan alat pihak ketiga yang mendukung koneksi ke toko yang kompatibel dengan Apache Iceberg. Integrasi ini memungkinkan Anda untuk melakukan analisis log komprehensif menggunakan alat preferensi Anda dan menghubungkan data dalam CloudWatch Log dengan CloudWatch non-data.

Memahami Integrasi Tabel S3

Integrasi Tabel Amazon S3 adalah solusi terkelola penuh yang membuat log Anda di CloudWatch Log tersedia sebagai tabel Amazon S3 terkelola. Dengan integrasi ini, Anda mendapatkan fleksibilitas yang lebih besar tentang cara Anda menganalisis log Anda selain fitur CloudWatch Log.

Integrasi berfungsi dengan membuat bucket tabel Amazon S3 terkelola (`aws-cloudwatch`) dan mengaitkan sumber log tertentu dengan Tabel Amazon S3 berdasarkan nama dan jenis sumber data (yang dapat dikelola dari tab Manajemen Log > Sumber Data di Konsol Log). CloudWatch Setelah dikaitkan, data CloudWatch Log dapat diakses melalui Tabel Amazon S3 menggunakan format Apache Iceberg. Format ini menyediakan cara standar untuk berbagai mesin analitik untuk menanyakan data secara efisien.

Komponen Inti

Asosiasi Sumber Data

Proses menghubungkan sumber CloudWatch Log tertentu ke integrasi Tabel S3 berdasarkan sumber data dan kriteria tipe.

Tabel Gunung Es Apache

Format tabel dasar yang digunakan oleh Tabel S3 yang menyediakan penyimpanan data terstruktur dan memungkinkan kompatibilitas dengan beberapa mesin analitik.

Aliran data ke tabel S3

Memahami bagaimana aliran data antara CloudWatch Log dan Tabel S3 membantu Anda merencanakan integrasi dan mengelola data log Anda secara efektif.

Saat Anda membuat asosiasi, CloudWatch Log secara otomatis mengirimkan peristiwa log baru yang cocok dengan nama sumber data terkait dan mengetik ke bucket tabel S3 yang CloudWatch dikelola. Anda dapat menemukan peristiwa ini di namespace log di bawah tabel yang sesuai untuk sumber data tersebut. Proses integrasi hanya mencatat peristiwa yang ditambahkan setelah Anda membuat asosiasi dan tidak mengisi kembali log sebelum asosiasi dibuat.

Penyimpanan data di bucket tabel S3 cocok dengan set kebijakan retensi untuk grup log. Misalnya, jika Anda menyetel grup log ke retensi 1 hari, CloudWatch Log akan menghapus data dari CloudWatch Log dan Tabel S3 setelah satu hari. Saat Anda menghapus grup log atau aliran CloudWatch log, Log juga menghapus data dari bucket tabel S3.

Kapan Menggunakan Integrasi Tabel S3

Pertimbangkan untuk menggunakan integrasi Tabel S3 untuk mengkorelasikan data log dengan data eksternal atau CloudWatch non-data lainnya atau bila Anda lebih suka menggunakan alat analisis lain seperti Amazon Athena untuk melakukan analitik CloudWatch pada data Log. Gunakan integrasi ini saat Anda membutuhkan kemampuan yang melampaui apa yang tersedia di CloudWatch Log. Integrasi ini sangat berharga ketika:

- Anda perlu menjalankan kueri mirip SQL yang kompleks di seluruh volume besar data log
- Anda ingin mengintegrasikan analisis log dengan alur kerja dan alat analitik yang ada
- Anda memerlukan kemampuan analisis log komprehensif yang mencakup beberapa sumber data

Tidak ada biaya penyimpanan atau pemeliharaan meja tambahan untuk tabel S3 yang dibuat melalui integrasi ini, di luar CloudWatch penetapan dan harga penyimpanan yang ada.

Prasyarat

Sebelum menerapkan integrasi, pastikan Anda memiliki yang berikut:

- Data CloudWatch Log yang ada
- Izin IAM yang sesuai untuk akses lintas layanan antara CloudWatch Log dan Tabel S3, seperti yang dijelaskan di bagian berikut

Izin IAM

Untuk mengintegrasikan CloudWatch Log dengan Tabel S3, Anda perlu mengonfigurasi izin IAM untuk dua entitas terpisah: pengguna atau peran yang mengatur integrasi, dan peran layanan yang diasumsikan CloudWatch Log untuk menulis data ke Tabel S3.

Untuk peran atau pengguna yang membuat integrasi

Pengguna atau peran yang mengatur integrasi memerlukan izin berikut:

- `observabilityadmin:CreateS3TableIntegration` untuk membuat integrasi dan `logs:AssociateSourceToS3TableIntegration` menambahkan sumber
- `s3tables:CreateTableBucket`, `s3tables:PutTableBucketEncryption`, dan `s3tables:PutTableBucketPolicy` untuk mengkonfigurasi bucket tabel S3

Untuk peran layanan

Lampirkan kebijakan IAM berikut ke peran layanan IAM yang digunakan CloudWatch Log untuk menulis data ke bucket tabel. Kebijakan ini memberikan izin untuk menulis ke tabel. Ganti *aws-region123456789012*, dan *log-group-name* dengan AWS Region, ID akun, dan nama grup log Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:integrateWithS3Table"
      ],
      "Resource": ["arn:aws:logs:aws-region:123456789012:log-group:log-group-name"],
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Lampirkan kebijakan kepercayaan berikut ke peran layanan IAM yang akan diasumsikan oleh CloudWatch Log untuk menulis data log ke Tabel S3. Anda membuat atau memilih peran ini selama penyiapan integrasi. Kondisi membatasi peran sehingga CloudWatch Log hanya dapat menganggapnya untuk akun dan grup log yang ditentukan. Ganti *aws-region123456789012*, dan *log-group-name* dengan AWS Region, ID akun, dan nama grup log Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": ["arn:aws:logs:aws-region:123456789012:log-
group:log-group-name"]
        }
      }
    }
  ]
}
```

Kebijakan kunci KMS (untuk data terenkripsi)

Jika Anda menggunakan kunci yang dikelola pelanggan untuk mengenkripsi data log Anda, Anda harus memberikan akses utama CloudWatch layanan dan prinsipal layanan pemeliharaan Tabel S3 ke kunci tersebut. Tambahkan pernyataan berikut ke kebijakan kunci KMS Anda. Ganti nilai placeholder dengan ID, Region, Akun AWS ID kunci KMS, dan ARN tabel atau tabel S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableSystemTablesKeyUsage",
      "Effect": "Allow",
```

```

    "Principal": {
      "Service": "systemtables.cloudwatch.amazonaws.com"
    },
    "Action": [
      "kms:DescribeKey",
      "kms:GenerateDataKey",
      "kms:Decrypt"
    ],
    "Resource": "arn:aws:kms:aws-region:123456789012:key/key-id",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  },
  {
    "Sid": "EnableKeyUsage",
    "Effect": "Allow",
    "Principal": {
      "Service": "maintenance.s3tables.amazonaws.com"
    },
    "Action": [
      "kms:GenerateDataKey",
      "kms:Decrypt"
    ],
    "Resource": "arn:aws:kms:aws-region:123456789012:key/key-id",
    "Condition": {
      "StringLike": {
        "kms:EncryptionContext:aws:s3:arn": "<table-or-table-bucket-arn>/*"
      }
    }
  }
]
}

```

Memulai

Untuk memulai Integrasi Tabel S3, Anda perlu mengatur integrasi antara CloudWatch Log dan Tabel S3 Anda. Proses ini melibatkan konfigurasi asosiasi sumber data dan pengaturan izin IAM yang sesuai.

Untuk membuat Integrasi Tabel S3

1. Buka konsol CloudWatch Log di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Pengaturan, Global, Buat Integrasi Tabel S3.
3. Sesuaikan bagaimana log akan dienkripsi dalam Tabel S3, dan peran yang akan digunakan Log untuk menulis CloudWatch log Anda ke Tabel S3.
4. Pilih Buat Integrasi Tabel S3.

Untuk mengaitkan sumber ke Integrasi Tabel S3

1. Buka konsol CloudWatch Log di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Pengaturan, Global, Kelola Integrasi Tabel S3.
3. Pilih Sumber data asosiasi.
4. Pilih nama sumber data dan tipe sumber data yang ingin Anda aktifkan integrasi.
5. Pilih Sumber data asosiasi.

Untuk mengaitkan sumber ke Integrasi Tabel S3 dari Halaman Manajemen Log

1. Buka konsol CloudWatch Log di <https://console.aws.amazon.com/cloudwatch/>.
2. Pilih Manajemen Log di panel navigasi.
3. Pilih tab Sumber Data.
4. Pilih nama sumber data dan tipe sumber data yang ingin Anda integrasikan.
5. Pilih Tindakan sumber data.
6. Pilih Kaitkan dengan Integrasi Tabel S3.
7. Tinjau sumber data, lalu pilih Sumber Data Asosiasi.

Sebelum Anda dapat menggunakan data, Anda harus melakukan 3 langkah berikut:

1. Mengintegrasikan Tabel Amazon S3 Amazon AWS dengan layanan analitik - Menggunakan konsol Amazon S3
2. Konfigurasikan Izin Lake Formation
3. Connect dengan Analytics Tools

Mengintegrasikan Tabel Amazon S3 Amazon AWS dengan layanan analitik - Menggunakan konsol Amazon S3 (Tautan)

Untuk mengaktifkan integrasi Tabel S3 menggunakan konsol S3

1. Buka konsol Amazon S3 di <https://console.aws.amazon.com/s3/>
2. Di panel navigasi kiri, pilih Bucket tabel.
3. Klik Aktifkan integrasi di bagian atas.
4. Pertama kali Anda mengintegrasikan bucket tabel di Wilayah mana pun, Amazon Amazon S3 membuat peran layanan IAM baru atas nama Anda. Peran ini memungkinkan Lake Formation mengakses semua bucket tabel di akun Anda dan menggabungkan akses ke tabel Anda di AWS Glue Data Catalog.

Konfigurasi Izin Lake Formation

Meskipun CloudWatch Log memiliki izin untuk menulis ke tabel (dikonfigurasi pada langkah sebelumnya), pengguna dan peran analitik tidak secara otomatis memiliki izin untuk membaca data. Anda harus secara eksplisit memberikan akses menggunakan AWS Lake Formation. Anda harus melakukan ini untuk setiap kepala sekolah IAM yang ingin Anda berikan akses ke tabel.

Untuk memberikan akses kueri ke pengguna atau peran

Anda harus memberikan izin SELECT dan DESCRIBE kepada prinsipal IAM (pengguna atau peran) yang akan menjalankan kueri di Athena atau Redshift.

1. Buka konsol AWS Lake Formation.
2. Di panel navigasi, di bawah Izin, pilih Izin danau data.
3. Pilih Izin.
4. Prinsip: Pilih pengguna IAM atau peran yang memerlukan akses (misalnya, analisis data Anda atau peran Admin yang sedang Anda gunakan).
5. Tag LF atau sumber daya Katalog: Pilih sumber daya Katalog Data Bernama.
6. Database dan Tabel:
 - Pilih bucket Table S3 yang dibuat oleh CloudWatch integration (`aws-cloudwatch`).
 - Pilih tabel spesifik yang terkait dengan sumber data Anda (opsional).
7. Izin Tabel: Pilih dan Jelaskan.

8. PilihIzin.

Note

Jika Anda menemukan kesalahan “Akses Ditolak” saat menanyakan log di Athena, pastikan bahwa pengguna yang menjalankan kueri memiliki izin IAM yang sesuai untuk Athena dan izin Lake Formation yang ditentukan di atas.

Pelajari lebih lanjut tentang izin Lake Formation di <https://docs.aws.amazon.com/lake-formation/latest/dg/granting-catalog-permissions.html>.

Connect dengan Analytics Tools

Setelah izin diberikan, Anda dapat mengonfigurasi layanan analitik pilihan Anda untuk menanyakan Tabel S3. Tabel S3 menggunakan format Apache Iceberg, yang secara native didukung oleh Amazon Athena, Amazon Redshift, dan Amazon EMR.

Untuk menanyakan data log di Amazon Athena

Amazon Athena berinteraksi dengan Tabel S3 melalui katalog Tabel Amazon S3.

Untuk mengatur Athena untuk menanyakan data log Anda

1. Buka konsol Amazon Athena di <https://console.aws.amazon.com/athena/>
2. Di editor kueri, pilih katalog Amazon S3 Tables dari dropdown sumber data.
3. Jika Anda tidak melihat katalog, pastikan Anda telah menyelesaikan langkah-langkah izin Lake Formation di atas untuk peran pengguna spesifik Anda.
4. Setelah katalog dipilih, tabel log Anda akan muncul di daftar database. Anda sekarang dapat menjalankan kueri SQL standar terhadap data log Anda.

Contoh Query: `SELECT * FROM "amazon_vpc__flow" LIMIT 100;`

Pelajari lebih lanjut tentang menghubungkan layanan Analytics dengan Tabel S3 di <https://docs.aws.amazon.com/AmazonS3/latest/userguide/s3-tables-integrating-aws.html>.

Membuat metrik dari peristiwa log dengan menggunakan filter

Anda dapat mencari dan memfilter data log yang masuk ke CloudWatch Log dengan membuat satu atau beberapa filter metrik. Filter metrik menentukan istilah dan pola yang harus dicari dalam data log saat dikirim ke CloudWatch Log. CloudWatch Log menggunakan filter metrik ini untuk mengubah data log menjadi CloudWatch metrik numerik yang dapat Anda buat grafik atau nyalakan alarm.

Saat membuat metrik dari filter log, Anda juga dapat memilih untuk menetapkan dimensi dan unit ke metrik. Jika Anda menentukan unit, pastikan untuk menentukan yang benar saat Anda membuat filter. Mengubah unit untuk filter nanti tidak akan berpengaruh.

Jika Anda telah mengonfigurasi AWS Organizations dan bekerja dengan akun anggota, Anda dapat menggunakan sentralisasi log untuk mengumpulkan data log dari akun sumber ke akun pemantauan pusat.

Saat bekerja dengan grup log terpusat, Anda dapat menggunakan dimensi bidang sistem ini saat membuat filter metrik.

- `@aws.account`- Dimensi ini mewakili ID AWS akun dari mana peristiwa log berasal.
- `@aws.region`- Dimensi ini mewakili AWS wilayah tempat peristiwa log dihasilkan.

Dimensi ini membantu dalam mengidentifikasi sumber data log, memungkinkan penyaringan dan analisis metrik yang lebih terperinci yang berasal dari log terpusat. Untuk informasi selengkapnya, lihat [Sentralisasi log lintas wilayah lintas akun](#).

Jika grup log dengan langganan menggunakan transformasi log, pola filter diterapkan ke versi peristiwa log yang diubah. Untuk informasi selengkapnya, lihat [Ubah log selama konsumsi](#).

Note

Filter metrik hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

Anda dapat menggunakan semua jenis CloudWatch statistik, termasuk statistik persentil, saat melihat metrik ini atau mengatur alarm.

 Note

Statistik persentil hanya didukung untuk metrik jika tidak ada nilai metrik yang negatif. Jika Anda mengatur filter metrik sehingga dapat melaporkan angka negatif, statistik persentil tidak akan tersedia untuk metrik tersebut saat ada angka negatif sebagai nilai. Untuk informasi selengkapnya, lihat [Persentil](#).

Filter tidak memfilter data secara retroaktif. Filter hanya memublikasikan titik data metrik untuk kejadian yang terjadi setelah filter dibuat. Saat menguji pola filter, pratinjau hasil Filter menampilkan hingga 50 baris log pertama yang cocok untuk tujuan validasi. Jika stempel waktu pada hasil yang difilter lebih awal dari waktu pembuatan metrik, tidak ada log yang ditampilkan.

Daftar Isi

- [Konsep](#)
- [Filter sintaks pola untuk filter metrik](#)
- [Membuat filter metrik](#)
- [Daftar filter metrik](#)
- [Menghapus filter metrik](#)

Konsep

Setiap filter metrik terdiri dari elemen kunci berikut:

nilai default

Nilai yang dilaporkan ke filter metrik selama periode ketika log dicerna tetapi tidak ada log yang cocok ditemukan. Dengan menyetel ini ke 0, Anda memastikan bahwa data dilaporkan selama setiap periode tersebut, mencegah metrik “jerawatan” dengan periode tanpa data yang cocok. Jika tidak ada log yang dicerna selama periode satu menit, maka tidak ada nilai yang dilaporkan.

Jika Anda menetapkan dimensi ke metrik yang dibuat oleh filter metrik, Anda tidak dapat menetapkan nilai default untuk metrik tersebut.

dimensi

Dimensi adalah pasangan kunci-nilai yang menentukan metrik lebih lanjut. Anda dapat menetapkan dimensi ke metrik yang dibuat dari filter metrik. Karena dimensi adalah bagian dari

pengenal unik untuk metrik, setiap kali name/value pasangan unik diekstraksi dari log Anda, Anda membuat variasi baru dari metrik tersebut.

pola filter

Deskripsi simbolis tentang bagaimana CloudWatch Log harus menafsirkan data di setiap peristiwa log. Sebagai contoh, entri log mungkin berisi stempel waktu, alamat IP, string, dan sebagainya. Anda menggunakan pola untuk menentukan apa yang harus dicari dalam berkas log.

nama metrik

Nama CloudWatch metrik tempat informasi log yang dipantau harus dipublikasikan. Misalnya, Anda dapat mempublikasikan ke metrik yang disebut `ErrorCount`.

namespace metrik

Namespace tujuan metrik baru CloudWatch .

nilai metrik

Nilai numerik untuk dipublikasikan ke metrik setiap kali log yang cocok ditemukan. Sebagai contoh, jika Anda menghitung kemunculan istilah tertentu, seperti "Error", nilainya adalah "1" untuk setiap kejadian. Jika Anda menghitung byte yang ditransfer, Anda dapat menambahkannya berdasarkan jumlah aktual byte yang ditemukan dalam log acara.

Filter sintaks pola untuk filter metrik

Note

Perbedaan filter metrik dari kueri Wawasan CloudWatch Log

Filter metrik berbeda dari kueri Wawasan CloudWatch Log karena nilai numerik tertentu ditambahkan ke filter metrik setiap kali log yang cocok ditemukan. Untuk informasi selengkapnya, lihat [Mengkonfigurasi nilai metrik untuk filter metrik](#).

Untuk informasi tentang cara menanyakan grup log Anda dengan bahasa kueri Amazon CloudWatch Logs Insights, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

Contoh pola filter generik

Untuk informasi selengkapnya tentang sintaks pola filter generik yang berlaku untuk filter metrik serta [filter langganan](#) dan [peristiwa log filter](#), lihat [Filter sintaks pola untuk filter metrik, filter langganan, dan peristiwa log filter](#), yang mencakup contoh berikut:

- Sintaks ekspresi reguler (regex) yang didukung

- Istilah pencocokan dalam peristiwa log tidak terstruktur
- Ketentuan yang cocok dalam peristiwa log JSON
- Istilah yang cocok dalam peristiwa log yang dibatasi ruang

Filter metrik memungkinkan Anda untuk mencari dan memfilter data log yang masuk ke CloudWatch Log, mengekstrak pengamatan metrik dari data log yang difilter, dan mengubah titik data menjadi metrik CloudWatch Log. Anda menentukan istilah dan pola yang harus dicari dalam data log saat dikirim ke CloudWatch Log. Filter metrik ditetapkan untuk grup log, dan semua filter yang ditetapkan ke grup log diterapkan ke pengaliran log mereka.

Ketika filter metrik cocok dengan istilah, itu menambah jumlah metrik dengan nilai numerik tertentu. Misalnya, Anda dapat membuat filter metrik yang menghitung berapa kali kata ERROR terjadi dalam peristiwa log Anda.

Anda dapat menetapkan satuan ukuran dan dimensi ke metrik. Misalnya, jika Anda membuat filter metrik yang menghitung berapa kali kata ERROR terjadi dalam peristiwa log Anda, Anda dapat menentukan dimensi yang dipanggil `ErrorCode` untuk menunjukkan jumlah total peristiwa log yang berisi kata ERROR dan memfilter data berdasarkan kode kesalahan yang dilaporkan.

Tip

Saat Anda menetapkan satuan ukuran ke metrik, pastikan untuk menentukan yang benar. Jika Anda mengubah unit nanti, perubahan Anda mungkin tidak berlaku. Untuk daftar lengkap unit yang CloudWatch mendukung, lihat [MetricDatum](#) di Referensi Amazon CloudWatch API.

Topik

- [Mengkonfigurasi nilai metrik untuk filter metrik](#)
- [Menerbitkan dimensi dengan metrik dari nilai di JSON atau peristiwa log yang dibatasi ruang](#)
- [Menggunakan nilai dalam peristiwa log untuk menambah nilai metrik](#)

Mengkonfigurasi nilai metrik untuk filter metrik

Saat Anda membuat filter metrik, Anda menentukan pola filter dan menentukan nilai metrik dan nilai default Anda. Anda dapat mengatur nilai metrik ke angka, pengidentifikasi bernama, atau pengidentifikasi numerik. Jika Anda tidak menentukan nilai default, tidak CloudWatch akan

melaporkan data saat filter metrik Anda tidak menemukan kecocokan. Kami menyarankan Anda menentukan nilai default, bahkan jika nilainya 0. Menyetel nilai default membantu CloudWatch melaporkan data dengan lebih akurat dan CloudWatch mencegah agregasi metrik jerawatan. CloudWatch agregat dan melaporkan nilai metrik setiap menit.

Ketika filter metrik Anda menemukan kecocokan dalam peristiwa log Anda, itu menambah jumlah metrik Anda dengan nilai metrik Anda. Jika filter metrik Anda tidak menemukan kecocokan, CloudWatch laporkan nilai default metrik. Misalnya, grup log Anda menerbitkan dua catatan setiap menit, nilai metriknya adalah 1, dan nilai defaultnya adalah 0. Jika filter metrik Anda menemukan kecocokan di kedua catatan log dalam menit pertama, nilai metrik untuk menit itu adalah 2. Jika filter metrik Anda tidak menemukan kecocokan di kedua rekaman selama menit kedua, nilai default untuk menit itu adalah 0. Jika Anda menetapkan dimensi ke metrik yang dihasilkan oleh filter metrik, Anda tidak dapat menentukan nilai default untuk metrik tersebut.

Anda juga dapat mengatur filter metrik untuk menambah metrik dengan nilai yang diekstrak dari peristiwa log, bukan nilai statis. Untuk informasi selengkapnya, lihat [Menggunakan nilai dalam peristiwa log untuk menambah nilai metrik](#).

Menerbitkan dimensi dengan metrik dari nilai di JSON atau peristiwa log yang dibatasi ruang

Anda dapat menggunakan CloudWatch konsol atau AWS CLI untuk membuat filter metrik yang mempublikasikan dimensi dengan metrik yang dihasilkan oleh JSON dan peristiwa log yang dibatasi ruang. Dimensi adalah pasangan name/value nilai dan hanya tersedia untuk JSON dan pola filter yang dibatasi ruang. Anda dapat membuat filter metrik JSON dan spasi-terbatas hingga tiga dimensi. Untuk informasi selengkapnya tentang dimensi dan informasi tentang cara menetapkan dimensi ke metrik, lihat bagian berikut:

- [Dimensi](#) dalam panduan CloudWatch Pengguna Amazon
- [Contoh: Ekstrak bidang dari log Apache dan tetapkan dimensi di Panduan](#) Pengguna Amazon CloudWatch Logs

Important

Dimensi berisi nilai yang mengumpulkan biaya sama dengan metrik kustom. Untuk mencegah muatan tak terduga, jangan tentukan bidang kardinalitas tinggi, seperti `IPAddress` atau `requestID`, sebagai dimensi.

Jika Anda mengekstrak metrik dari peristiwa log, Anda dikenakan biaya untuk metrik khusus. Untuk mencegah Anda mengumpulkan muatan tinggi yang tidak disengaja, Amazon mungkin menonaktifkan filter metrik Anda jika menghasilkan 1000 name/value pasangan berbeda untuk dimensi tertentu selama jangka waktu tertentu.

Anda dapat membuat alarm penagihan yang memberi tahu Anda tentang perkiraan biaya Anda. Untuk informasi selengkapnya, lihat [Membuat alarm penagihan untuk memantau perkiraan AWS biaya](#).

Memublikasikan dimensi dengan metrik dari log acara JSON

Contoh berikut berisi cuplikan kode yang menjelaskan cara menentukan dimensi dalam filter metrik JSON.

Example: JSON log event

```
{
  "eventType": "UpdateTrail",
  "sourceIPAddress": "111.111.111.111",
  "arrayKey": [
    "value",
    "another value"
  ],
  "objectList": [
    {"name": "a",
     "id": 1
    },
    {"name": "b",
     "id": 2
    }
  ]
}
```

Note

Jika Anda menguji filter metrik contoh dengan contoh peristiwa log JSON, Anda harus memasukkan contoh log JSON pada satu baris.

Example: Metric filter

Filter metrik menambah metrik setiap kali peristiwa log JSON berisi properti `eventType` dan `"sourceIPAddress"`

```
{ $.eventType = "*" && $.sourceIPAddress != 123.123.* }
```

Saat Anda membuat filter metrik JSON, Anda dapat menentukan properti apa pun di filter metrik sebagai dimensi. Misalnya, untuk mengatur `eventType` sebagai dimensi, gunakan yang berikut ini:

```
"eventType" : $.eventType
```

Contoh metrik berisi dimensi yang diberi nama `"eventType"`, dan nilai dimensi dalam peristiwa log contoh adalah `"UpdateTrail"`.

Memublikasikan dimensi dengan metrik dari log acara yang dipisahkan dengan spasi

Contoh berikut berisi cuplikan kode yang menjelaskan cara menentukan dimensi dalam filter metrik yang dibatasi ruang.

Example: Space-delimited log event

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404  
1534
```

Example: Metric filter

```
[ip, server, username, timestamp, request, status_code, bytes > 1000]
```

Filter metrik menambah metrik ketika peristiwa log yang dibatasi spasi menyertakan salah satu bidang yang ditentukan dalam filter. Misalnya, filter metrik menemukan bidang dan nilai berikut dalam contoh peristiwa log yang dibatasi ruang.

```
{
  "$bytes": "1534",
  "$status_code": "404",

  "$request": "GET /index.html HTTP/1.0",
  "$timestamp": "10/Oct/2000:13:25:15 -0700",
  "$username": "frank",
  "$server": "Prod",
  "$ip": "127.0.0.1"
}
```

Saat Anda membuat filter metrik yang dibatasi spasi, Anda dapat menentukan salah satu bidang dalam filter metrik sebagai dimensi. Misalnya, untuk mengatur `server` sebagai dimensi, gunakan yang berikut ini:

```
"server" : $server
```

Contoh filter metrik memiliki dimensi yang diberi nama `server`, dan nilai dimensi dalam peristiwa log contoh adalah `"Prod"`.

Example: Match terms with AND (&&) and OR (||)

Anda dapat menggunakan operator logis AND (“&&”) dan OR (“||”) untuk membuat filter metrik yang dibatasi spasi yang berisi kondisi. Filter metrik berikut mengembalikan peristiwa log di mana kata pertama dalam peristiwa adalah `ERROR` atau substring dari `WARN`.

```
[w1=ERROR || w1=%WARN%, w2]
```


Menggunakan nilai dalam peristiwa log untuk menambah nilai metrik

Anda dapat membuat filter metrik yang mempublikasikan nilai numerik yang ditemukan di peristiwa log Anda. Prosedur di bagian ini menggunakan contoh filter metrik berikut untuk menunjukkan bagaimana Anda dapat mempublikasikan nilai numerik dalam peristiwa log JSON ke metrik.

```
{ $.latency = * } metricValue: $.latency
```

Untuk membuat filter metrik yang menerbitkan nilai dalam peristiwa log

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Log, lalu pilih Grup log.
3. Pilih atau buat grup log.

Untuk informasi tentang cara membuat grup log, lihat [Membuat grup log di CloudWatch](#) Log di Panduan Pengguna CloudWatch Log Amazon.

4. Pilih Tindakan, lalu pilih Buat filter metrik.
5. Untuk Pola Filter{ **\$.latency = *** }, masukkan, lalu pilih Berikutnya.
6. Untuk Nama Metrik, masukkan MyMetric.
7. Untuk Metric Value (Nilai Metrik), masukkan **\$.latency**.
8. (Opsional) Untuk Nilai Default, masukkan 0, lalu pilih Berikutnya.

Kami menyarankan Anda menentukan nilai default, bahkan jika nilainya 0. Menyetel nilai default membantu CloudWatch melaporkan data dengan lebih akurat dan CloudWatch mencegah agregasi metrik jerawatan. CloudWatch agregat dan melaporkan nilai metrik setiap menit.

9. Pilih Create metric filter (Buat filter metrik).

Filter metrik contoh cocok dengan istilah **"latency"** dalam contoh peristiwa log JSON dan menerbitkan nilai numerik 50 ke MyMetric metrik.

```
{  
  "latency": 50,  
  "requestType": "GET"  
}
```

Membuat filter metrik

Prosedur dan contoh berikut menunjukkan cara membuat filter metrik.

Contoh

- [Membuat filter metrik untuk grup log](#)
- [Contoh: Hitung peristiwa log](#)
- [Contoh: Hitung kemunculan suatu istilah](#)
- [Contoh: Hitung kode HTTP 404](#)
- [Contoh: Hitung kode HTTP 4xx](#)
- [Contoh: Mengekstraksi bidang dari log Apache dan menetapkan dimensi](#)

Membuat filter metrik untuk grup log

Untuk membuat filter metrik untuk grup log, ikuti langkah-langkah ini. Metrik tidak akan terlihat sampai ada beberapa titik data untuk itu.

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Log, lalu pilih Grup log.
3. Pilih nama grup log.
4. Pilih **Actions**, lalu pilih **Buat filter metrik**.
5. Untuk pola Filter, masukkan pola filter. Untuk informasi selengkapnya, lihat [Filter sintaks pola untuk filter metrik, filter langganan, peristiwa log filter, dan Live Tail](#).
6. (Opsional) Jika Anda menggunakan grup log terpusat, di bawah Kriteria pemilihan filter, Anda dapat menentukan filter berdasarkan akun sumber (@aws.account), wilayah sumber (@aws.region), atau kedua kondisi.
7. (Opsional) Untuk menguji pola filter Anda, di bawah Pola Uji, masukkan satu atau beberapa peristiwa log untuk menguji pola. Setiap peristiwa log harus diformat pada satu baris. Jeda baris digunakan untuk memisahkan peristiwa log di kotak pesan peristiwa Log.
8. Pilih **Berikutnya**, lalu masukkan nama untuk filter metrik Anda.

9. Di bawah Detail metrik, untuk namespace Metrik, masukkan nama untuk CloudWatch namespace tempat metrik akan dipublikasikan. Jika namespace belum ada, pastikan Create new dipilih.
10. Untuk Metric name (Nama metrik), masukkan nama untuk metrik baru.
11. Untuk Metric value (Nilai metrik), jika filter metrik Anda menghitung kemunculan kata kunci dalam filter, masukkan 1. Peningkatan akan menambahkan metrik dengan kelipatan sebesar 1 untuk setiap log acara yang mencakup salah satu kata kunci.

Atau, masukkan token, seperti `$size`. Peningkatan ini akan menambahkan metrik sebesar nilai angka di bidang size untuk setiap log acara yang berisi bidang size.

12. (Opsional) Untuk Unit, pilih unit yang akan ditetapkan ke metrik. Jika Anda tidak menentukan unit, unit ditetapkan sebagai None.
13. (Opsional) Masukkan nama dan token untuk sebanyak tiga dimensi untuk metrik. Jika Anda menetapkan dimensi ke metrik yang dibuat oleh filter metrik, Anda tidak dapat menetapkan nilai default untuk metrik tersebut.

 Note

Dimensi hanya didukung di JSON atau filter metrik yang dibatasi ruang.

14. Pilih Create metric filter (Buat filter metrik). Anda dapat menemukan filter metrik yang Anda buat dari panel navigasi. Pilih Log, lalu pilih Grup log. Pilih nama grup log tempat Anda membuat filter metrik, lalu pilih tab Filter metrik.

Contoh: Hitung peristiwa log

Jenis paling sederhana dari pemantauan log acara adalah menghitung jumlah log acara yang terjadi. Anda mungkin ingin melakukan ini untuk menghitung jumlah semua kejadian, untuk membuat monitor gaya "detak jantung" atau hanya untuk berlatih membuat filter metrik.

Dalam contoh CLI berikut, filter metrik yang disebut MyAppAccessCount diterapkan ke grup log MyApp /access.log untuk membuat metrik EventCount di namespace. CloudWatch MyNamespace Filter dikonfigurasi untuk mencocokkan konten log acara dan menambahkan metrik dengan kelipatan sebesar "1".

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih nama grup log.
4. Pilih Actions, Create metric filter (Buat filter metrik).
5. Biarkan Filter Pattern (Pola Filter) dan Select Log Data to Test (Pilih Data Log untuk Pengujian) kosong.
6. Pilih Next (Selanjutnya), lalu untuk Filter Name (Nama Filter), ketik **EventCount**.
7. Di bawah Metric Details (Detail Metrik), untuk Metric Namespace, ketik **MyNameSpace**.
8. Untuk Metric Name (Nama Metrik), ketik **MyAppEventCount**.
9. Konfirmasi bahwa Metric Value (Nilai Metrik) adalah 1. Ini menentukan bahwa jumlah bertambah 1 untuk setiap log acara.
10. Masukkan 0 untuk Default Value (Nilai Default), lalu pilih Next (Selanjutnya). Menentukan nilai default memastikan bahwa data dilaporkan bahkan selama periode ketika tidak ada log acara terjadi sehingga mencegah metrik tidak teratur saat data terkadang tidak ada.
11. Pilih Create metric filter (Buat filter metrik).

Untuk membuat filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs put-metric-filter \  
  --log-group-name MyApp/access.log \  
  --filter-name EventCount \  
  --filter-pattern " " \  
  --metric-transformations \  
    metricName=MyAppEventCount,metricNamespace=MyNameSpace,metricValue=1,defaultValue=0
```

Anda dapat menguji kebijakan baru ini dengan memposting data kejadian apa pun. Anda akan melihat titik data yang dipublikasikan ke metrik MyAppAccessEventCount.

Untuk memposting data acara menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs put-log-events \  

```

```
--log-group-name MyApp/access.log --log-stream-name TestStream1 \  
--log-events \  
  timestamp=1394793518000,message="Test event 1" \  
  timestamp=1394793518000,message="Test event 2" \  
  timestamp=1394793528000,message="This message also contains an Error"
```

Contoh: Hitung kemunculan suatu istilah

Log acara sering mencakup pesan penting yang ingin Anda hitung, mungkin tentang keberhasilan atau kegagalan operasi. Sebagai contoh, kesalahan dapat terjadi dan dicatat ke berkas log jika operasi tertentu gagal. Anda mungkin ingin memantau entri ini untuk memahami tren kesalahan Anda.

Dalam contoh di bawah ini, filter metrik dibuat untuk memantau istilah Error. Kebijakan telah dibuat dan ditambahkan ke grup log MyApp/message.log. CloudWatch Log menerbitkan titik data ke metrik CloudWatch kustom ErrorCount di namespace MyApp/message.log dengan nilai "1" untuk setiap peristiwa yang berisi Kesalahan. Jika tidak ada kejadian berisi kata Error, nilai 0 akan dipublikasikan. Saat membuat grafik data ini di CloudWatch konsol, pastikan untuk menggunakan statistik jumlah.

Setelah membuat filter metrik, Anda dapat melihat metrik di CloudWatch konsol. Saat Anda memilih metrik yang akan ditampilkan, pilih namespace metrik yang cocok dengan nama grup log. Untuk informasi selengkapnya, lihat [Melihat Metrik yang Tersedia](#).

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih nama grup log.
4. Pilih Actions (Tindakan), Create metric filter (Buat filter metrik).
5. Untuk Filter Pattern (Pola Filter), masukkan **Error**.

Note

Semua entri di Filter Pattern (Pola Filter) peka huruf besar-kecil.

6. (Opsional) Untuk menguji pola filter Anda, di Test Pattern (Pola Uji), masukkan satu atau beberapa log acara untuk digunakan menguji pola. Setiap log acara harus dalam satu baris, karena jeda baris yang digunakan untuk memisahkan log acara di kotak pesan log acara (Pesan log acara).

7. Pilih Next (Selanjutnya), lalu di halaman Assign metric (Tetapkan metrik), untuk Filter Name (Nama Filter), ketik **MyAppErrorCount**.
8. Di bawah Metric Details (Detail Metrik), untuk Metric Namespace, ketik MyNamespace.
9. Untuk Metric Name (Nama Metrik), ketik ErrorCount.
10. Konfirmasi bahwa Metric Value (Nilai Metrik) adalah 1. Ini menentukan bahwa jumlah bertambah 1 untuk setiap log acara yang berisi "Error".
11. Untuk Default Value (Nilai Default) ketik 0, lalu pilih Next (Selanjutnya).
12. Pilih Create metric filter (Buat filter metrik).

Untuk membuat filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs put-metric-filter \  
  --log-group-name MyApp/message.log \  
  --filter-name MyAppErrorCount \  
  --filter-pattern 'Error' \  
  --metric-transformations \  
    metricName=ErrorCount,metricNamespace=MyNamespace,metricValue=1,defaultValue=0
```

Anda dapat menguji kebijakan baru ini dengan memposting kejadian yang berisi kata "Error" dalam pesannya.

Untuk memposting acara menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut. Perhatikan bahwa pola peka huruf besar dan kecil.

```
aws logs put-log-events \  
  --log-group-name MyApp/access.log --log-stream-name TestStream1 \  
  --log-events \  
    timestamp=1394793518000,message="This message contains an Error" \  
    timestamp=1394793528000,message="This message also contains an Error"
```

Contoh: Hitung kode HTTP 404

Menggunakan CloudWatch Log, Anda dapat memantau berapa kali server Apache Anda mengembalikan respons HTTP 404, yang merupakan kode respons untuk halaman yang tidak ditemukan. Anda mungkin ingin memantau ini untuk memahami seberapa sering pengunjung situs

Anda tidak menemukan sumber daya yang mereka cari. Asumsikan bahwa struktur catatan log Anda menyertakan informasi berikut untuk setiap log acara (kunjungan situs):

- Alamat IP Peminta
- Identitas RFC 1413
- Nama pengguna
- Stempel waktu
- Metode permintaan dengan protokol dan sumber daya yang diminta
- Kode respons HTTP terhadap permintaan
- Byte yang ditransfer dalam permintaan

Contohnya dapat terlihat seperti berikut:

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 404 2326
```

Anda dapat menentukan aturan yang mencoba untuk mencocokkan kejadian dengan struktur seperti itu untuk kesalahan HTTP 404, seperti yang ditunjukkan dalam contoh berikut:

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih Actions, Create metric filter (Buat filter metrik).
4. Untuk Filter Pattern (Pola Filter), ketik **[IP, UserInfo, User, Timestamp, RequestInfo, StatusCode=404, Bytes]**.
5. (Opsional) Untuk menguji pola filter Anda, di Test Pattern (Pola Uji), masukkan satu atau beberapa log acara untuk digunakan menguji pola. Setiap log acara harus dalam satu baris, karena jeda baris yang digunakan untuk memisahkan log acara di kotak pesan log acara (Pesan log acara).
6. Pilih Berikutnya, dan kemudian untuk Nama Filter, ketik HTTP404Kesalahan.
7. Di bawah Metric Details (Detail Metrik), untuk Metric Namespace (Namespace Metrik), masukkan **MyNameSpace**.
8. Untuk Metric Name (Nama Metrik), masukkan **ApacheNotFoundErrorCount**.

9. Konfirmasi bahwa Metric Value (Nilai Metrik) adalah 1. Ini menentukan bahwa jumlah bertambah 1 untuk setiap kejadian 404 Error.
10. Masukkan 0 untuk Default Value (Nilai Default), lalu pilih Next (Selanjutnya).
11. Pilih Create metric filter (Buat filter metrik).

Untuk membuat filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs put-metric-filter \  
  --log-group-name MyApp/access.log \  
  --filter-name HTTP404Errors \  
  --filter-pattern '[ip, id, user, timestamp, request, status_code=404, size]' \  
  --metric-transformations \  
    metricName=ApacheNotFoundErrorCode,metricNamespace=MyNamespace,metricValue=1
```

Dalam contoh ini, digunakan karakter literal, seperti tanda kurung siku kiri dan kanan, tanda kutip ganda, dan string karakter 404. Pola harus cocok dengan seluruh pesan log acara agar log acara dipertimbangkan untuk pemantauan.

Anda dapat memverifikasi pembuatan filter metrik dengan menggunakan perintah `describe-metric-filters`. Anda akan melihat output seperti ini:

```
aws logs describe-metric-filters --log-group-name MyApp/access.log  
  
{  
  "metricFilters": [  
    {  
      "filterName": "HTTP404Errors",  
      "metricTransformations": [  
        {  
          "metricValue": "1",  
          "metricNamespace": "MyNamespace",  
          "metricName": "ApacheNotFoundErrorCode"  
        }  
      ],  
      "creationTime": 1399277571078,  
      "filterPattern": "[ip, id, user, timestamp, request, status_code=404,  
size]"  
    }  
  ]  
}
```



```
}
```

Sekarang Anda dapat memposting beberapa kejadian secara manual:

```
aws logs put-log-events \  
--log-group-name MyApp/access.log --log-stream-name hostname \  
--log-events \  
timestamp=1394793518000,message="127.0.0.1 - bob [10/Oct/2000:13:55:36 -0700] \"GET /  
apache_pb.gif HTTP/1.0\" 404 2326" \  
timestamp=1394793528000,message="127.0.0.1 - bob [10/Oct/2000:13:55:36 -0700] \"GET /  
apache_pb2.gif HTTP/1.0\" 200 2326"
```

Segera setelah meletakkan contoh peristiwa log ini, Anda dapat mengambil metrik yang dinamai di CloudWatch konsol sebagai `ApacheNotFoundErrorCount`.

Contoh: Hitung kode HTTP 4xx

Seperti dalam contoh sebelumnya, Anda mungkin ingin memantau log akses layanan web Anda dan memantau tingkat kode respons HTTP. Misalnya, Anda mungkin ingin memantau semua kesalahan HTTP di tingkat 400. Namun, Anda mungkin tidak ingin menentukan filter metrik baru untuk setiap kode yang dihasilkan.

Contoh berikut menunjukkan cara membuat metrik yang mencakup semua respons kode HTTP di tingkat 400 dari log akses menggunakan format log akses Apache dari contoh [Contoh: Hitung kode HTTP 404](#).

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih nama grup log untuk server Apache.
4. Pilih Actions, Create metric filter (Buat filter metrik).
5. Untuk Filter pattern (Pola filter), masukkan **[ip, id, user, timestamp, request, status_code=4*, size]**.
6. (Opsional) Untuk menguji pola filter Anda, di Test Pattern (Pola Uji), masukkan satu atau beberapa log acara untuk digunakan menguji pola. Setiap log acara harus dalam satu baris, karena jeda baris yang digunakan untuk memisahkan log acara di kotak pesan log acara (Pesan log acara).

7. Pilih Next (Selanjutnya), lalu untuk Filter name (Nama filter), ketik **HTTP4xxErrors**.
8. Di bawah Metric details (Detail metrik), untuk Metric namespace (Namespace metrik), masukkan **MyNameSpace**.
9. Untuk nama Metrik, masukkan HTTP4xxKesalahan.
10. Untuk Metric value (Nilai metrik), masukkan 1. Ini menentukan bahwa jumlah bertambah 1 untuk setiap log acara yang berisi kesalahan 4xx.
11. Masukkan 0 untuk Default value (Nilai default), lalu pilih Next (Selanjutnya).
12. Pilih Create metric filter (Buat filter metrik).

Untuk membuat filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs put-metric-filter \  
  --log-group-name MyApp/access.log \  
  --filter-name HTTP4xxErrors \  
  --filter-pattern '[ip, id, user, timestamp, request, status_code=4*, size]' \  
  --metric-transformations \  
    metricName=HTTP4xxErrors,metricNamespace=MyNameSpace,metricValue=1,defaultValue=0
```

Anda dapat menggunakan data berikut dalam panggilan put-event untuk menguji aturan ini. Jika Anda tidak menghapus aturan pemantauan di contoh sebelumnya, Anda akan menghasilkan dua metrik yang berbeda.

```
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287  
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287  
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /~test/ HTTP/1.1" 200 3  
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308  
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308  
127.0.0.1 - - [24/Sep/2013:11:51:34 -0700] "GET /~test/index.html HTTP/1.1" 200 3
```

Contoh: Mengekstraksi bidang dari log Apache dan menetapkan dimensi

Kadang-kadang, alih-alih menghitung, akan lebih berguna jika Anda menggunakan nilai dalam log acara individual untuk nilai metrik. Contoh ini menunjukkan cara Anda dapat membuat aturan ekstraksi untuk membuat metrik yang mengukur byte yang ditransfer oleh server web Apache.

Contoh ini juga menunjukkan cara menetapkan dimensi ke metrik yang Anda buat.

Untuk membuat filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih nama grup log untuk server Apache.
4. Pilih Actions, Create metric filter (Buat filter metrik).
5. Untuk Filter pattern (Pola filter), masukkan **[ip, id, user, timestamp, request, status_code, size]**.
6. (Opsional) Untuk menguji pola filter Anda, di Test Pattern (Pola Uji), masukkan satu atau beberapa log acara untuk digunakan menguji pola. Setiap log acara harus dalam satu baris, karena jeda baris yang digunakan untuk memisahkan log acara di kotak pesan log acara (Pesan log acara).
7. Pilih Next (Selanjutnya), lalu untuk Filter name (Nama filter), ketik **size**.
8. Di bawah Metric details (Detail metrik), untuk Metric namespace (Namespace metrik), masukkan **MyNameSpace**. Karena ini adalah namespace baru, pastikan bahwa Create new (Buat baru) dipilih.
9. Untuk Metric name (Nama metrik), masukkan **BytesTransferred**
10. Untuk Metric value (Nilai metrik), masukkan **\$size**.
11. Untuk Unit, pilih Byte.
12. Untuk Dimension Name (Nama Dimensi), ketik **IP**.
13. Untuk Dimension Value (Nilai Dimensi), ketik **\$ip**, lalu pilih Next (Selanjutnya).
14. Pilih Create metric filter (Buat filter metrik).

Untuk membuat filter metrik ini menggunakan AWS CLI

Di jendela perintah, jalankan perintah berikut

```
aws logs put-metric-filter \  
--log-group-name MyApp/access.log \  
--filter-name BytesTransferred \  
--filter-pattern '[ip, id, user, timestamp, request, status_code, size]' \  
--metric-transformations \  
metricName=BytesTransferred,metricNamespace=MyNameSpace,metricValue='$size'
```

```
aws logs put-metric-filter \  

```

```
--log-group-name MyApp/access.log \
--filter-name BytesTransferred \
--filter-pattern '[ip, id, user, timestamp, request, status_code, size]' \
--metric-transformations \
metricName=BytesTransferred,metricNamespace=MyNamespace,metricValue='$size',unit=Bytes,dimension1=$ip}}'
```

Note

Dalam perintah ini, gunakan format ini untuk menentukan beberapa dimensi.

```
aws logs put-metric-filter \
--log-group-name my-log-group-name \
--filter-name my-filter-name \
--filter-pattern 'my-filter-pattern' \
--metric-transformations \
metricName=my-metric-name,metricNamespace=my-metric-namespace,metricValue=my-token,unit=unit,dimensions='{dimension1=$dim,dimension2=$dim2,dim3=$dim3}'
```

Anda dapat menggunakan data berikut dalam put-log-event panggilan untuk menguji aturan ini. Ini akan menghasilkan dua metrik yang berbeda jika Anda tidak menghapus aturan pemantauan dalam contoh sebelumnya.

```
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:49:52 -0700] "GET /index.html HTTP/1.1" 404 287
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /~test/ HTTP/1.1" 200 3
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:50:51 -0700] "GET /favicon.ico HTTP/1.1" 404 308
127.0.0.1 - - [24/Sep/2013:11:51:34 -0700] "GET /~test/index.html HTTP/1.1" 200 3
```

Daftar filter metrik

Anda dapat membuat daftar semua filter metrik dalam grup log.

Untuk membuat daftar filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.

3. Di panel konten, di daftar grup log, di kolom Metric Filters (Filter Metrik), pilih jumlah filter.

Layar Log Groups > Filters for (Grup Log > Filter untuk) mencantumkan semua filter metrik yang terkait dengan grup log.

Untuk membuat daftar filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs describe-metric-filters --log-group-name MyApp/access.log
```

Berikut ini adalah output contoh:

```
{
  "metricFilters": [
    {
      "filterName": "HTTP404Errors",
      "metricTransformations": [
        {
          "metricValue": "1",
          "metricNamespace": "MyNamespace",
          "metricName": "ApacheNotFoundErrorCode"
        }
      ],
      "creationTime": 1399277571078,
      "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
size]"
    }
  ]
}
```

Menghapus filter metrik

Kebijakan diidentifikasi berdasarkan nama dan grup lognya.

Untuk menghapus filter metrik menggunakan CloudWatch konsol

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Di panel konten, di kolom Metric Filter (Filter Metrik), pilih jumlah filter metrik untuk grup log.

4. Di layar Metric Filters (Filter Metrik), centang kotak di sebelah kanan nama filter yang ingin Anda hapus. Lalu pilih Hapus.
5. Saat diminta konfirmasi, pilih Hapus.

Untuk menghapus filter metrik menggunakan AWS CLI

Pada jendela perintah, jalankan perintah berikut:

```
aws logs delete-metric-filter --log-group-name MyApp/access.log \  
--filter-name MyFilterName
```

Pemrosesan data log secara real-time dengan langganan

Anda dapat menggunakan langganan untuk mendapatkan akses ke umpan real-time peristiwa CloudWatch log dari Log dan mengirimkannya ke layanan lain seperti aliran Amazon Kinesis, aliran Amazon Data Firehose, AWS Lambda atau untuk pemrosesan, analisis, atau pemuatan kustom ke sistem lain. Ketika peristiwa log dikirim ke layanan penerima, mereka dikodekan base64 dan dikompresi dengan format gzip.

Anda juga dapat menggunakan sentralisasi CloudWatch Log untuk mereplikasi data log dari beberapa akun dan wilayah ke lokasi pusat. Untuk informasi selengkapnya, lihat [Sentralisasi log lintas wilayah lintas akun](#).

Untuk mulai berlangganan peristiwa log, buat sumber daya penerima, seperti aliran Amazon Kinesis Data Streams, tempat acara akan dikirimkan. Filter langganan mendefinisikan pola filter yang akan digunakan untuk memfilter peristiwa log mana yang dikirim ke AWS sumber daya Anda, serta informasi tentang ke mana harus mengirim peristiwa log yang cocok. Peristiwa log dikirim ke sumber daya penerima segera setelah dicerna, biasanya kurang dari tiga menit.

Note

Jika grup log dengan langganan menggunakan transformasi log, pola filter dibandingkan dengan versi peristiwa log yang diubah. Untuk informasi selengkapnya, lihat [Ubah log selama konsumsi](#).

Anda dapat membuat langganan di tingkat akun dan di tingkat grup log. Setiap akun dapat memiliki satu filter langganan tingkat akun per Wilayah. Setiap grup log dapat memiliki hingga dua filter langganan yang terkait dengan grup.

Note

Jika layanan tujuan mengembalikan kesalahan yang dapat dicoba ulang seperti pengecualian pembatasan atau pengecualian layanan yang dapat dicoba ulang (misalnya HTTP 5xx), CloudWatch Log terus mencoba lagi pengiriman hingga 24 jam. CloudWatch Log tidak mencoba mengirimkan kembali jika kesalahannya adalah kesalahan yang tidak dapat dicoba ulang, seperti `AccessDeniedException` `ResourceNotFoundException`. Dalam kasus ini,

filter langganan dinonaktifkan hingga 10 menit, dan kemudian CloudWatch Log mencoba mengirim log ke tujuan. Selama periode dinonaktifkan ini, log dilewati.

CloudWatch Log juga menghasilkan CloudWatch metrik tentang penerusan peristiwa log ke langganan. Untuk informasi selengkapnya, lihat [Pemantauan dengan CloudWatch metrik](#).

Anda juga dapat menggunakan langganan CloudWatch Log untuk mengalirkan data log dalam waktu dekat ke kluster OpenSearch Layanan Amazon. Untuk informasi selengkapnya, lihat [data Streaming CloudWatch Log ke OpenSearch Layanan Amazon](#).

Langganan hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

 Note

Filter langganan dapat mengumpulkan peristiwa log untuk mengoptimalkan transmisi dan mengurangi jumlah panggilan yang dilakukan ke tujuan. Batching tidak dijamin tetapi digunakan bila memungkinkan.

Untuk pemrosesan batch dan analisis data log sesuai jadwal, pertimbangkan untuk menggunakan [Mengotomatiskan analisis log dengan kueri terjadwal](#). Kueri terjadwal menjalankan kueri CloudWatch Logs Insights secara otomatis dan memberikan hasil ke tujuan seperti bucket Amazon S3 atau bus acara Amazon. EventBridge

 Note

Filter langganan memastikan setidaknya satu kali pengiriman acara, sementara kejadian duplikat kadang-kadang dapat terjadi.

Daftar Isi

- [Konsep](#)
- [Filter langganan tingkat grup log](#)
- [Filter berlangganan tingkat akun](#)
- [Langganan lintas akun Lintas wilayah](#)

- [Pencegahan Deputi Bingung](#)
- [Pencegahan rekursi log](#)

Konsep

Setiap filter langganan terdiri dari elemen kunci berikut:

pola filter

Deskripsi simbolis tentang bagaimana CloudWatch Log harus menafsirkan data di setiap peristiwa log, bersama dengan ekspresi pemfilteran yang membatasi apa yang dikirim ke sumber daya tujuan. AWS Untuk informasi selengkapnya tentang sintaks pola filter, lihat [Filter sintaks pola untuk filter metrik, filter langganan, peristiwa log filter, dan Live Tail](#).

arn tujuan

Nama Sumber Daya Amazon (ARN) dari aliran Data Amazon Kinesis Streams, aliran Firehose, atau fungsi Lambda yang ingin Anda gunakan sebagai tujuan feed langganan.

arn peran

Peran IAM yang memberikan CloudWatch Log izin yang diperlukan untuk memasukkan data ke tujuan yang dipilih. Peran ini tidak diperlukan untuk tujuan Lambda karena CloudWatch Log bisa mendapatkan izin yang diperlukan dari pengaturan kontrol akses pada fungsi Lambda itu sendiri.

distribusi

Metode yang digunakan untuk mendistribusikan data log ke tujuan, ketika tujuan adalah aliran di Amazon Kinesis Data Streams. Secara default, data log dikelompokkan berdasarkan pengaliran log. Untuk distribusi yang lebih merata, Anda dapat mengelompokkan data log secara acak.

Untuk langganan tingkat grup log, elemen kunci berikut juga disertakan:

nama grup log

Grup log yang dikaitkan dengan filter langganan. Semua log acara yang diunggah ke grup log ini akan dikenakan filter langganan, dan log acara yang cocok dengan filter akan dikirim ke layanan tujuan yang menerima log acara yang cocok.

Untuk langganan tingkat akun, elemen kunci berikut juga disertakan:

kriteria seleksi

Kriteria yang digunakan untuk memilih grup log mana yang menerapkan filter langganan tingkat akun. Jika Anda tidak menentukan ini, filter langganan tingkat akun diterapkan ke semua grup log di akun. Bidang ini digunakan untuk mencegah loop log tak terbatas. Untuk informasi selengkapnya tentang masalah loop log tak terbatas, lihat [Pencegahan rekursi log](#).

Kriteria seleksi memiliki batas ukuran 25 KB.

Untuk grup log terpusat, elemen kunci berikut juga disertakan. Elemen-elemen ini dapat digunakan sebagai kriteria pemilihan lapangan untuk membantu mengidentifikasi sumber data log, memungkinkan penyaringan dan analisis metrik yang lebih terperinci yang berasal dari log terpusat.

`@aws .akun`

Bidang ini mengidentifikasi ID AWS akun dari mana peristiwa log berasal.

`@aws .wilayah`

Bidang ini mengidentifikasi AWS Wilayah tempat peristiwa log dihasilkan.

Filter langganan tingkat grup log

Anda dapat menggunakan filter langganan dengan Amazon Kinesis Data AWS Lambda Streams, Amazon Data Firehose, atau Amazon Service. OpenSearch Log yang dikirim ke layanan melalui filter langganan dikodekan base64 dan dikompresi dengan format gzip. Jika Anda menggunakan log terpusat dengan Anda AWS Organizations, Anda dapat memilih untuk memancarkan bidang `@aws .account` dan `@aws .region` sistem untuk mengidentifikasi data mana yang berasal dari akun dan wilayah mana di organisasi Anda. Bagian ini memberikan contoh yang dapat Anda ikuti untuk membuat filter langganan CloudWatch Log yang mengirimkan data log ke Firehose, Lambda, Amazon Kinesis Data Streams, dan Layanan. OpenSearch

Note

Jika Anda ingin mencari data log Anda, lihat [Filter dan sintaks pola](#).

Contoh

- [Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams](#)
- [Contoh 2: Filter berlangganan dengan AWS Lambda](#)
- [Contoh 3: Filter berlangganan dengan Amazon Data Firehose](#)
- [Contoh 4: Filter berlangganan dengan OpenSearch Layanan Amazon](#)

Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams

Contoh berikut mengaitkan filter langganan dengan grup log yang berisi AWS CloudTrail peristiwa. Filter langganan mengirimkan setiap aktivitas yang dicatat yang dibuat oleh AWS kredensi “Root” ke aliran di Amazon Kinesis Data Streams yang disebut “.” RootAccess Untuk informasi selengkapnya tentang cara mengirim AWS CloudTrail peristiwa ke CloudWatch Log, lihat [Mengirim CloudTrail Acara ke CloudWatch Log](#) di Panduan AWS CloudTrail Pengguna.

Note

Sebelum Anda membuat aliran, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat aliran dengan pecahan yang cukup untuk menangani volume ini. Jika pengaliran tidak memiliki serpihan yang cukup, pengaliran log akan mengalami throttling. Untuk informasi selengkapnya tentang batas volume streaming, lihat [Kuota dan Batas](#). Kiriman yang dibatasi dicoba ulang hingga 24 jam. Setelah 24 jam, kiriman yang gagal dijatuhkan.

Untuk mengurangi risiko pelambatan, Anda dapat mengambil langkah-langkah berikut:

- Tentukan random distribution kapan Anda membuat filter langganan dengan [PutSubscriptionFilter](#) atau [put-subscription-filter](#). Secara default, distribusi filter aliran adalah dengan aliran log dan ini dapat menyebabkan pelambatan.
- Pantau streaming Anda menggunakan CloudWatch metrik. Ini membantu Anda mengidentifikasi pelambatan apa pun dan menyesuaikan konfigurasi Anda sesuai dengan itu. Misalnya, `DeliveryThrottling` metrik dapat digunakan untuk melacak jumlah peristiwa CloudWatch log yang Log dibatasi saat meneruskan data ke tujuan langganan. Untuk informasi selengkapnya tentang pemantauan, lihat [Pemantauan dengan CloudWatch metrik](#).
- Gunakan mode kapasitas sesuai permintaan untuk streaming Anda di Amazon Kinesis Data Streams. Mode sesuai permintaan langsung mengakomodasi beban kerja Anda saat naik atau turun. Informasi selengkapnya tentang mode kapasitas sesuai permintaan, lihat Mode [sesuai permintaan](#).

- Batasi pola filter CloudWatch langganan agar sesuai dengan kapasitas streaming Anda di Amazon Kinesis Data Streams. Jika Anda mengirim terlalu banyak data ke aliran, Anda mungkin perlu mengurangi ukuran filter atau menyesuaikan kriteria filter.

Untuk membuat filter langganan untuk Amazon Kinesis Data Streams

1. Buat aliran tujuan menggunakan perintah berikut:

```
$ C:\> aws kinesis create-stream --stream-name "RootAccess" --shard-count 1
```

2. Tunggu hingga aliran menjadi Aktif (ini mungkin memakan waktu satu atau dua menit). Anda dapat menggunakan perintah Amazon Kinesis [Data Streams describe-stream](#) berikut untuk memeriksa `StreamDescription` `StreamStatus` properti. Selain itu, perhatikan `StreamDescription` nilai `streamArn`, karena Anda akan membutuhkannya di langkah selanjutnya:

```
aws kinesis describe-stream --stream-name "RootAccess"
```

Berikut ini adalah output contoh:

```
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "RootAccess",
    "StreamARN": "arn:aws:kinesis:us-east-1:123456789012:stream/RootAccess",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "340282366920938463463374607431768211455",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
            "49551135218688818456679503831981458784591352702181572610"
        }
      }
    ]
  }
}
```

3. Buat peran IAM yang akan memberikan izin CloudWatch Log untuk memasukkan data ke aliran Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file (misalnya, `~/TrustPolicyForCWL-Kinesis.json`). Gunakan editor teks untuk membuat kebijakan ini. Jangan gunakan konsol IAM untuk membuatnya.

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global untuk membantu mencegah masalah keamanan wakil yang membingungkan. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
    }
  }
}
```

4. Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dihasilkan, karena Anda juga akan membutuhkannya untuk langkah selanjutnya:

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
file://~/TrustPolicyForCWL-Kinesis.json
```

Berikut adalah contoh output.

```
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.amazonaws.com"
        },
        "Condition": {
          "StringLike": {
            "aws:SourceArn": { "arn:aws:logs:region:123456789012:*" }
          }
        }
      }
    }
  }
}
```

```

    }
  }
},
"RoleId": "AA0IIAH450GAB4HC5F431",
"CreateDate": "2015-05-29T13:46:29.431Z",
"RoleName": "CWLtoKinesisRole",
"Path": "/",
"Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
}
}

```

5. Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, Anda akan membuat kebijakan izin dalam file (misalnya, ~/PermissionsForCWL-Kinesis.json). Gunakan editor teks untuk membuat kebijakan ini. Jangan gunakan konsol IAM untuk membuatnya.

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/RootAccess"
    }
  ]
}

```

6. Kaitkan kebijakan izin dengan peran menggunakan [put-role-policy](#) perintah berikut:

```
aws iam put-role-policy --role-name CWLtoKinesisRole --policy-name Permissions-
Policy-For-CWL --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. Setelah streaming dalam status Aktif dan Anda telah membuat peran IAM, Anda dapat membuat filter langganan CloudWatch Log. Filter langganan segera memulai aliran data log waktu nyata dari grup log yang dipilih ke aliran Anda:

```
aws logs put-subscription-filter \
  --log-group-name "CloudTrail/logs" \
  --filter-name "RootAccess" \
  --filter-pattern "{$.userIdentity.type = Root}" \
  --destination-arn "arn:aws:kinesis:region:123456789012:stream/RootAccess" \
  --role-arn "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
```

8. Setelah Anda mengatur filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter ke aliran Anda. Anda dapat memverifikasi bahwa ini terjadi dengan mengambil iterator pecahan Amazon Kinesis Data Streams dan menggunakan perintah Amazon Kinesis Data Streams `get-records` untuk mengambil beberapa rekaman Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator --stream-name RootAccess --shard-id
shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
  "ShardIterator":
  "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+0IK8zM5My8ID
+g6rMo7UKWeI4+IWiK20Sh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+0IK8zM5My8ID
+g6rMo7UKWeI4+IWiK20Sh0uP"
```

Perhatikan bahwa Anda mungkin perlu melakukan panggilan ini beberapa kali sebelum Amazon Kinesis Data Streams mulai mengembalikan data.

Anda akan melihat respons dengan array catatan. Atribut Data dalam catatan Amazon Kinesis Data Streams dikodekan base64 dan dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "owner": "111111111111",
  "logGroup": "CloudTrail/logs",
  "logStream": "111111111111_CloudTrail/logs_us-east-1",
```

```

    "subscriptionFilters": [
      "Destination"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
      {
        "id": "31953106606966983378809025079804211143289615424298221568",
        "timestamp": 1432826855000,
        "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
      },
      {
        "id": "31953106606966983378809025079804211143289615424298221569",
        "timestamp": 1432826855000,
        "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
      },
      {
        "id": "31953106606966983378809025079804211143289615424298221570",
        "timestamp": 1432826855000,
        "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
      }
    ]
  }

```

Elemen kunci dalam struktur data di atas adalah sebagai berikut:

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

messageType

Pesan data akan menggunakan tipe "DATA_MESSAGE". Terkadang CloudWatch Log dapat memancarkan catatan Amazon Kinesis Data Streams dengan tipe "CONTROL_MESSAGE", terutama untuk memeriksa apakah tujuan dapat dijangkau.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti "id" adalah pengenal unik untuk setiap log acara.

Contoh 2: Filter berlangganan dengan AWS Lambda

Dalam contoh ini, Anda akan membuat filter langganan CloudWatch Log yang mengirimkan data log ke AWS Lambda fungsi Anda.

Note

Sebelum membuat fungsi Lambda, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat fungsi yang dapat menangani volume ini. Jika fungsi tidak memiliki volume yang cukup, pengaliran log akan mengalami throttling. Untuk informasi selengkapnya tentang batas Lambda, lihat [Batas AWS Lambda](#).

Untuk membuat filter langganan untuk Lambda

1. Buat AWS Lambda fungsinya.

Pastikan bahwa Anda telah mengatur peran eksekusi Lambda. Untuk informasi selengkapnya, lihat: [Langkah 2.2: Buat IAM role \(peran eksekusi\)](#) dalam Panduan Developer AWS Lambda .

2. Buka editor teks dan buat file bernama `helloWorld.js` dengan isi sebagai berikut:

```
var zlib = require('zlib');
exports.handler = function(input, context) {
  var payload = Buffer.from(input.awslogs.data, 'base64');
  zlib.gunzip(payload, function(e, result) {
    if (e) {
      context.fail(e);
    } else {
      result = JSON.parse(result.toString());
    }
  });
}
```

```

        console.log("Event Data:", JSON.stringify(result, null, 2));
        context.succeed();
    }
});
};

```

3. Buat zip file helloWorld.js dan simpan dengan nama helloWorld.zip.
4. Gunakan perintah berikut, di mana perannya adalah peran eksekusi Lambda yang Anda atur di langkah pertama:

```

aws lambda create-function \
  --function-name helloworld \
  --zip-file fileb://file-path/helloWorld.zip \
  --role lambda-execution-role-arn \
  --handler helloWorld.handler \
  --runtime nodejs12.x

```

5. Berikan CloudWatch Log izin untuk menjalankan fungsi Anda. Gunakan perintah berikut, dengan mengganti akun placeholder dengan akun Anda sendiri dan grup log placeholder dengan grup log yang akan diproses:

```

aws lambda add-permission \
  --function-name "helloworld" \
  --statement-id "helloworld" \
  --principal "logs.amazonaws.com" \
  --action "lambda:InvokeFunction" \
  --source-arn "arn:aws:logs:region:123456789123:log-group:TestLambda:*" \
  --source-account "123456789012"

```

6. Buat filter langganan menggunakan perintah berikut, dengan mengganti akun placeholder dengan akun Anda sendiri dan grup log placeholder dengan grup log yang akan diproses:

```

aws logs put-subscription-filter \
  --log-group-name myLogGroup \
  --filter-name demo \
  --filter-pattern "" \
  --destination-arn arn:aws:lambda:region:123456789123:function:helloworld

```

7. (Opsional) Uji menggunakan contoh log acara. Di jendela perintah, jalankan perintah berikut, yang akan menempatkan pesan log sederhana ke dalam pengaliran langganan.

Untuk melihat output dari fungsi Lambda Anda, navigasikan ke fungsi Lambda di mana Anda akan melihat output di: `aws/lambda/helloworld`

```
aws logs put-log-events --log-group-name myLogGroup --log-stream-name stream1 --
log-events "[{"timestamp\":"<CURRENT_TIMESTAMP_MILLIS> , \"message\": \"Simple
Lambda Test\"}]"]
```

Anda akan melihat respons dengan array Lambda. Atribut Data dalam catatan Lambda adalah base64 dikodekan dan dikompresi dengan format gzip. Muatan sebenarnya yang diterima oleh Lambda memiliki format berikut `{ "awslogs": { "data": "BASE64ENCODED_GZIP_COMPRESSED_DATA" } }` Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "owner": "123456789012",
  "logGroup": "CloudTrail",
  "logStream": "123456789012_CloudTrail_us-east-1",
  "subscriptionFilters": [
    "Destination"
  ],
  "messageType": "DATA_MESSAGE",
  "logEvents": [
    {
      "id": "31953106606966983378809025079804211143289615424298221568",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\", \"userIdentity\": {\"type\": \"Root\"}}",
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221569",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\", \"userIdentity\": {\"type\": \"Root\"}}",
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221570",
```

```

      "timestamp": 1432826855000,
      "message": "{\\"eventVersion\\":\\"1.03\\",\\"userIdentity\\":{\\"type\\":
\\"Root\\"}"
    }
  ]
}

```

Elemen kunci dalam struktur data di atas adalah sebagai berikut:

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

messageType

Pesan data akan menggunakan tipe "DATA_MESSAGE". Terkadang CloudWatch Log dapat memancarkan catatan Lambda dengan tipe "CONTROL_MESSAGE", terutama untuk memeriksa apakah tujuan dapat dijangkau.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti "id" adalah pengenalan unik untuk setiap log acara.

Contoh 3: Filter berlangganan dengan Amazon Data Firehose

Dalam contoh ini, Anda akan membuat langganan CloudWatch Log yang mengirimkan peristiwa log masuk yang cocok dengan filter yang Anda tentukan ke aliran pengiriman Amazon Data Firehose Anda. Data yang dikirim dari CloudWatch Log ke Amazon Data Firehose sudah dikompresi dengan kompresi gzip level 6, jadi Anda tidak perlu menggunakan kompresi dalam aliran pengiriman Firehose

Anda. Anda kemudian dapat menggunakan fitur dekompresi di Firehose untuk mendekompresi log secara otomatis. Untuk informasi selengkapnya, lihat [Mengirim CloudWatch Log ke Firehose](#).

Note

Sebelum Anda membuat aliran Firehose, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat aliran Firehose yang dapat menangani volume ini. Jika pengaliran tidak dapat menangani volume, pengaliran log akan mengalami throttling. Untuk informasi selengkapnya tentang batas volume aliran Firehose, lihat Batas Data [Firehose Amazon](#) Data.

Untuk membuat filter langganan untuk Firehose

1. Buat bucket Amazon Simple Storage Service (Amazon S3). Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang sudah ada, lewati ke langkah 2.

Jalankan perintah berikut, dengan mengganti Wilayah placeholder dengan Wilayah yang ingin Anda gunakan:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration
LocationConstraint=region
```

Berikut ini adalah output contoh:

```
{
  "Location": "/amzn-s3-demo-bucket2"
}
```

2. Buat peran IAM yang memberikan izin Amazon Data Firehose untuk memasukkan data ke dalam bucket Amazon S3 Anda.

Untuk informasi selengkapnya, lihat [Mengontrol Akses dengan Amazon Data Firehose di Panduan Pengembang](#) Amazon Data Firehose.

Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForFirehose.json` sebagai berikut:

```
{
  "Statement": {
```

```

    "Effect": "Allow",
    "Principal": { "Service": "firehose.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}

```

- Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dihasilkan, karena Anda akan membutuhkannya dalam langkah selanjutnya:

```

aws iam create-role \
  --role-name FirehoseToS3Role \
  --assume-role-policy-document file://~/TrustPolicyForFirehose.json

{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "firehose.amazonaws.com"
        }
      }
    },
    "RoleId": "AA0IIAH450GAB4HC5F431",
    "CreateDate": "2015-05-29T13:46:29.431Z",
    "RoleName": "FirehoseToS3Role",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:role/FirehoseToS3Role"
  }
}

```

- Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan Firehose di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file `~/PermissionsForFirehose.json`:

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:AbortMultipartUpload",

```

```

        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:PutObject" ],
    "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket2",
        "arn:aws:s3:::amzn-s3-demo-bucket2/*" ]
    }
  ]
}

```

5. Kaitkan kebijakan izin dengan peran menggunakan put-role-policy perintah berikut:

```

aws iam put-role-policy --role-name FirehoseToS3Role --policy-name Permissions-Policy-For-Firehose --policy-document file:///~/PermissionsForFirehose.json

```

6. Buat aliran pengiriman Firehose tujuan sebagai berikut, ganti nilai placeholder untuk RoleArn dan BucketArn dengan peran dan bucket yang Anda buat: ARNs

```

aws firehose create-delivery-stream \
  --delivery-stream-name 'my-delivery-stream' \
  --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::123456789012:role/FirehoseToS3Role", "BucketARN":  

  "arn:aws:s3:::amzn-s3-demo-bucket2"}'

```

Perhatikan bahwa Firehose secara otomatis menggunakan awalan dalam format waktu YYYY/MM/DD/HH UTC untuk objek Amazon S3 yang dikirimkan. Anda dapat menentukan prefiks tambahan untuk ditambahkan di depan prefiks format waktu. Jika prefiks berakhir dengan garis miring (/), itu akan muncul sebagai folder dalam bucket Amazon S3.

7. Tunggu sampai pengaliran menjadi aktif (ini mungkin memakan waktu beberapa menit). Anda dapat menggunakan describe-delivery-stream perintah Firehose untuk memeriksa. DeliveryStreamDescription DeliveryStreamStatus properti. Selain itu, perhatikan DeliveryStreamDescription. DeliveryStreamNilai ARN, karena Anda akan membutuhkannya di langkah selanjutnya:

```

aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
{
  "DeliveryStreamDescription": {
    "HasMoreDestinations": false,

```

```

    "VersionId": "1",
    "CreateTimestamp": 1446075815.822,
    "DeliveryStreamARN": "arn:aws:firehose:us-
east-1:123456789012:deliverystream/my-delivery-stream",
    "DeliveryStreamStatus": "ACTIVE",
    "DeliveryStreamName": "my-delivery-stream",
    "Destinations": [
      {
        "DestinationId": "destinationId-0000000000001",
        "S3DestinationDescription": {
          "CompressionFormat": "UNCOMPRESSED",
          "EncryptionConfiguration": {
            "NoEncryptionConfig": "NoEncryption"
          },
          "RoleARN": "delivery-stream-role",
          "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket2",
          "BufferingHints": {
            "IntervalInSeconds": 300,
            "SizeInMBs": 5
          }
        }
      }
    ]
  }
}

```

8. Buat peran IAM yang memberikan izin CloudWatch Log untuk memasukkan data ke aliran pengiriman Firehose Anda. Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForCWL.json`:

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global untuk membantu mencegah masalah keamanan wakil yang membingungkan. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```

{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringLike": {
        "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
      }
    }
  }
}

```



```

    }
  }
}

```

9. Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dihasilkan, karena Anda akan membutuhkannya dalam langkah selanjutnya:

```

aws iam create-role \
--role-name CWLtoKinesisFirehoseRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.amazonaws.com"
        },
        "Condition": {
          "StringLike": {
            "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
          }
        }
      }
    },
    "RoleId": "AA0IIAH450GAB4HC5F431",
    "CreateDate": "2015-05-29T13:46:29.431Z",
    "RoleName": "CWLtoKinesisFirehoseRole",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
  }
}

```

10. Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat file kebijakan izin (misalnya, `~/PermissionsForCWL.json`):

```

{
  "Statement": [
    {

```

```

    "Effect": "Allow",
    "Action": ["firehose:PutRecord"],
    "Resource": [
        "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
    ]
}

```

11. Kaitkan kebijakan izin dengan peran menggunakan `put-role-policy` perintah:

```

aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json

```

12. Setelah aliran pengiriman Amazon Data Firehose dalam keadaan aktif dan Anda telah membuat peran IAM, Anda dapat membuat filter langganan CloudWatch Log. Filter langganan segera memulai aliran data log waktu nyata dari grup log yang dipilih ke aliran pengiriman Amazon Data Firehose Anda:

```

aws logs put-subscription-filter \
    --log-group-name "CloudTrail" \
    --filter-name "Destination" \
    --filter-pattern "${$.userIdentity.type = Root}" \
    --destination-arn "arn:aws:firehose:region:123456789012:deliverystream/my-
delivery-stream" \
    --role-arn "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"

```

13. Setelah Anda mengatur filter langganan, CloudWatch Log akan meneruskan semua peristiwa log masuk yang cocok dengan pola filter ke aliran pengiriman Amazon Data Firehose Anda. Data Anda akan mulai muncul di Amazon S3 berdasarkan interval buffer waktu yang ditetapkan pada aliran pengiriman Amazon Data Firehose Anda. Setelah waktu tertentu berlalu, Anda dapat memverifikasi data dengan memeriksa Bucket Amazon S3 Anda.

```

aws s3api list-objects --bucket 'amzn-s3-demo-bucket2' --prefix 'firehose/'
{
    "Contents": [
        {
            "LastModified": "2015-10-29T00:01:25.000Z",
            "ETag": "\"a14589f8897f4089d3264d9e2d1f1610\"",
            "StorageClass": "STANDARD",
            "Key": "firehose/2015/10/29/00/my-delivery-stream-2015-10-29-00-01-21-
a188030a-62d2-49e6-b7c2-b11f1a7ba250",

```

```

      "Owner": {
        "DisplayName": "cloudwatch-logs",
        "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b5"
      },
      "Size": 593
    },
    {
      "LastModified": "2015-10-29T00:35:41.000Z",
      "ETag": "\"a7035b65872bb2161388ffb63dd1aec5\"",
      "StorageClass": "STANDARD",
      "Key": "firehose/2015/10/29/00/my-delivery-
stream-2015-10-29-00-35-40-7cc92023-7e66-49bc-9fd4-fc9819cc8ed3",
      "Owner": {
        "DisplayName": "cloudwatch-logs",
        "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b6"
      },
      "Size": 5752
    }
  ]
}

```

```

aws s3api get-object --bucket 'amzn-s3-demo-bucket2' --key 'firehose/2015/10/29/00/
my-delivery-stream-2015-10-29-00-01-21-a188030a-62d2-49e6-b7c2-b11f1a7ba250'
testfile.gz

```

```

{
  "AcceptRanges": "bytes",
  "ContentType": "application/octet-stream",
  "LastModified": "Thu, 29 Oct 2015 00:07:06 GMT",
  "ContentLength": 593,
  "Metadata": {}
}

```

Data dalam objek Amazon S3 dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
zcat testfile.gz
```

Contoh 4: Filter berlangganan dengan OpenSearch Layanan Amazon

Dalam contoh ini, Anda akan membuat langganan CloudWatch Log yang mengirimkan peristiwa log masuk yang cocok dengan filter yang ditentukan ke domain OpenSearch Layanan Anda.

Untuk membuat filter langganan untuk OpenSearch Layanan

1. Buat domain OpenSearch Layanan. Untuk informasi selengkapnya, lihat [Membuat domain OpenSearch Layanan](#)
2. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
3. Pada panel navigasi, pilih Grup log.
4. Pilih nama grup log.
5. Pilih Tindakan, Filter langganan, Buat filter langganan OpenSearch Layanan Amazon.
6. Pilih apakah Anda ingin melakukan pengaliran ke kluster di akun ini atau akun lain.
 - Jika Anda memilih Akun ini, pilih domain yang Anda buat di langkah 1.
 - Jika Anda memilih Akun lain, masukkan ARN dan titik akhir domain tersebut.
7. Jika Anda memilih akun lain, berikan domain ARN dan endpoint.
8. Untuk Amazon OpenSearch Service cluster pilih nama cluster tempat data grup log akan dikirimkan
9. Pilih format log.
10. Untuk pola filter Langganan, masukkan istilah atau pola yang akan ditemukan di peristiwa log Anda. Ini memastikan bahwa Anda hanya mengirim data yang Anda minati ke kluster OpenSearch Layanan Anda. Untuk informasi selengkapnya, lihat [Filter sintaks pola untuk filter metrik](#).
11. (Opsional) Untuk Pilih data log yang akan diuji, pilih aliran log lalu pilih Pola uji untuk memverifikasi bahwa filter pencarian Anda mengembalikan hasil yang Anda harapkan.
12. Pilih Mulai streaming.

Filter berlangganan tingkat akun

Important

Ada risiko menyebabkan loop rekursif tak terbatas dengan filter berlangganan yang dapat menyebabkan peningkatan besar dalam penagihan konsumsi jika tidak ditangani. Untuk

mengurangi risiko ini, kami menyarankan Anda menggunakan kriteria pemilihan dalam filter langganan tingkat akun Anda untuk mengecualikan grup log yang mengambil data log dari sumber daya yang merupakan bagian dari alur kerja pengiriman langganan. Untuk informasi selengkapnya tentang masalah ini dan menentukan grup log mana yang akan dikecualikan, lihat [Pencegahan rekursi log](#).

Anda dapat menetapkan kebijakan berlangganan tingkat akun yang menyertakan subset grup log di akun. Kebijakan berlangganan akun dapat berfungsi dengan Amazon Kinesis Data AWS Lambda Streams, atau Amazon Data Firehose. Log yang dikirim ke layanan melalui kebijakan berlangganan tingkat akun dikodekan base64 dan dikompresi dengan format gzip. Bagian ini memberikan contoh yang dapat Anda ikuti untuk membuat langganan tingkat akun untuk Amazon Kinesis Data Streams, Lambda, dan Firehose.

Note

Untuk melihat daftar semua kebijakan filter langganan di akun Anda, gunakan `describe-account-policies` perintah dengan nilai `SUBSCRIPTION_FILTER_POLICY` `--policy-type` parameter. Untuk informasi lebih lanjut, lihat [describe-account-policies](#).

Contoh

- [Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams](#)
- [Contoh 2: Filter berlangganan dengan AWS Lambda](#)
- [Contoh 3: Filter berlangganan dengan Amazon Data Firehose](#)

Contoh 1: Filter langganan dengan Amazon Kinesis Data Streams

Sebelum Anda membuat aliran data Amazon Kinesis Data Streams untuk digunakan dengan kebijakan berlangganan tingkat akun, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat aliran dengan pecahan yang cukup untuk menangani volume ini. Jika aliran tidak memiliki cukup pecahan, itu dibatasi. Untuk informasi selengkapnya tentang batas volume streaming, lihat [Kuota dan Batas](#) dalam dokumentasi Amazon Kinesis Data Streams.

⚠ Warning

Karena peristiwa log dari beberapa grup log diteruskan ke tujuan, ada risiko pembatasan. Kiriman yang dibatasi dicoba ulang hingga 24 jam. Setelah 24 jam, kiriman yang gagal dijatuhkan.

Untuk mengurangi risiko pelambatan, Anda dapat mengambil langkah-langkah berikut:

- Pantau aliran Amazon Kinesis Data Streams CloudWatch Anda dengan metrik. Ini membantu Anda mengidentifikasi pelambatan dan menyesuaikan konfigurasi Anda sesuai dengan itu. Misalnya, `DeliveryThrottling` metrik melacak jumlah peristiwa log yang CloudWatch Log dibatasi saat meneruskan data ke tujuan langganan. Untuk informasi selengkapnya, lihat [Pemantauan dengan CloudWatch metrik](#).
- Gunakan mode kapasitas sesuai permintaan untuk streaming Anda di Amazon Kinesis Data Streams. Mode sesuai permintaan langsung mengakomodasi beban kerja Anda saat naik atau turun. Untuk informasi selengkapnya, lihat [Mode sesuai permintaan](#).
- Batasi pola filter langganan CloudWatch Log agar sesuai dengan kapasitas streaming Anda di Amazon Kinesis Data Streams. Jika Anda mengirim terlalu banyak data ke aliran, Anda mungkin perlu mengurangi ukuran filter atau menyesuaikan kriteria filter.

Contoh berikut menggunakan kebijakan berlangganan tingkat akun untuk meneruskan semua peristiwa log ke aliran di Amazon Kinesis Data Streams. Pola filter cocok dengan peristiwa log apa pun dengan teks `Test` dan meneruskannya ke aliran di Amazon Kinesis Data Streams.

Membuat kebijakan berlangganan tingkat akun untuk Amazon Kinesis Data Streams

1. Buat aliran tujuan menggunakan perintah berikut:

```
$ C:\> aws kinesis create-stream --stream-name "TestStream" --shard-count 1
```

2. Tunggu beberapa menit hingga streaming menjadi aktif. Anda dapat memverifikasi apakah aliran aktif dengan menggunakan perintah [describe-stream](#) untuk memeriksa `StreamDescription.StreamStatusproperty`.

```
aws kinesis describe-stream --stream-name "TestStream"
```

Berikut ini adalah output contoh:

```
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "TestStream",
    "StreamARN": "arn:aws:kinesis:region:123456789012:stream/TestStream",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "EXAMPLE8463463374607431768211455",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
            "EXAMPLE688818456679503831981458784591352702181572610"
        }
      }
    ]
  }
}
```

3. Buat peran IAM yang akan memberikan izin CloudWatch Log untuk memasukkan data ke aliran Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file (misalnya, ~/TrustPolicyForCWL-Kinesis.json). Gunakan editor teks untuk membuat kebijakan ini.

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global untuk membantu mencegah masalah keamanan wakil yang membingungkan. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
    }
  }
}
```

- Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dihasilkan, karena Anda juga akan membutuhkannya untuk langkah selanjutnya:

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
file:///~/TrustPolicyForCWL-Kinesis.json
```

Berikut adalah contoh output.

```
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.amazonaws.com"
        },
        "Condition": {
          "StringLike": {
            "aws:SourceArn": { "arn:aws:logs:region:123456789012:*" }
          }
        }
      }
    },
    "RoleId": "EXAMPLE450GAB4HC5F431",
    "CreateDate": "2023-05-29T13:46:29.431Z",
    "RoleName": "CWLtoKinesisRole",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
  }
}
```

- Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, Anda akan membuat kebijakan izin dalam file (misalnya, `~/PermissionsForCWL-Kinesis.json`). Gunakan editor teks untuk membuat kebijakan ini. Jangan gunakan konsol IAM untuk membuatnya.

```
{
  "Statement": [
    {
```



```

    "Effect": "Allow",
    "Action": "kinesis:PutRecord",
    "Resource": "arn:aws:kinesis:region:123456789012:stream/TestStream"
  }
]
}

```

6. Kaitkan kebijakan izin dengan peran menggunakan [put-role-policy](#) perintah berikut:

```

aws iam put-role-policy --role-name CWLtoKinesisRole --policy-name Permissions-
Policy-For-CWL --policy-document file://~/PermissionsForCWL-Kinesis.json

```

7. Setelah aliran dalam status Aktif dan Anda telah membuat peran IAM, Anda dapat membuat kebijakan filter langganan CloudWatch Log. Kebijakan segera memulai aliran data log waktu nyata ke aliran Anda. Dalam contoh ini, semua peristiwa log yang berisi string dialirkan, kecuali yang ERROR ada di grup log bernama LogGroupToExclude1 dan LogGroupToExclude2.

```

aws logs put-account-policy \
  --policy-name "ExamplePolicy" \
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \
  --policy-document '{"RoleArn":"arn:aws:iam::123456789012:role/
CWLtoKinesisRole", "DestinationArn":"arn:aws:kinesis:region:123456789012:stream/
TestStream", "FilterPattern": "Test", "Distribution": "Random"}' \
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
"LogGroupToExclude2"]' \
  --scope "ALL"

```

8. Setelah menyiapkan filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter dan kriteria pemilihan ke aliran Anda.

`selection-criteria` Bidang ini opsional, tetapi penting untuk mengecualikan grup log yang dapat menyebabkan rekursi log tak terbatas dari filter langganan. Untuk informasi selengkapnya tentang masalah ini dan menentukan grup log mana yang akan dikecualikan, lihat [Pencegahan rekursi log](#). Saat ini, NOT IN adalah satu-satunya operator yang didukung untuk `selection-criteria`.

Anda dapat memverifikasi bahwa alur peristiwa log menggunakan iterator pecahan Amazon Kinesis Data Streams dan menggunakan perintah Amazon Kinesis Data Streams untuk mengambil beberapa rekaman Amazon `get-records` Kinesis Data Streams:

```
aws kinesis get-shard-iterator --stream-name TestStream --shard-id
shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
  "ShardIterator":
  "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRgB9v4scv+3vaq+f+0IK8zM5My8ID
+g6rMo7UKWeI4+IWIK20Sh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRgB9v4scv+3vaq+f+0IK8zM5My8ID
+g6rMo7UKWeI4+IWIK20Sh0uP"
```

Anda mungkin perlu menggunakan perintah ini beberapa kali sebelum Amazon Kinesis Data Streams mulai mengembalikan data.

Anda akan melihat respons dengan array catatan. Atribut Data dalam catatan Amazon Kinesis Data Streams dikodekan base64 dan dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "messageType": "DATA_MESSAGE",
  "owner": "123456789012",
  "logGroup": "Example1",
  "logStream": "logStream1",
  "subscriptionFilters": [
    "ExamplePolicy"
  ],
  "logEvents": [
    {
```

```

      "id": "31953106606966983378809025079804211143289615424298221568",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221569",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221570",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}"}
    }
  ],
  "policyLevel": "ACCOUNT_LEVEL_POLICY"
}

```

Elemen kunci dalam struktur data adalah sebagai berikut:

messageType

Pesan data akan menggunakan tipe "DATA_MESSAGE". Terkadang CloudWatch Log mungkin memancarkan catatan Amazon Kinesis Data Streams dengan tipe "CONTROL_MESSAGE", terutama untuk memeriksa apakah tujuan dapat dijangkau.

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti "id" adalah pengenalan unik untuk setiap log acara.

PolicyLevel

Tingkat di mana kebijakan itu ditegakkan. "ACCOUNT_LEVEL_POLICY" adalah `policyLevel` untuk kebijakan filter langganan tingkat akun.

Contoh 2: Filter berlangganan dengan AWS Lambda

Dalam contoh ini, Anda akan membuat kebijakan filter langganan tingkat akun CloudWatch Log yang mengirimkan data log ke fungsi Anda AWS Lambda .

Warning

Sebelum membuat fungsi Lambda, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat fungsi yang dapat menangani volume ini. Jika fungsi tidak dapat menangani volume, aliran log akan dibatasi. Karena peristiwa log dari semua grup log atau subset grup log akun diteruskan ke tujuan, ada risiko pembatasan. Untuk informasi selengkapnya tentang batas Lambda, lihat [Batas AWS Lambda](#).

Untuk membuat kebijakan filter langganan tingkat akun untuk Lambda

1. Buat AWS Lambda fungsinya.

Pastikan bahwa Anda telah mengatur peran eksekusi Lambda. Untuk informasi selengkapnya, lihat: [Langkah 2.2: Buat IAM role \(peran eksekusi\)](#) dalam Panduan Developer AWS Lambda .

2. Buka editor teks dan buat file bernama `helloWorld.js` dengan isi sebagai berikut:

```
var zlib = require('zlib');
exports.handler = function(input, context) {
  var payload = Buffer.from(input.awslogs.data, 'base64');
  zlib.gunzip(payload, function(e, result) {
    if (e) {
      context.fail(e);
    } else {
      result = JSON.parse(result.toString());
    }
  });
}
```

```

        console.log("Event Data:", JSON.stringify(result, null, 2));
        context.succeed();
    }
});
};

```

3. Buat zip file helloWorld.js dan simpan dengan nama helloWorld.zip.
4. Gunakan perintah berikut, di mana perannya adalah peran eksekusi Lambda yang Anda atur di langkah pertama:

```

aws lambda create-function \
  --function-name helloworld \
  --zip-file fileb://file-path/helloWorld.zip \
  --role lambda-execution-role-arn \
  --handler helloWorld.handler \
  --runtime nodejs18.x

```

5. Berikan CloudWatch Log izin untuk menjalankan fungsi Anda. Gunakan perintah berikut, ganti akun placeholder dengan akun Anda sendiri.

```

aws lambda add-permission \
  --function-name "helloworld" \
  --statement-id "helloworld" \
  --principal "logs.amazonaws.com" \
  --action "lambda:InvokeFunction" \
  --source-arn "arn:aws:logs:region:123456789012:log-group:*" \
  --source-account "123456789012"

```

6. Buat kebijakan filter langganan tingkat akun menggunakan perintah berikut, ganti akun placeholder dengan akun Anda sendiri. Dalam contoh ini, semua peristiwa log yang berisi string dialirkan, kecuali yang ERROR ada di grup log bernama LogGroupToExclude1 dan LogGroupToExclude2.

```

aws logs put-account-policy \
  --policy-name "ExamplePolicyLambda" \
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \
  --policy-document
'{"DestinationArn":"arn:aws:lambda:region:123456789012:function:helloWorld",
"FilterPattern": "Test", "Distribution": "Random"}' \
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
"LogGroupToExclude2"]' \

```

```
--scope "ALL"
```

Setelah menyiapkan filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter dan kriteria pemilihan ke aliran Anda.

`selection-criteria` Bidang ini opsional, tetapi penting untuk mengecualikan grup log yang dapat menyebabkan rekursi log tak terbatas dari filter langganan. Untuk informasi selengkapnya tentang masalah ini dan menentukan grup log mana yang akan dikecualikan, lihat [Pencegahan rekursi log](#). Saat ini, NOT IN adalah satu-satunya operator yang didukung untuk `selection-criteria`.

7. (Opsional) Uji menggunakan contoh log acara. Di jendela perintah, jalankan perintah berikut, yang akan menempatkan pesan log sederhana ke dalam pengaliran langganan.

Untuk melihat output dari fungsi Lambda Anda, navigasikan ke fungsi Lambda di mana Anda akan melihat output di: `aws/lambda/helloworld`

```
aws logs put-log-events --log-group-name Example1 --log-stream-name logStream1 --
log-events "[{\"timestamp\":\"CURRENT_TIMESTAMP_MILLIS\", \"message\": \"Simple Lambda
Test\"}]"
```

Anda akan melihat respons dengan array Lambda. Atribut Data dalam catatan Lambda adalah base64 dikodekan dan dikompresi dengan format gzip. Muatan sebenarnya yang diterima oleh Lambda memiliki format berikut `{ "awslogs": { "data": "BASE64ENCODED_GZIP_COMPRESSED_DATA" } }` Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "messageType": "DATA_MESSAGE",
  "owner": "123456789012",
  "logGroup": "Example1",
  "logStream": "logStream1",
  "subscriptionFilters": [
    "ExamplePolicyLambda"
  ],
  "logEvents": [
```

```

    {
      "id": "31953106606966983378809025079804211143289615424298221568",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}\"
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221569",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}\"
    },
    {
      "id": "31953106606966983378809025079804211143289615424298221570",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":
\\\"Root\\\"}\"
    }
  ],
  "policyLevel": "ACCOUNT_LEVEL_POLICY"
}

```

Note

Filter langganan tingkat akun tidak akan diterapkan ke grup log fungsi Lambda tujuan. Ini untuk mencegah rekursi log tak terbatas yang dapat menyebabkan peningkatan penagihan konsumsi. Untuk informasi lebih lanjut tentang masalah ini, lihat [Pencegahan rekursi log](#).

Elemen kunci dalam struktur data adalah sebagai berikut:

messageType

Pesan data akan menggunakan tipe "DATA_MESSAGE". Terkadang CloudWatch Log mungkin memancarkan catatan Amazon Kinesis Data Streams dengan tipe "CONTROL_MESSAGE", terutama untuk memeriksa apakah tujuan dapat dijangkau.

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti "id" adalah pengenal unik untuk setiap log acara.

PolicyLevel

Tingkat di mana kebijakan itu ditegakkan. "ACCOUNT_LEVEL_POLICY" adalah `policyLevel` untuk kebijakan filter langganan tingkat akun.

Contoh 3: Filter berlangganan dengan Amazon Data Firehose

Dalam contoh ini, Anda akan membuat kebijakan filter langganan tingkat akun CloudWatch Log yang mengirimkan peristiwa log masuk yang cocok dengan filter yang ditentukan ke aliran pengiriman Amazon Data Firehose. Data yang dikirim dari CloudWatch Log ke Amazon Data Firehose sudah dikompresi dengan kompresi gzip level 6, jadi Anda tidak perlu menggunakan kompresi dalam aliran pengiriman Firehose Anda. Anda kemudian dapat menggunakan fitur dekompresi di Firehose untuk mendekompresi log secara otomatis. Untuk informasi selengkapnya, lihat [Menulis ke Kinesis Data CloudWatch Firehose Menggunakan Log](#).

Warning

Sebelum Anda membuat aliran Firehose, hitung volume data log yang akan dihasilkan. Pastikan untuk membuat aliran Firehose yang dapat menangani volume ini. Jika pengaliran tidak dapat menangani volume, pengaliran log akan mengalami throttling. Untuk informasi selengkapnya tentang batas volume aliran Firehose, lihat Batas Data [Firehose Amazon](#) Data.

Untuk membuat filter langganan untuk Firehose

1. Buat bucket Amazon Simple Storage Service (Amazon S3). Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang sudah ada, lewati ke langkah 2.

Jalankan perintah berikut, dengan mengganti Wilayah placeholder dengan Wilayah yang ingin Anda gunakan:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration  
LocationConstraint=region
```

Berikut ini adalah output contoh:

```
{  
  "Location": "/amzn-s3-demo-bucket2"  
}
```

2. Buat peran IAM yang memberikan izin Amazon Data Firehose untuk memasukkan data ke dalam bucket Amazon S3 Anda.

Untuk informasi selengkapnya, lihat [Mengontrol Akses dengan Amazon Data Firehose di Panduan Pengembang](#) Amazon Data Firehose.

Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForFirehose.json` sebagai berikut:

```
{  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": { "Service": "firehose.amazonaws.com" },  
      "Action": "sts:AssumeRole"  
    }  
  ]  
}
```

3. Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Catat nilai `Role.Arn` yang dikembalikan, karena Anda akan membutuhkannya di langkah selanjutnya:

```
aws iam create-role \  
--role-name FirehoseToS3Role \  

```

```
--assume-role-policy-document file://~/TrustPolicyForFirehose.json

{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "firehose.amazonaws.com"
        }
      }
    },
    "RoleId": "EXAMPLE50GAB4HC5F431",
    "CreateDate": "2023-05-29T13:46:29.431Z",
    "RoleName": "FirehoseToS3Role",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:role/FirehoseToS3Role"
  }
}
```

4. Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan Firehose di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file ~/PermissionsForFirehose.json:

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:AbortMultipartUpload",
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:PutObject" ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket2",
        "arn:aws:s3:::amzn-s3-demo-bucket2/*" ]
    }
  ]
}
```

5. Kaitkan kebijakan izin dengan peran menggunakan `put-role-policy` perintah berikut:

```
aws iam put-role-policy --role-name FirehoseToS3Role --policy-name Permissions-Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. Buat aliran pengiriman Firehose tujuan sebagai berikut, ganti nilai placeholder untuk `RoleARN` dan `BucketARN` dengan peran dan bucket yang Anda buat: ARNs

```
aws firehose create-delivery-stream \
  --delivery-stream-name 'my-delivery-stream' \
  --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::123456789012:role/FirehoseToS3Role", "BucketARN":  
  "arn:aws:s3:::amzn-s3-demo-bucket2"}'
```

Firehose secara otomatis menggunakan awalan dalam format waktu `YYYY/MM/DD/HH` UTC untuk objek Amazon S3 yang dikirimkan. Anda dapat menentukan prefiks tambahan untuk ditambahkan di depan prefiks format waktu. Jika prefiks berakhir dengan garis miring (`/`), itu akan muncul sebagai folder dalam bucket Amazon S3.

7. Tunggu beberapa menit hingga streaming menjadi aktif. Anda dapat menggunakan `describe-delivery-stream` perintah Firehose untuk memeriksa `DeliveryStreamDescription` `DeliveryStreamStatus` properti. Selain itu, perhatikan `DeliveryStreamDescription`. `DeliveryStream` Nilai ARN, karena Anda akan membutuhkannya di langkah selanjutnya:

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
{
  "DeliveryStreamDescription": {
    "HasMoreDestinations": false,
    "VersionId": "1",
    "CreateTimestamp": 1446075815.822,
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:123456789012:deliverystream/my-delivery-stream",
    "DeliveryStreamStatus": "ACTIVE",
    "DeliveryStreamName": "my-delivery-stream",
    "Destinations": [
      {
        "DestinationId": "destinationId-000000000001",
        "S3DestinationDescription": {
          "CompressionFormat": "UNCOMPRESSED",
          "EncryptionConfiguration": {
            "NoEncryptionConfig": "NoEncryption"
          }
        }
      }
    ]
  }
}
```

```

    },
    "RoleARN": "delivery-stream-role",
    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket2",
    "BufferingHints": {
      "IntervalInSeconds": 300,
      "SizeInMBs": 5
    }
  }
]
}
}

```

8. Buat peran IAM yang memberikan izin CloudWatch Log untuk memasukkan data ke aliran pengiriman Firehose Anda. Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForCWL.json`:

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global untuk membantu mencegah masalah keamanan wakil yang membingungkan. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```

{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringLike": {
        "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
      }
    }
  }
}

```

9. Gunakan perintah `create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Catat nilai `Role.Arn` yang dikembalikan, karena Anda akan membutuhkannya di langkah selanjutnya:

```

aws iam create-role \
--role-name CWLtoKinesisFirehoseRole \
--assume-role-policy-document file:///~/TrustPolicyForCWL.json

```

```
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.amazonaws.com"
        },
        "Condition": {
          "StringLike": {
            "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
          }
        }
      }
    },
    "RoleId": "AA01IAH450GAB4HC5F431",
    "CreateDate": "2015-05-29T13:46:29.431Z",
    "RoleName": "CWLtoKinesisFirehoseRole",
    "Path": "/",
    "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
  }
}
```

10. Buat kebijakan izin untuk menentukan tindakan apa yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat file kebijakan izin (misalnya, ~/PermissionsForCWL.json):

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["firehose:PutRecord"],
      "Resource": [
        "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-  
name"]
      }
    ]
  }
}
```

11. Kaitkan kebijakan izin dengan peran menggunakan put-role-policy perintah:

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. Setelah aliran pengiriman Amazon Data Firehose dalam status aktif dan Anda telah membuat peran IAM, Anda dapat membuat kebijakan filter langganan tingkat akun CloudWatch Log. Kebijakan segera memulai aliran data log real-time dari grup log yang dipilih ke aliran pengiriman Amazon Data Firehose Anda:

```
aws logs put-account-policy \
  --policy-name "ExamplePolicyFirehose" \
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \
  --policy-document '{"RoleArn":"arn:aws:iam::123456789012:role/
CWLtoKinesisFirehoseRole", "DestinationArn":"arn:aws:firehose:us-
east-1:123456789012:deliverystream/delivery-stream-name", "FilterPattern": "Test",
"Distribution": "Random"}' \
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
"LogGroupToExclude2"]' \
  --scope "ALL"
```

13. Setelah menyiapkan filter langganan, CloudWatch Log meneruskan peristiwa log masuk yang cocok dengan pola filter ke aliran pengiriman Amazon Data Firehose Anda.

`selection-criteria` Bidang ini opsional, tetapi penting untuk mengecualikan grup log yang dapat menyebabkan rekursi log tak terbatas dari filter langganan. Untuk informasi selengkapnya tentang masalah ini dan menentukan grup log mana yang akan dikecualikan, lihat [Pencegahan rekursi log](#). Saat ini, NOT IN adalah satu-satunya operator yang didukung untuk `selection-criteria`.

Data Anda akan mulai muncul di Amazon S3 berdasarkan interval buffer waktu yang ditetapkan pada aliran pengiriman Amazon Data Firehose Anda. Setelah waktu tertentu berlalu, Anda dapat memverifikasi data dengan memeriksa Bucket Amazon S3 Anda.

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket2' --prefix 'firehose/'
{
  "Contents": [
    {
      "LastModified": "2023-10-29T00:01:25.000Z",
      "ETag": "\"a14589f8897f4089d3264d9e2d1f1610\"",
      "StorageClass": "STANDARD",
```

```

      "Key": "firehose/2015/10/29/00/my-delivery-stream-2015-10-29-00-01-21-
a188030a-62d2-49e6-b7c2-b11f1a7ba250",
      "Owner": {
        "DisplayName": "cloudwatch-logs",
        "ID": "1ec9cf700ef6be062b19584e0b7d84ecc19237f87b5"
      },
      "Size": 593
    },
    {
      "LastModified": "2015-10-29T00:35:41.000Z",
      "ETag": "\"a7035b65872bb2161388ffb63dd1aec5\"",
      "StorageClass": "STANDARD",
      "Key": "firehose/2023/10/29/00/my-delivery-stream-2023-10-29-00-35-40-
EXAMPLE-7e66-49bc-9fd4-fc9819cc8ed3",
      "Owner": {
        "DisplayName": "cloudwatch-logs",
        "ID": "EXAMPLE6be062b19584e0b7d84ecc19237f87b6"
      },
      "Size": 5752
    }
  ]
}

```

```

aws s3api get-object --bucket 'amzn-s3-demo-bucket2' --key 'firehose/2023/10/29/00/
my-delivery-stream-2023-10-29-00-01-21-a188030a-62d2-49e6-b7c2-b11f1a7ba250'
testfile.gz

```

```

{
  "AcceptRanges": "bytes",
  "ContentType": "application/octet-stream",
  "LastModified": "Thu, 29 Oct 2023 00:07:06 GMT",
  "ContentLength": 593,
  "Metadata": {}
}

```

Data dalam objek Amazon S3 dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
zcat testfile.gz
```

Langganan lintas akun Lintas wilayah

Anda dapat berkolaborasi dengan pemilik AWS akun lain dan menerima peristiwa log mereka di AWS sumber daya Anda, seperti Amazon Kinesis atau Amazon Data Firehose stream (ini dikenal sebagai berbagi data lintas akun). Misalnya, data peristiwa log ini dapat dibaca dari Amazon Kinesis Data Streams atau aliran Firehose terpusat untuk melakukan pemrosesan dan analisis kustom. Pemrosesan khusus sangat berguna saat Anda berkolaborasi dan menganalisis data di banyak akun.

Misalnya, grup keamanan informasi perusahaan mungkin ingin menganalisis data untuk deteksi intrusi waktu nyata atau perilaku anomali agar bisa melakukan audit akun di semua divisi di perusahaan dengan mengumpulkan log produksi gabungan mereka untuk pemrosesan pusat. Aliran data peristiwa secara real-time di seluruh akun tersebut dapat dirakit dan dikirimkan ke grup keamanan informasi, yang dapat menggunakan Amazon Kinesis Data Streams untuk melampirkan data ke sistem analitik keamanan yang ada.

Note

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, sumber daya AWS yang ditunjuk oleh tujuan dapat berada di Wilayah yang berbeda. Dalam contoh di bagian berikut, semua sumber daya khusus Wilayah dibuat di AS Timur (Virginia N.).

Jika Anda telah mengonfigurasi AWS Organizations dan bekerja dengan akun anggota, Anda dapat menggunakan sentralisasi log untuk mengumpulkan data log dari akun sumber ke akun pemantauan pusat.

Saat bekerja dengan grup log terpusat, Anda dapat menggunakan dimensi bidang sistem ini saat membuat filter langganan:

- `@aws.account`- Dimensi ini mewakili ID AWS akun dari mana peristiwa log berasal.
- `@aws.region`- Dimensi ini mewakili AWS wilayah tempat peristiwa log dihasilkan.

Dimensi ini membantu dalam mengidentifikasi sumber data log, memungkinkan penyaringan dan analisis metrik yang lebih terperinci yang berasal dari log terpusat.

Topik

- [Berbagi data log lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams](#)

- [Berbagi data log lintas wilayah lintas akun menggunakan Firehose](#)
- [Langganan tingkat akun lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams](#)
- [Langganan tingkat akun lintas wilayah lintas akun menggunakan Firehose](#)

Berbagi data log lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams

Saat membuat langganan lintas akun, Anda dapat menentukan satu akun atau organisasi untuk menjadi pengirim. Jika Anda menentukan organisasi, maka prosedur ini memungkinkan semua akun di organisasi untuk mengirim log ke akun penerima.

Untuk berbagi data log lintas akun, Anda perlu membuat pengirim dan penerima data log:

- Pengirim data log —mendapatkan informasi tujuan dari penerima dan memberi tahu CloudWatch Log bahwa Log siap mengirim peristiwa lognya ke tujuan yang ditentukan. Dalam prosedur di bagian lainnya, pengirim data log ditampilkan dengan nomor AWS akun fiksi 111111111111.

Jika Anda akan memiliki beberapa akun dalam satu organisasi yang mengirim log ke satu akun penerima, Anda dapat membuat kebijakan yang memberikan izin kepada semua akun di organisasi untuk mengirim log ke akun penerima. Anda masih harus menyiapkan filter langganan terpisah untuk setiap akun pengirim.

- Penerima data log —menyiapkan tujuan yang merangkum aliran Amazon Kinesis Data Streams dan CloudWatch memberi tahu Log bahwa penerima ingin menerima data log. Penerima kemudian membagikan informasi tentang tujuan ini dengan pengirim. Dalam prosedur di bagian lainnya, penerima data log ditampilkan dengan nomor AWS akun fiksi 999999999999.

Untuk mulai menerima peristiwa log dari pengguna lintas akun, penerima data log terlebih dahulu membuat tujuan CloudWatch Log. Setiap tujuan terdiri atas elemen kunci berikut:

Nama tujuan

Nama tujuan yang ingin Anda buat.

ARN Target

Nama Sumber Daya Amazon (ARN) dari AWS sumber daya yang ingin Anda gunakan sebagai tujuan umpan berlangganan.

Peran ARN

Peran AWS Identity and Access Management (IAM) yang memberikan CloudWatch Log izin yang diperlukan untuk memasukkan data ke aliran yang dipilih.

Kebijakan akses

Dokumen kebijakan IAM (dalam format JSON, ditulis menggunakan tata bahasa kebijakan IAM) yang mengatur set pengguna yang diizinkan untuk menulis ke tujuan Anda.

Note

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, AWS sumber daya yang ditunjuk tujuan dapat ditemukan di Wilayah yang berbeda. Dalam contoh di bagian berikut, semua sumber daya khusus Wilayah dibuat di US East (N. Virginia).

Topik

- [Menyiapkan langganan lintas akun baru](#)
- [Memperbarui langganan lintas akun yang ada](#)

Menyiapkan langganan lintas akun baru

Ikuti langkah-langkah di bagian ini untuk menyiapkan langganan log lintas akun baru.

Topik

- [Langkah 1: Buat tujuan](#)
- [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#)
- [Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun](#)
- [Langkah 4: Buat filter berlangganan](#)
- [Validasi alur peristiwa log](#)
- [Ubah keanggotaan tujuan saat runtime](#)

Langkah 1: Buat tujuan

Important

Semua langkah dalam prosedur ini harus dilakukan di akun penerima data log.

Untuk contoh ini, akun penerima data log memiliki ID akun 999999999999, sedangkan ID AWS akun pengirim AWS data log adalah 111111111111.

Contoh ini membuat tujuan menggunakan aliran Amazon Kinesis Data RecipientStream Streams yang disebut, dan peran CloudWatch yang memungkinkan Log untuk menulis data ke sana.

Saat tujuan dibuat, CloudWatch Log mengirimkan pesan pengujian ke tujuan atas nama akun penerima. Saat filter langganan aktif nanti, CloudWatch Log mengirimkan peristiwa log ke tujuan atas nama akun sumber.

Untuk membuat tujuan

1. Di akun penerima, buat aliran tujuan di Amazon Kinesis Data Streams. Di jendela perintah, ketik:

```
aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
```

2. Tunggu hingga streaming menjadi aktif. Anda dapat menggunakan perintah `aws kinesis describe-stream` untuk memeriksa `StreamDescription` `StreamStatus` properti. Selain itu, perhatikan `StreamDescription` nilai `streamArn` karena Anda akan meneruskannya ke CloudWatch Log nanti:

```
aws kinesis describe-stream --stream-name "RecipientStream"
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "RecipientStream",
    "StreamARN": "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "34028236692093846346337460743176EXAMPLE",
          "StartingHashKey": "0"
        }
      }
    ]
  }
}
```

```

    "SequenceNumberRange": {
      "StartingSequenceNumber":
"4955113521868881845667950383198145878459135270218EXAMPLE"
    }
  }
]
}
}
}

```

Mungkin diperlukan satu atau dua menit bagi pengaliran Anda untuk muncul dalam keadaan aktif.

3. Buat peran IAM yang memberikan izin kepada CloudWatch Log untuk memasukkan data ke aliran Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file `~/TrustPolicyForCWL.json`. Gunakan editor teks untuk membuat file kebijakan ini, jangan menggunakan konsol IAM.

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global yang menentukan `sourceAccountId` untuk membantu mencegah masalah keamanan wakil yang membingungkan. Jika Anda belum mengetahui ID akun sumber pada panggilan pertama, kami sarankan Anda memasukkan ARN tujuan di bidang ARN sumber. Dalam panggilan berikutnya, Anda harus mengatur ARN sumber menjadi ARN sumber sebenarnya yang Anda kumpulkan dari panggilan pertama. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```

{
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "logs.amazonaws.com"
    },
    "Condition": {
      "StringLike": {
        "aws:SourceArn": [
          "arn:aws:logs:region:sourceAccountId:*",
          "arn:aws:logs:region:recipientAccountId:*"
        ]
      }
    },
    "Action": "sts:AssumeRole"
  }
}

```

4. Gunakan perintah `aws iam create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dikembalikan karena itu juga akan diteruskan ke Log nanti: CloudWatch

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Condition": {
          "StringLike": {
            "aws:SourceArn": [
              "arn:aws:logs:region:sourceAccountId:*",
              "arn:aws:logs:region:recipientAccountId:*"
            ]
          }
        },
        "Principal": {
          "Service": "logs.amazonaws.com"
        }
      }
    },
    "RoleId": "AA0IIAH450GAB4HC5F431",
    "CreateDate": "2015-05-29T13:46:29.431Z",
    "RoleName": "CWLtoKinesisRole",
    "Path": "/",
    "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
  }
}
```

5. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file `~/PermissionsForCWL.json`:

```
{
  "Statement": [
    {
```

```

    "Effect": "Allow",
    "Action": "kinesis:PutRecord",
    "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
  }
]
}

```

6. Kaitkan kebijakan izin dengan peran dengan menggunakan perintah `aws iam: put-role-policy`

```

aws iam put-role-policy \
  --role-name CWLtoKinesisRole \
  --policy-name Permissions-Policy-For-CWL \
  --policy-document file://~/PermissionsForCWL.json

```

7. Setelah aliran dalam keadaan aktif dan Anda telah membuat peran IAM, Anda dapat membuat tujuan CloudWatch Log.
 - a. Langkah ini tidak mengaitkan kebijakan akses dengan tujuan Anda dan hanya langkah pertama dari dua langkah yang menyelesaikan pembuatan tujuan. Catat `DestinationArn` yang dikembalikan dalam muatan:

```

aws logs put-destination \
  --destination-name "testDestination" \
  --target-arn "arn:aws:kinesis:region:999999999999:stream/RecipientStream" \
  --role-arn "arn:aws:iam::999999999999:role/CWLtoKinesisRole"

{
  "DestinationName" : "testDestination",
  "RoleArn" : "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
  "DestinationArn" : "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
  "TargetArn" : "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream"
}

```

- b. Setelah langkah 7a selesai, di akun penerima data log, kaitkan kebijakan akses dengan tujuan. Kebijakan ini harus menentukan `PutSubscriptionFilter` tindakan log: dan memberikan izin ke akun pengirim untuk mengakses tujuan.

Kebijakan memberikan izin ke AWS akun yang mengirim log. Anda dapat menentukan hanya satu akun ini dalam kebijakan, atau jika akun pengirim adalah anggota organisasi, kebijakan dapat menentukan ID organisasi organisasi. Dengan cara ini, Anda dapat

membuat hanya satu kebijakan untuk mengizinkan beberapa akun dalam satu organisasi mengirim log ke akun tujuan ini.

Gunakan editor teks untuk membuat file bernama `~/AccessPolicy.json` dengan salah satu pernyataan kebijakan berikut.

Kebijakan contoh pertama ini memungkinkan semua akun di organisasi yang memiliki ID `o-1234567890` untuk mengirim log ke akun penerima.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": [
            "o-1234567890"
          ]
        }
      }
    }
  ]
}
```

Contoh berikutnya ini memungkinkan hanya akun pengirim data log (111111111111) untuk mengirim log ke akun penerima data log.

JSON

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

```

    "Statement": [
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": {
          "AWS": "111111111111"
        },
        "Action": "logs:PutSubscriptionFilter",
        "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination"
      }
    ]
  }
}

```

- c. Lampirkan kebijakan yang Anda buat pada langkah sebelumnya ke tujuan.

```

aws logs put-destination-policy \
  --destination-name "testDestination" \
  --access-policy file://~/AccessPolicy.json

```

Kebijakan akses ini memungkinkan pengguna di AWS Akun dengan ID 111111111111 untuk memanggil tujuan dengan ARN `arn:aws:logs:PutSubscriptionFilter::999999999999:destination:testDestination`. *region* Upaya pengguna lain untuk menelepon PutSubscriptionFilter terhadap tujuan ini akan ditolak.

Untuk memvalidasi hak istimewa pengguna berdasarkan kebijakan akses, lihat [Menggunakan Validator Kebijakan](#) dalam Panduan Pengguna IAM.

Setelah selesai, jika Anda menggunakan AWS Organizations izin lintas akun, ikuti langkah-langkahnya. [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#) Jika Anda memberikan izin langsung ke akun lain alih-alih menggunakan Organizations, Anda dapat melewati langkah itu dan melanjutkan ke. [Langkah 4: Buat filter berlangganan](#)

Langkah 2: (Hanya jika menggunakan organisasi) Buat peran IAM

Di bagian sebelumnya, jika Anda membuat tujuan menggunakan kebijakan akses yang memberikan izin kepada organisasi tempat akun 111111111111 berada, alih-alih memberikan izin langsung ke akun111111111111, ikuti langkah-langkah di bagian ini. Jika tidak, Anda dapat melompat ke [Langkah 4: Buat filter berlangganan](#).

Langkah-langkah di bagian ini membuat peran IAM, yang CloudWatch dapat mengasumsikan dan memvalidasi apakah akun pengirim memiliki izin untuk membuat filter langganan terhadap tujuan penerima.

Lakukan langkah-langkah di bagian ini di akun pengirim. Peran harus ada di akun pengirim, dan Anda menentukan ARN peran ini dalam filter berlangganan. Dalam contoh ini, akun pengirim adalah 111111111111.

Untuk membuat peran IAM yang diperlukan untuk langganan log lintas akun menggunakan AWS Organizations

1. Buat kebijakan kepercayaan berikut dalam sebuah file/
`TrustPolicyForCWLSubscriptionFilter.json`. Gunakan editor teks untuk membuat file kebijakan ini; jangan gunakan konsol IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. Buat peran IAM yang menggunakan kebijakan ini. Perhatikan `Arn` nilai yang dikembalikan oleh perintah, Anda akan membutuhkannya nanti dalam prosedur ini. Dalam contoh ini, kita gunakan `CWLtoSubscriptionFilterRole` untuk nama peran yang kita buat.

```
aws iam create-role \
  --role-name CWLtoSubscriptionFilterRole \
  --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda.
 - a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `~/PermissionsForCWLSubscriptionFilter.json`.

```
{
  "Statement": [
    {
```

```
        "Effect": "Allow",
        "Action": "logs:PutLogEvents",
        "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
    }
  ]
}
```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran yang Anda buat di langkah 2.

```
aws iam put-role-policy
--role-name CWLtoSubscriptionFilterRole
--policy-name Permissions-Policy-For-CWL-Subscription-filter
--policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

Setelah selesai, Anda dapat melanjutkan ke [Langkah 4: Buat filter berlangganan](#).

Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun

Menurut logika evaluasi kebijakan AWS lintas akun, untuk mengakses sumber daya lintas akun (seperti aliran Kinesis atau Firehose yang digunakan sebagai tujuan filter langganan), Anda harus memiliki kebijakan berbasis identitas di akun pengirim yang menyediakan akses eksplisit ke sumber tujuan lintas akun. Untuk informasi selengkapnya tentang logika evaluasi kebijakan, lihat [Logika evaluasi kebijakan lintas akun](#).

Anda dapat melampirkan kebijakan berbasis identitas ke peran IAM atau pengguna IAM yang Anda gunakan untuk membuat filter langganan. Kebijakan ini harus ada di akun pengiriman. Jika Anda menggunakan peran Administrator untuk membuat filter langganan, Anda dapat melewati langkah ini dan melanjutkan ke [Langkah 4: Buat filter berlangganan](#).

Untuk menambah atau memvalidasi izin IAM yang diperlukan untuk lintas akun

1. Masukkan perintah berikut untuk memeriksa peran IAM atau pengguna IAM mana yang digunakan untuk menjalankan perintah AWS log.

```
aws sts get-caller-identity
```

Perintah tersebut mengembalikan output serupa dengan berikut ini:

```
{
  "UserId": "User ID",
  "Account": "sending account id",
  "Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

Catat nilai yang diwakili oleh *RoleName* atau *UserName*.

2. Konsol Manajemen AWS Masuk ke akun pengiriman dan cari kebijakan terlampir dengan peran IAM atau pengguna IAM yang dikembalikan dalam output perintah yang Anda masukkan pada langkah 1.
3. Verifikasi bahwa kebijakan yang dilampirkan pada peran ini atau pengguna memberikan izin eksplisit untuk memanggil sumber `logs:PutSubscriptionFilter` daya tujuan lintas akun.

Kebijakan berikut memberikan izin untuk membuat filter langganan pada sumber daya tujuan apa pun hanya dalam satu AWS akun, akun999999999999:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSubscriptionFiltersOnAccountResources",
      "Effect": "Allow",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": [
        "arn:aws:logs::*:log-group:*",
        "arn:aws:logs*:123456789012:destination:*"
      ]
    }
  ]
}
```

Kebijakan berikut memberikan izin untuk membuat filter langganan hanya pada sumber daya tujuan tertentu yang dinamai `sampleDestination` dalam satu AWS akun, akun123456789012:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSubscriptionFilteronAccountResource",
      "Effect": "Allow",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": [
        "arn:aws:logs:*:*:log-group:*",
        "arn:aws:logs:*:123456789012:destination:sampleDestination"
      ]
    }
  ]
}
```

Langkah 4: Buat filter berlangganan

Setelah Anda membuat tujuan, akun penerima data log dapat berbagi ARN tujuan (arn:aws:logs:us-east-1:999999999999:destination:testDestination) dengan akun AWS lain sehingga mereka dapat mengirim log acara ke tujuan yang sama. Para pengguna akun pengirim ini kemudian membuat filter langganan pada grup log masing-masing berdasarkan tujuan ini. Filter langganan segera memulai aliran data log waktu nyata dari grup log yang dipilih ke tujuan yang ditentukan.

 Note

Jika Anda memberikan izin untuk filter langganan ke seluruh organisasi, Anda harus menggunakan ARN dari peran IAM yang Anda buat. [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#)

Dalam contoh berikut, filter langganan dibuat di akun pengiriman. filter dikaitkan dengan grup log yang berisi AWS CloudTrail peristiwa sehingga setiap aktivitas yang dicatat yang dibuat oleh AWS kredensial "Root" dikirimkan ke tujuan yang Anda buat sebelumnya. Tujuan itu merangkum aliran yang disebut "". RecipientStream

Langkah-langkah lainnya di bagian berikut mengasumsikan bahwa Anda telah mengikuti petunjuk dalam [Mengirim CloudTrail Acara ke CloudWatch Log](#) di Panduan AWS CloudTrail Pengguna dan membuat grup log yang berisi CloudTrail peristiwa Anda. Langkah-langkah ini mengasumsikan bahwa nama grup log ini adalah CloudTrail/logs.

Ketika Anda memasukkan perintah berikut, pastikan Anda masuk sebagai pengguna IAM atau menggunakan peran IAM yang Anda tambahkan kebijakan untuk, in. [Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun](#)

```
aws logs put-subscription-filter \
  --log-group-name "CloudTrail/logs" \
  --filter-name "RecipientStream" \
  --filter-pattern "${$.userIdentity.type = Root}" \
  --destination-arn "arn:aws:logs:region:999999999999:destination:testDestination"
```

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, tujuan dapat menunjuk ke AWS sumber daya seperti aliran Amazon Kinesis Data Streams yang terletak di Wilayah yang berbeda.

Validasi alur peristiwa log

Setelah Anda membuat filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter ke aliran yang dienkapsulasi dalam aliran tujuan yang disebut "". RecipientStream Pemilik tujuan dapat memverifikasi bahwa ini terjadi dengan menggunakan get-shard-iterator perintah aws kinesis untuk mengambil pecahan Amazon Kinesis Data Streams, dan menggunakan perintah aws kinesis get-records untuk mengambil beberapa catatan Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator \
  --stream-name RecipientStream \
  --shard-id shardId-000000000000 \
  --shard-iterator-type TRIM_HORIZON

{
  "ShardIterator":
  "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRgB9v4scv+3vaq+f
+0IK8zM5My8ID+g6rMo7UKWeI4+IWIKEXAMPLE"
}
```

```
aws kinesis get-records \
  --limit 10 \
  --shard-iterator
  "AAAAAAAAAAFGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRgB9v4scv+3vaq+f
+0IK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

Note

Anda mungkin perlu menjalankan kembali perintah `get-records` beberapa kali sebelum Amazon Kinesis Data Streams mulai mengembalikan data.

Anda akan melihat respons dengan larik catatan Amazon Kinesis Data Streams. Atribut data dalam catatan Amazon Kinesis Data Streams dikompresi dalam format gzip dan kemudian base64 dikodekan. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "owner": "111111111111",
  "logGroup": "CloudTrail/logs",
  "logStream": "111111111111_CloudTrail/logs_us-east-1",
  "subscriptionFilters": [
    "RecipientStream"
  ],
  "messageType": "DATA_MESSAGE",
  "logEvents": [
    {
      "id": "3195310660696698337880902507980421114328961542429EXAMPLE",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
    }"
  ],
  {
    "id": "3195310660696698337880902507980421114328961542429EXAMPLE",
    "timestamp": 1432826855000,
```

```

    "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
  \",
  },
  {
    \"id\": \"3195310660696698337880902507980421114328961542429EXAMPLE\",
    \"timestamp\": 1432826855000,
    \"message\": \"{\\\"eventVersion\\\":\\\"1.03\\\",\\\"userIdentity\\\":{\\\"type\\\":\\\"Root
  \",
  },
  }
]
}

```

Elemen kunci dalam struktur data ini adalah sebagai berikut:

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

messageType

Pesan data menggunakan tipe “DATA_MESSAGE”. Terkadang CloudWatch Log dapat memancarkan catatan Amazon Kinesis Data Streams dengan tipe “CONTROL_MESSAGE”, terutama untuk memeriksa apakah tujuan dapat dijangkau.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti ID adalah pengenalan unik untuk setiap log acara.

Ubah keanggotaan tujuan saat runtime

Anda mungkin mengalami situasi ketika Anda harus menambahkan atau menghapus keanggotaan beberapa pengguna dari tujuan yang Anda miliki. Anda dapat menggunakan perintah `put-`

destination-policy di tujuan Anda dengan kebijakan akses baru. Dalam contoh berikut, akun 111111111111 yang ditambahkan sudah sebelumnya dihentikan dari mengirim data log lagi, dan akun 222222222222 diaktifkan.

1. Ambil kebijakan yang saat ini terkait dengan TestDestination tujuan dan catat: AccessPolicy

```
aws logs describe-destinations \
  --destination-name-prefix "testDestination"

{
  "Destinations": [
    {
      "DestinationName": "testDestination",
      "RoleArn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
      "DestinationArn":
        "arn:aws:logs:region:999999999999:destination:testDestination",
      "TargetArn": "arn:aws:kinesis:region:999999999999:stream/RecipientStream",
      "AccessPolicy": "{\n\"Version\": \"2012-10-17\", \"Statement\":
        [{\n\"Sid\": \"\", \"Effect\": \"Allow\", \"Principal\": {\n\"AWS\":
        \"111111111111\"}, \"Action\": \"logs:PutSubscriptionFilter\", \"Resource\":
        \"arn:aws:logs:region:999999999999:destination:testDestination\"}] }"
    }
  ]
}
```

2. Perbarui kebijakan agar menunjukkan bahwa akun 111111111111 dihentikan, dan akun 222222222222 diaktifkan. Letakkan kebijakan ini di file ~/NewAccessPolicy.json:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "222222222222"
      },
      "Action": "logs:PutSubscriptionFilter",
      "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination"
    }
  ]
}
```



```
}  
]  
}
```

3. Panggilan `PutDestinationPolicy` untuk mengaitkan kebijakan yang ditentukan dalam `NewAccessPolicyfile.json` dengan tujuan:

```
aws logs put-destination-policy \  
--destination-name "testDestination" \  
--access-policy file://~/NewAccessPolicy.json
```

Ini pada akhirnya akan menonaktifkan log acara dari ID akun 111111111111. Log acara dari ID akun 222222222222 mulai mengalir ke tujuan segera setelah pemilik akun 222222222222 membuat filter langganan.

Memperbarui langganan lintas akun yang ada

Jika saat ini Anda memiliki langganan log lintas akun di mana akun tujuan hanya memberikan izin ke akun pengirim tertentu, dan Anda ingin memperbarui langganan ini sehingga akun tujuan memberikan akses ke semua akun di organisasi, ikuti langkah-langkah di bagian ini.

Topik

- [Langkah 1: Perbarui filter berlangganan](#)
- [Langkah 2: Perbarui kebijakan akses tujuan yang ada](#)

Langkah 1: Perbarui filter berlangganan

Note

Langkah ini diperlukan hanya untuk langganan lintas akun untuk log yang dibuat oleh layanan yang tercantum di [Aktifkan logging dari AWS layanan](#). Jika Anda tidak bekerja dengan log yang dibuat oleh salah satu grup log ini, Anda dapat melompat ke [Langkah 2: Perbarui kebijakan akses tujuan yang ada](#).

Dalam kasus tertentu, Anda harus memperbarui filter langganan di semua akun pengirim yang mengirim log ke akun tujuan. Pembaruan menambahkan peran IAM, yang CloudWatch dapat

mengasumsikan dan memvalidasi bahwa akun pengirim memiliki izin untuk mengirim log ke akun penerima.

Ikuti langkah-langkah di bagian ini untuk setiap akun pengirim yang ingin Anda perbarui untuk menggunakan ID organisasi untuk izin berlangganan lintas akun.

Dalam contoh di bagian ini, dua akun, 111111111111 dan 222222222222 sudah memiliki filter berlangganan yang dibuat untuk mengirim log ke akun999999999999. Nilai filter langganan yang ada adalah sebagai berikut:

```
## Existing Subscription Filter parameter values
\ --log-group-name "my-log-group-name"
\ --filter-name "RecipientStream"
\ --filter-pattern "{$.userIdentity.type = Root}"
\ --destination-arn "arn:aws:logs:region:999999999999:destination:testDestination"
```

Jika Anda perlu menemukan nilai parameter filter langganan saat ini, masukkan perintah berikut.

```
aws logs describe-subscription-filters
\ --log-group-name "my-log-group-name"
```

Untuk memperbarui filter langganan untuk mulai menggunakan organisasi IDs untuk izin log lintas akun

1. Buat kebijakan kepercayaan berikut dalam sebuah file `~/TrustPolicyForCWL.json`. Gunakan editor teks untuk membuat file kebijakan ini; jangan gunakan konsol IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. Buat peran IAM yang menggunakan kebijakan ini. Perhatikan nilai Arn Arn nilai yang dikembalikan oleh perintah, Anda akan membutuhkannya nanti dalam prosedur ini. Dalam contoh ini, kita gunakan `CWLtoSubscriptionFilterRole` untuk nama peran yang kita buat.

```
aws iam create-role
\ --role-name CWLtoSubscriptionFilterRole
```

```
\ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda.

- a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `PermissionsForCWLSubscriptionFilter.json`.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "logs:PutLogEvents",
      "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
    }
  ]
}
```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran yang Anda buat di langkah 2.

```
aws iam put-role-policy
  --role-name CWLtoSubscriptionFilterRole
  --policy-name Permissions-Policy-For-CWL-Subscription-filter
  --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. Masukkan perintah berikut untuk memperbarui filter langganan.

```
aws logs put-subscription-filter
\ --log-group-name "my-log-group-name"
\ --filter-name "RecipientStream"
\ --filter-pattern "${$.userIdentity.type = Root}"
\ --destination-arn
"arn:aws:logs:region:999999999999:destination:testDestination"
\ --role-arn "arn:aws:iam::111111111111:role/CWLtoSubscriptionFilterRole"
```

Langkah 2: Perbarui kebijakan akses tujuan yang ada

Setelah memperbarui filter langganan di semua akun pengirim, Anda dapat memperbarui kebijakan akses tujuan di akun penerima.

Dalam contoh berikut, akun penerima adalah 999999999999 dan tujuan diberi `namatestDestination`.

Pembaruan memungkinkan semua akun yang merupakan bagian dari organisasi dengan ID `o-1234567890` untuk mengirim log ke akun penerima. Hanya akun yang memiliki filter langganan yang dibuat yang benar-benar akan mengirim log ke akun penerima.

Untuk memperbarui kebijakan akses tujuan di akun penerima untuk mulai menggunakan ID organisasi untuk izin

1. Di akun penerima, gunakan editor teks untuk membuat `~/AccessPolicy.json` file dengan konten berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": [
            "o-1234567890"
          ]
        }
      }
    }
  ]
}
```

2. Masukkan perintah berikut untuk melampirkan kebijakan yang baru saja Anda buat ke tujuan yang ada. Untuk memperbarui tujuan agar menggunakan kebijakan akses dengan ID organisasi, bukan kebijakan akses yang mencantumkan AWS akun tertentu IDs, sertakan `force` parameternya.

⚠ Warning

Jika Anda bekerja dengan log yang dikirim oleh AWS layanan yang terdaftar di [Aktifkan logging dari AWS layanan](#), maka sebelum melakukan langkah ini, Anda harus terlebih dahulu memperbarui filter langganan di semua akun pengirim seperti yang dijelaskan di [Langkah 1: Perbarui filter berlangganan](#).

```
aws logs put-destination-policy
\ --destination-name "testDestination"
\ --access-policy file://~/AccessPolicy.json
\ --force
```

Berbagi data log lintas wilayah lintas akun menggunakan Firehose

Untuk berbagi data log lintas akun, Anda perlu membuat pengirim dan penerima data log:

- Pengirim data log —mendapatkan informasi tujuan dari penerima dan memberi tahu CloudWatch Log bahwa ia siap untuk mengirim peristiwa lognya ke tujuan yang ditentukan. Dalam prosedur di bagian lainnya, pengirim data log ditampilkan dengan nomor AWS akun fiksi 111111111111.
- Penerima data log —menyiapkan tujuan yang merangkum aliran Amazon Kinesis Data Streams dan CloudWatch memberi tahu Log bahwa penerima ingin menerima data log. Penerima kemudian membagikan informasi tentang tujuan ini dengan pengirim. Dalam prosedur di bagian lainnya, penerima data log ditampilkan dengan nomor AWS akun fiksi 222222222222.

Contoh di bagian ini menggunakan aliran pengiriman Firehose dengan penyimpanan Amazon S3. Anda juga dapat mengatur aliran pengiriman Firehose dengan pengaturan berbeda. Untuk informasi selengkapnya, lihat [Membuat Aliran Pengiriman Firehose](#).

i Note

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, AWS sumber daya yang ditunjuk tujuan dapat ditemukan di Wilayah yang berbeda.

 Note

Filter langganan Firehose untuk akun yang sama dan aliran pengiriman lintas wilayah didukung.

Topik

- [Langkah 1: Buat aliran pengiriman Firehose](#)
- [Langkah 2: Buat tujuan](#)
- [Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun](#)
- [Langkah 4: Buat filter berlangganan](#)
- [Memvalidasi alur peristiwa log](#)
- [Memodifikasi keanggotaan tujuan saat runtime](#)

Langkah 1: Buat aliran pengiriman Firehose

 Important

Sebelum Anda menyelesaikan langkah-langkah berikut, Anda harus menggunakan kebijakan akses, sehingga Firehose dapat mengakses bucket Amazon S3 Anda. Untuk informasi selengkapnya, lihat [Mengontrol Akses](#) di Panduan Pengembang Amazon Data Firehose. Semua langkah di bagian ini (Langkah 1) harus dilakukan di akun penerima data log. US East (N. Virginia) digunakan dalam contoh perintah berikut. Ganti Wilayah ini dengan Wilayah yang benar untuk penerapan Anda.

Untuk membuat aliran pengiriman Firehose yang akan digunakan sebagai tujuan

1. Buat bucket Amazon S3:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
LocationConstraint=us-east-1
```

2. Buat peran IAM yang memberikan izin Firehose untuk memasukkan data ke dalam bucket.

- a. Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForFirehose.json`.

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
  "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
  { "StringEquals": { "sts:ExternalId": "222222222222" } } } }
```

- b. Buat IAM role dengan menentukan file kebijakan kepercayaan yang baru saja Anda buat.

```
aws iam create-role \
  --role-name FirehosetoS3Role \
  --assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

- c. Output perintah ini akan terlihat serupa dengan yang berikut ini. Catat nama peran dan ARN peran.

```
{
  "Role": {
    "Path": "/",
    "RoleName": "FirehosetoS3Role",
    "RoleId": "AROAR3BXASEKW7K635M53",
    "Arn": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
    "CreateDate": "2021-02-02T07:53:10+00:00",
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Effect": "Allow",
        "Principal": {
          "Service": "firehose.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
            "sts:ExternalId": "222222222222"
          }
        }
      }
    }
  }
}
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan Firehose di akun Anda.
- a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `~/PermissionsForFirehose.json`. Bergantung pada kasus penggunaan Anda, Anda mungkin perlu menambahkan lebih banyak izin ke file ini.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "s3:PutObject",
      "s3:PutObjectAcl",
      "s3:ListBucket"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }]
}
```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran IAM.

```
aws iam put-role-policy --role-name FirehoseToS3Role --policy-name
Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. Masukkan perintah berikut untuk membuat aliran pengiriman Firehose. Ganti *my-role-arn* dan *amzn-s3-demo-bucket2-arn* dengan nilai yang benar untuk penerapan Anda.

```
aws firehose create-delivery-stream \
  --delivery-stream-name 'my-delivery-stream' \
  --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::222222222222:role/FirehoseToS3Role", "BucketARN":
  "arn:aws:s3:::amzn-s3-demo-bucket"}'
```

Outputnya akan serupa dengan yang berikut ini:

```
{
  "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```


Langkah 2: Buat tujuan

Important

Semua langkah dalam prosedur ini harus dilakukan di akun penerima data log.

Saat tujuan dibuat, CloudWatch Log mengirimkan pesan pengujian ke tujuan atas nama akun penerima. Saat filter langganan aktif nanti, CloudWatch Log mengirimkan peristiwa log ke tujuan atas nama akun sumber.

Untuk membuat tujuan

1. Tunggu hingga aliran Firehose yang Anda buat [Langkah 1: Buat aliran pengiriman Firehose](#) menjadi aktif. Anda dapat menggunakan perintah berikut untuk memeriksa StreamDescription.StreamStatusproperty.

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
```

Selain itu, perhatikan DeliveryStreamDescription. DeliveryStreamNilai ARN, karena Anda harus menggunakannya di langkah selanjutnya. Contoh output dari perintah ini:

```
{
  "DeliveryStreamDescription": {
    "DeliveryStreamName": "my-delivery-stream",
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-delivery-stream",
    "DeliveryStreamStatus": "ACTIVE",
    "DeliveryStreamEncryptionConfiguration": {
      "Status": "DISABLED"
    },
    "DeliveryStreamType": "DirectPut",
    "VersionId": "1",
    "CreateTimestamp": "2021-02-01T23:59:15.567000-08:00",
    "Destinations": [
      {
        "DestinationId": "destinationId-000000000001",
        "S3DestinationDescription": {
          "RoleARN": "arn:aws:iam::222222222222:role/FirehosetoS3Role",
          "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
          "BufferingHints": {
```

```

        "SizeInMBs": 5,
        "IntervalInSeconds": 300
    },
    "CompressionFormat": "UNCOMPRESSED",
    "EncryptionConfiguration": {
        "NoEncryptionConfig": "NoEncryption"
    },
    "CloudWatchLoggingOptions": {
        "Enabled": false
    }
},
"ExtendedS3DestinationDescription": {
    "RoleARN": "arn:aws:iam::222222222222:role/FirehoseToS3Role",
    "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
    "BufferingHints": {
        "SizeInMBs": 5,
        "IntervalInSeconds": 300
    },
    "CompressionFormat": "UNCOMPRESSED",
    "EncryptionConfiguration": {
        "NoEncryptionConfig": "NoEncryption"
    },
    "CloudWatchLoggingOptions": {
        "Enabled": false
    },
    "S3BackupMode": "Disabled"
}
    ],
    "HasMoreDestinations": false
}
}

```

Mungkin diperlukan satu atau dua menit bagi aliran pengiriman Anda untuk muncul dalam keadaan aktif.

2. Saat aliran pengiriman aktif, buat peran IAM yang akan memberikan izin kepada CloudWatch Log untuk memasukkan data ke aliran Firehose Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file `~/TrustPolicyFor CWL.json`. Gunakan editor teks untuk membuat kebijakan ini. Untuk informasi selengkapnya tentang titik akhir CloudWatch Log, lihat [titik akhir dan CloudWatch kuota Amazon Logs](#).

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global yang menentukan `sourceAccountId` untuk membantu mencegah masalah keamanan wakil yang membingungkan. Jika Anda belum mengetahui ID akun sumber pada panggilan pertama, kami sarankan Anda memasukkan ARN tujuan di bidang ARN sumber. Dalam panggilan berikutnya, Anda harus mengatur ARN sumber menjadi ARN sumber sebenarnya yang Anda kumpulkan dari panggilan pertama. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "logs.region.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringLike": {
        "aws:SourceArn": [
          "arn:aws:logs:region:sourceAccountId:",
          "arn:aws:logs:region:recipientAccountId:"
        ]
      }
    }
  }
}
```

- Gunakan perintah `aws iam create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan yang baru saja Anda buat.

```
aws iam create-role \
  --role-name CWLtoKinesisFirehoseRole \
  --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

Berikut ini adalah contoh output. Perhatikan nilai `Role.Arn` yang dikembalikan, karena Anda akan perlu menggunakannya di langkah berikutnya.

```
{
  "Role": {
    "Path": "/",
    "RoleName": "CWLtoKinesisFirehoseRole",
    "RoleId": "AR0AR3BXASEKYJYWF243H",
```

```

    "Arn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
    "CreateDate": "2021-02-02T08:10:43+00:00",
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.region.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringLike": {
            "aws:SourceArn": [
              "arn:aws:logs:region:sourceAccountId:*",
              "arn:aws:logs:region:recipientAccountId:"
            ]
          }
        }
      }
    }
  }
}

```

4. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file ~/PermissionsForCWL.json:

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["firehose:*"],
      "Resource": ["arn:aws:firehose:region:222222222222:*"]
    }
  ]
}

```

5. Kaitkan kebijakan izin dengan peran tersebut dengan memasukkan perintah berikut:

```

aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json

```

6. Setelah aliran pengiriman Firehose dalam status aktif dan Anda telah membuat peran IAM, Anda dapat membuat tujuan Log. CloudWatch

- a. Langkah ini tidak akan mengaitkan kebijakan akses dengan tujuan Anda dan hanya merupakan langkah pertama dari dua langkah yang akan menyelesaikan pembuatan tujuan. Catat ARN tujuan baru yang dikembalikan di payload, karena Anda akan menggunakan ini sebagai langkah `destination.arn` selanjutnya.

```
aws logs put-destination \

    --destination-name "testFirehoseDestination" \
    --target-arn "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-
delivery-stream" \
    --role-arn "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole"

{
  "destination": {
    "destinationName": "testFirehoseDestination",
    "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
    "roleArn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
    "arn": "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"}
}
```

- b. Setelah langkah sebelumnya selesai, dalam akun penerima data log (222222222222), kaitkan kebijakan akses dengan tujuan.

Kebijakan ini memungkinkan akun pengirim data log (11111111111111) untuk mengakses tujuan hanya di akun penerima data log (222222222222). Anda dapat menggunakan editor teks untuk meletakkan kebijakan ini di file `~/AccessPolicy.json`:

JSON

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : "logs:PutLogEvents"
```

```

    "Action" : "logs:PutSubscriptionFilter",
    "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
  }
]
}

```

- c. Ini membuat kebijakan yang menentukan siapa yang memiliki akses menulis ke tujuan. Kebijakan ini harus menentukan PutSubscriptionFilter tindakan log: untuk mengakses tujuan. Pengguna lintas akun akan menggunakan PutSubscriptionFilter tindakan untuk mengirim peristiwa log ke tujuan:

```

aws logs put-destination-policy \
  --destination-name "testFirehoseDestination" \
  --access-policy file://~/AccessPolicy.json

```

Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun

Menurut logika evaluasi kebijakan AWS lintas akun, untuk mengakses sumber daya lintas akun (seperti aliran Kinesis atau Firehose yang digunakan sebagai tujuan filter langganan), Anda harus memiliki kebijakan berbasis identitas di akun pengirim yang menyediakan akses eksplisit ke sumber tujuan lintas akun. Untuk informasi selengkapnya tentang logika evaluasi kebijakan, lihat [Logika evaluasi kebijakan lintas akun](#).

Anda dapat melampirkan kebijakan berbasis identitas ke peran IAM atau pengguna IAM yang Anda gunakan untuk membuat filter langganan. Kebijakan ini harus ada di akun pengiriman. Jika Anda menggunakan peran Administrator untuk membuat filter langganan, Anda dapat melewati langkah ini dan melanjutkan ke [Langkah 4: Buat filter berlangganan](#).

Untuk menambah atau memvalidasi izin IAM yang diperlukan untuk lintas akun

1. Masukkan perintah berikut untuk memeriksa peran IAM atau pengguna IAM mana yang digunakan untuk menjalankan perintah AWS log.

```
aws sts get-caller-identity
```

Perintah tersebut mengembalikan output serupa dengan berikut ini:

```
{
```

```
"UserId": "User ID",
"Account": "sending account id",
"Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

Catat nilai yang diwakili oleh *RoleName* atau *UserName*.

2. Konsol Manajemen AWS Masuk ke akun pengiriman dan cari kebijakan terlampir dengan peran IAM atau pengguna IAM yang dikembalikan dalam output perintah yang Anda masukkan pada langkah 1.
3. Verifikasi bahwa kebijakan yang dilampirkan pada peran ini atau pengguna memberikan izin eksplisit untuk memanggil sumber `logs:PutSubscriptionFilter` daya tujuan lintas akun.

Kebijakan berikut memberikan izin untuk membuat filter langganan pada sumber daya tujuan apa pun hanya dalam satu AWS akun, akun999999999999:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":
      "AllowSubscriptionFiltersOnAnyResourceInOneSpecificAccount",
      "Effect": "Allow",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": [
        "arn:aws:logs::*:log-group:*",
        "arn:aws:logs::*:123456789012:destination:*"
      ]
    }
  ]
}
```

Kebijakan berikut memberikan izin untuk membuat filter langganan hanya pada sumber daya tujuan tertentu yang dinamai `sampleDestination` dalam satu AWS akun, akun123456789012:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSubscriptionFiltersOnSpecificResource",
      "Effect": "Allow",
      "Action": "logs:PutSubscriptionFilter",
      "Resource": [
        "arn:aws:logs:*:*:log-group:*",
        "arn:aws:logs:*:123456789012:destination:amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Langkah 4: Buat filter berlangganan

Beralihlah ke akun pengiriman, yaitu 111111111111 dalam contoh ini. Sekarang Anda akan membuat filter langganan di akun pengirim. Dalam contoh ini, filter dikaitkan dengan grup log yang berisi AWS CloudTrail peristiwa sehingga setiap aktivitas yang dicatat yang dibuat oleh AWS kredensial “Root” dikirimkan ke tujuan yang sebelumnya Anda buat. Untuk informasi selengkapnya tentang cara mengirim AWS CloudTrail peristiwa ke CloudWatch Log, lihat [Mengirim CloudTrail Acara ke CloudWatch Log](#) di Panduan AWS CloudTrail Pengguna.

Ketika Anda memasukkan perintah berikut, pastikan Anda masuk sebagai pengguna IAM atau menggunakan peran IAM yang Anda tambahkan kebijakan untuk, in. [Langkah 3: Add/validate Izin IAM untuk tujuan lintas akun](#)

```
aws logs put-subscription-filter \
  --log-group-name "aws-cloudtrail-logs-111111111111-300a971e" \
  --filter-name "firehose_test" \
  --filter-pattern "${$.userIdentity.type = AssumedRole}" \
  --destination-arn "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
```

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, tujuan dapat menunjuk ke AWS sumber daya seperti aliran Firehose yang terletak di Wilayah yang berbeda.

Memvalidasi alur peristiwa log

Setelah Anda membuat filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter ke aliran pengiriman Firehose. Data mulai muncul di bucket Amazon S3 Anda berdasarkan interval buffer waktu yang disetel pada aliran pengiriman Firehose. Setelah waktu tertentu berlalu, Anda dapat memverifikasi data dengan memeriksa bucket Amazon S3. Untuk memeriksa bucket, masukkan perintah berikut:

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

Output perintah tersebut akan serupa dengan yang berikut ini:

```
{
  "Contents": [
    {
      "Key": "2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba",
      "LastModified": "2021-02-02T09:00:26+00:00",
      "ETag": "\"EXAMPLEa817fb88fc770b81c8f990d\"",
      "Size": 198,
      "StorageClass": "STANDARD",
      "Owner": {
        "DisplayName": "firehose+2test",
        "ID": "EXAMPLE27fd05889c665d2636218451970ef79400e3d2aecca3adb1930042e0"
      }
    }
  ]
}
```

Anda kemudian dapat mengambil objek tertentu dari bucket dengan memasukkan perintah berikut. Ganti nilai key dengan nilai yang Anda temukan di perintah sebelumnya.

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket' --key '2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

Data dalam objek Amazon S3 dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan salah satu dari perintah berikut:

Linux:

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

Memodifikasi keanggotaan tujuan saat runtime

Anda mungkin mengalami situasi ketika Anda harus menambahkan atau menghapus pengirim log dari tujuan yang Anda miliki. Anda dapat menggunakan `PutDestinationPolicy` tindakan di tujuan Anda dengan kebijakan akses baru. Dalam contoh berikut, akun 111111111111 yang ditambahkan sudah sebelumnya dihentikan dari mengirim data log lagi, dan akun 333333333333 diaktifkan.

1. Ambil kebijakan yang saat ini terkait dengan `TestDestination` tujuan dan catat: `AccessPolicy`

```
aws logs describe-destinations \
  --destination-name-prefix "testFirehoseDestination"

{
  "destinations": [
    {
      "destinationName": "testFirehoseDestination",
      "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
      "roleArn": "arn:aws:iam:: 222222222222:role/CWLtoKinesisFirehoseRole",
      "accessPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Statement
\" : [\n    {\n      \"Sid\" : \"\",\n      \"Effect\" : \"Allow\",\n
      \"Principal\" : {\n        \"AWS\" : \"111111111111 \"\n      },\n      \"Action
\" : \"logs:PutSubscriptionFilter\",\n      \"Resource\" : \"arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination\"\n    }\n  ]\n}\n\n",
      "arn": "arn:aws:logs:us-east-1:
222222222222:destination:testFirehoseDestination",
      "creationTime": 1612256124430
    }
  ]
}
```

2. Perbarui kebijakan agar menunjukkan bahwa akun 111111111111 dihentikan, dan akun 333333333333 diaktifkan. Letakkan kebijakan ini di file `~/NewAccessPolicy.json`:

JSON

```
{
```

```
"Version": "2012-10-17",
"Statement" : [
  {
    "Sid" : "",
    "Effect" : "Allow",
    "Principal" : {
      "AWS" : "333333333333 "
    },
    "Action" : "logs:PutSubscriptionFilter",
    "Resource" : "arn:aws:logs:us-east-1:222222222222:destination:testFirehoseDestination"
  }
]
```

3. Gunakan perintah berikut untuk mengaitkan kebijakan yang ditentukan dalam NewAccessPolicyfile.json dengan tujuan:

```
aws logs put-destination-policy \
  --destination-name "testFirehoseDestination" \
  --access-policy file://~/NewAccessPolicy.json
```

Ini akhirnya akan menonaktifkan log acara dari ID akun 111111111111. Log acara dari ID akun 333333333333 mulai mengalir ke tujuan segera setelah pemilik akun 333333333333 membuat filter langganan.

Langganan tingkat akun lintas wilayah lintas akun menggunakan Amazon Kinesis Data Streams

Saat membuat langganan lintas akun, Anda dapat menentukan satu akun atau organisasi untuk menjadi pengirim. Jika Anda menentukan organisasi, maka prosedur ini memungkinkan semua akun di organisasi untuk mengirim log ke akun penerima.

Untuk berbagi data log lintas akun, Anda perlu membuat pengirim dan penerima data log:

- Pengirim data log —mendapatkan informasi tujuan dari penerima dan memberi tahu CloudWatch Log bahwa Log siap mengirim peristiwa lognya ke tujuan yang ditentukan. Dalam prosedur di bagian lainnya, pengirim data log ditampilkan dengan nomor AWS akun fiksi 111111111111.

Jika Anda akan memiliki beberapa akun dalam satu organisasi yang mengirim log ke satu akun penerima, Anda dapat membuat kebijakan yang memberikan izin kepada semua akun di organisasi untuk mengirim log ke akun penerima. Anda masih harus menyiapkan filter langganan terpisah untuk setiap akun pengirim.

- Penerima data log —menyiapkan tujuan yang merangkum aliran Amazon Kinesis Data Streams dan CloudWatch memberi tahu Log bahwa penerima ingin menerima data log. Penerima kemudian membagikan informasi tentang tujuan ini dengan pengirim. Dalam prosedur di bagian lainnya, penerima data log ditampilkan dengan nomor AWS akun fiksi 999999999999.

Untuk mulai menerima peristiwa log dari pengguna lintas akun, penerima data log terlebih dahulu membuat tujuan CloudWatch Log. Setiap tujuan terdiri atas elemen kunci berikut:

Nama tujuan

Nama tujuan yang ingin Anda buat.

ARN Target

Nama Sumber Daya Amazon (ARN) dari AWS sumber daya yang ingin Anda gunakan sebagai tujuan umpan berlangganan.

Peran ARN

Peran AWS Identity and Access Management (IAM) yang memberikan CloudWatch Log izin yang diperlukan untuk memasukkan data ke aliran yang dipilih.

Kebijakan akses

Dokumen kebijakan IAM (dalam format JSON, ditulis menggunakan tata bahasa kebijakan IAM) yang mengatur set pengguna yang diizinkan untuk menulis ke tujuan Anda.

Note

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, AWS sumber daya yang ditunjuk tujuan dapat ditemukan di Wilayah yang berbeda. Dalam contoh di bagian berikut, semua sumber daya khusus Wilayah dibuat di US East (N. Virginia).

Topik

- [Menyiapkan langganan lintas akun baru](#)
- [Memperbarui langganan lintas akun yang ada](#)

Menyiapkan langganan lintas akun baru

Ikuti langkah-langkah di bagian ini untuk menyiapkan langganan log lintas akun baru.

Topik

- [Langkah 1: Buat tujuan](#)
- [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#)
- [Langkah 3: Buat kebijakan filter langganan tingkat akun](#)
- [Validasi alur peristiwa log](#)
- [Ubah keanggotaan tujuan saat runtime](#)

Langkah 1: Buat tujuan

Important

Semua langkah dalam prosedur ini harus dilakukan di akun penerima data log.

Untuk contoh ini, akun penerima data log memiliki ID akun 999999999999, sedangkan ID AWS akun pengirim AWS data log adalah 111111111111.

Contoh ini membuat tujuan menggunakan aliran Amazon Kinesis Data RecipientStream Streams yang disebut, dan peran CloudWatch yang memungkinkan Log untuk menulis data ke sana.

Saat tujuan dibuat, CloudWatch Log mengirimkan pesan pengujian ke tujuan atas nama akun penerima. Saat filter langganan aktif nanti, CloudWatch Log mengirimkan peristiwa log ke tujuan atas nama akun sumber.

Untuk membuat tujuan

1. Di akun penerima, buat aliran tujuan di Amazon Kinesis Data Streams. Di jendela perintah, ketik:

```
aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
```

2. Tunggu hingga streaming menjadi aktif. Anda dapat menggunakan perintah `aws kinesis describe-stream` untuk memeriksa `StreamDescription StreamStatus` properti. Selain itu, perhatikan `StreamDescription` nilai `streamName` karena Anda akan meneruskannya ke CloudWatch Log nanti:

```
aws kinesis describe-stream --stream-name "RecipientStream"
{
  "StreamDescription": {
    "StreamStatus": "ACTIVE",
    "StreamName": "RecipientStream",
    "StreamARN": "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream",
    "Shards": [
      {
        "ShardId": "shardId-000000000000",
        "HashKeyRange": {
          "EndingHashKey": "34028236692093846346337460743176EXAMPLE",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
            "4955113521868881845667950383198145878459135270218EXAMPLE"
        }
      }
    ]
  }
}
```

Mungkin diperlukan satu atau dua menit bagi pengaliran Anda untuk muncul dalam keadaan aktif.

3. Buat peran IAM yang memberikan izin kepada CloudWatch Log untuk memasukkan data ke aliran Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file `~/TrustPolicyForCWL.json`. Gunakan editor teks untuk membuat file kebijakan ini, jangan menggunakan konsol IAM.

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global yang menentukan `sourceAccountId` untuk membantu mencegah masalah keamanan wakil yang membingungkan. Jika Anda belum mengetahui ID akun sumber pada panggilan pertama, kami sarankan Anda memasukkan ARN tujuan di bidang ARN sumber. Dalam panggilan berikutnya, Anda harus mengatur ARN sumber menjadi ARN sumber sebenarnya yang Anda kumpulkan dari panggilan pertama. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "logs.amazonaws.com"
    },
    "Condition": {
      "StringLike": {
        "aws:SourceArn": [
          "arn:aws:logs:region:sourceAccountId:",
          "arn:aws:logs:region:recipientAccountId:"
        ]
      }
    },
    "Action": "sts:AssumeRole"
  }
}
```

4. Gunakan perintah `aws iam create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan. Perhatikan nilai `Role.Arn` yang dikembalikan karena itu juga akan diteruskan ke Log nanti: CloudWatch

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Condition": {
          "StringLike": {
            "aws:SourceArn": [
              "arn:aws:logs:region:sourceAccountId:",
              "arn:aws:logs:region:recipientAccountId:"
            ]
          }
        },
        "Principal": {
          "Service": "logs.amazonaws.com"
        }
      }
    }
  }
}
```

```

    }
  },
  "RoleId": "AA0IIAH450GAB4HC5F431",
  "CreateDate": "2023-05-29T13:46:29.431Z",
  "RoleName": "CWLtoKinesisRole",
  "Path": "/",
  "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
}
}

```

5. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file ~/PermissionsForCWL.json:

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}

```

6. Kaitkan kebijakan izin dengan peran dengan menggunakan perintah `aws iam: put-role-policy`

```

aws iam put-role-policy \
  --role-name CWLtoKinesisRole \
  --policy-name Permissions-Policy-For-CWL \
  --policy-document file:///~/PermissionsForCWL.json

```

7. Setelah aliran dalam keadaan aktif dan Anda telah membuat peran IAM, Anda dapat membuat tujuan CloudWatch Log.
 - a. Langkah ini tidak mengaitkan kebijakan akses dengan tujuan Anda dan hanya langkah pertama dari dua langkah yang menyelesaikan pembuatan tujuan. Catat DestinationArn yang dikembalikan dalam muatan:

```

aws logs put-destination \
  --destination-name "testDestination" \
  --target-arn "arn:aws:kinesis:region:999999999999:stream/RecipientStream" \

```



```
--role-arn "arn:aws:iam::999999999999:role/CWLtoKinesisRole"

{
  "DestinationName" : "testDestination",
  "RoleArn" : "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
  "DestinationArn" : "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
  "TargetArn" : "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream"
}
```

- b. Setelah langkah 7a selesai, di akun penerima data log, kaitkan kebijakan akses dengan tujuan. Kebijakan ini harus menentukan PutSubscriptionFilter tindakan log: dan memberikan izin ke akun pengirim untuk mengakses tujuan.

Kebijakan memberikan izin ke AWS akun yang mengirim log. Anda dapat menentukan hanya satu akun ini dalam kebijakan, atau jika akun pengirim adalah anggota organisasi, kebijakan dapat menentukan ID organisasi organisasi. Dengan cara ini, Anda dapat membuat hanya satu kebijakan untuk mengizinkan beberapa akun dalam satu organisasi mengirim log ke akun tujuan ini.

Gunakan editor teks untuk membuat file bernama ~/AccessPolicy.json dengan salah satu pernyataan kebijakan berikut.

Kebijakan contoh pertama ini memungkinkan semua akun di organisasi yang memiliki ID o-1234567890 untuk mengirim log ke akun penerima.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "logs:PutSubscriptionFilter",
        "logs:PutAccountPolicy"
      ],
      "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination",
```

```

        "Condition": {
            "StringEquals": {
                "aws:PrincipalOrgID": [
                    "o-1234567890"
                ]
            }
        }
    }
}

```

Contoh berikutnya ini memungkinkan hanya akun pengirim data log (111111111111) untuk mengirim log ke akun penerima data log.

JSON

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111111111111"
            },
            "Action": [
                "logs:PutSubscriptionFilter",
                "logs:PutAccountPolicy"
            ],
            "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination"
        }
    ]
}

```

- c. Lampirkan kebijakan yang Anda buat pada langkah sebelumnya ke tujuan.

```

aws logs put-destination-policy \
  --destination-name "testDestination" \
  --access-policy file://~/AccessPolicy.json

```

Kebijakan akses ini memungkinkan pengguna di AWS Akun dengan ID 111111111111 untuk memanggil tujuan dengan ARN `arn:aws:logs:PutSubscriptionFilter::999999999999:destination:testDestination`. *region* Upaya pengguna lain untuk menelepon PutSubscriptionFilter terhadap tujuan ini akan ditolak.

Untuk memvalidasi hak istimewa pengguna berdasarkan kebijakan akses, lihat [Menggunakan Validator Kebijakan](#) dalam Panduan Pengguna IAM.

Setelah selesai, jika Anda menggunakan AWS Organizations izin lintas akun, ikuti langkah-langkahnya. [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#) Jika Anda memberikan izin langsung ke akun lain alih-alih menggunakan Organizations, Anda dapat melewati langkah itu dan melanjutkan ke. [Langkah 3: Buat kebijakan filter langganan tingkat akun](#)

Langkah 2: (Hanya jika menggunakan organisasi) Buat peran IAM

Di bagian sebelumnya, jika Anda membuat tujuan menggunakan kebijakan akses yang memberikan izin kepada organisasi tempat akun 111111111111 berada, alih-alih memberikan izin langsung ke akun111111111111, ikuti langkah-langkah di bagian ini. Jika tidak, Anda dapat melompat ke [Langkah 3: Buat kebijakan filter langganan tingkat akun](#).

Langkah-langkah di bagian ini membuat peran IAM, yang CloudWatch dapat mengasumsikan dan memvalidasi apakah akun pengirim memiliki izin untuk membuat filter langganan terhadap tujuan penerima.

Lakukan langkah-langkah di bagian ini di akun pengirim. Peran harus ada di akun pengirim, dan Anda menentukan ARN peran ini dalam filter berlangganan. Dalam contoh ini, akun pengirim adalah111111111111.

Untuk membuat peran IAM yang diperlukan untuk langganan log lintas akun menggunakan AWS Organizations

1. Buat kebijakan kepercayaan berikut dalam sebuah file/`TrustPolicyForCWLSubscriptionFilter.json`. Gunakan editor teks untuk membuat file kebijakan ini; jangan gunakan konsol IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
```

```
"Action": "sts:AssumeRole"
}
}
```

2. Buat peran IAM yang menggunakan kebijakan ini. Perhatikan Arn nilai yang dikembalikan oleh perintah, Anda akan membutuhkannya nanti dalam prosedur ini. Dalam contoh ini, kita gunakan `CWLtoSubscriptionFilterRole` untuk nama peran yang kita buat.

```
aws iam create-role \
  --role-name CWLtoSubscriptionFilterRole \
  --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda.
 - a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `~/PermissionsForCWLSubscriptionFilter.json`.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "logs:PutLogEvents",
      "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
    }
  ]
}
```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran yang Anda buat di langkah 2.

```
aws iam put-role-policy
  --role-name CWLtoSubscriptionFilterRole
  --policy-name Permissions-Policy-For-CWL-Subscription-filter
  --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

Setelah selesai, Anda dapat melanjutkan ke [Langkah 3: Buat kebijakan filter langganan tingkat akun](#).

Langkah 3: Buat kebijakan filter langganan tingkat akun

Setelah Anda membuat tujuan, akun penerima data log dapat berbagi ARN tujuan (arn:aws:logs:us-east-1:999999999999:destination:testDestination) dengan akun AWS lain sehingga mereka dapat mengirim log acara ke tujuan yang sama. Para pengguna akun pengirim ini kemudian membuat filter langganan pada grup log masing-masing berdasarkan tujuan ini. Filter langganan segera memulai aliran data log waktu nyata dari grup log yang dipilih ke tujuan yang ditentukan.

Note

Jika Anda memberikan izin untuk filter langganan ke seluruh organisasi, Anda harus menggunakan ARN dari peran IAM yang Anda buat. [Langkah 2: \(Hanya jika menggunakan organisasi\) Buat peran IAM](#)

Dalam contoh berikut, kebijakan filter langganan tingkat akun dibuat di akun pengiriman. filter dikaitkan dengan akun pengirim 111111111111 sehingga setiap peristiwa log yang cocok dengan filter dan kriteria pemilihan dikirim ke tujuan yang Anda buat sebelumnya. Tujuan itu merangkum aliran yang disebut "". RecipientStream

selection-criteriaBidang ini opsional, tetapi penting untuk mengecualikan grup log yang dapat menyebabkan rekursi log tak terbatas dari filter langganan. Untuk informasi selengkapnya tentang masalah ini dan menentukan grup log mana yang akan dikecualikan, lihat [Pencegahan rekursi log](#). Saat ini, NOT IN adalah satu-satunya operator yang didukung untuk selection-criteria.

```
aws logs put-account-policy \
  --policy-name "CrossAccountStreamsExamplePolicy" \
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \
  --policy-document
'{"DestinationArn":"arn:aws:logs:region:999999999999:destination:testDestination",
"FilterPattern": "", "Distribution": "Random"}' \
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
"LogGroupToExclude2"]' \
  --scope "ALL"
```

Grup log akun pengirim dan tujuan harus berada di AWS Wilayah yang sama. Namun, tujuan dapat menunjuk ke AWS sumber daya seperti aliran Amazon Kinesis Data Streams yang terletak di Wilayah yang berbeda.

Validasi alur peristiwa log

Setelah Anda membuat kebijakan filter langganan tingkat akun, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter dan kriteria pemilihan ke aliran yang dienkapsulasi dalam aliran tujuan yang disebut "". RecipientStream Pemilik tujuan dapat memverifikasi bahwa ini terjadi dengan menggunakan `get-shard-iterator` perintah `aws kinesis` untuk mengambil pecahan Amazon Kinesis Data Streams, dan menggunakan perintah `aws kinesis get-records` untuk mengambil beberapa catatan Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator \
    --stream-name RecipientStream \
    --shard-id shardId-000000000000 \
    --shard-iterator-type TRIM_HORIZON

{
  "ShardIterator":
  "AAAAAAAAAAGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+0IK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
}

aws kinesis get-records \
    --limit 10 \
    --shard-iterator
    "AAAAAAAAAAGU/
kLvNggvndHq2UIF0w5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3Igvq0b5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+0IK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

Note

Anda mungkin perlu menjalankan kembali `get-records` perintah beberapa kali sebelum Amazon Kinesis Data Streams mulai mengembalikan data.

Anda akan melihat respons dengan larik catatan Amazon Kinesis Data Streams. Atribut data dalam catatan Amazon Kinesis Data Streams dikompresi dalam format gzip dan kemudian base64 dikodekan. Anda dapat memeriksa data mentah dari baris perintah menggunakan perintah Unix berikut:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Data yang didekode dan didekompresi base64 diformat sebagai JSON dengan struktur berikut:

```
{
  "owner": "111111111111",
  "logGroup": "CloudTrail/logs",
  "logStream": "111111111111_CloudTrail/logs_us-east-1",
  "subscriptionFilters": [
    "RecipientStream"
  ],
  "messageType": "DATA_MESSAGE",
  "logEvents": [
    {
      "id": "3195310660696698337880902507980421114328961542429EXAMPLE",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
    }",
    },
    {
      "id": "3195310660696698337880902507980421114328961542429EXAMPLE",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
    }",
    },
    {
      "id": "3195310660696698337880902507980421114328961542429EXAMPLE",
      "timestamp": 1432826855000,
      "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
    }",
  ]
}
```

Elemen kunci dalam struktur data adalah sebagai berikut:

messageType

Pesan data akan menggunakan tipe "DATA_MESSAGE". Terkadang CloudWatch Log mungkin memancarkan catatan Amazon Kinesis Data Streams dengan tipe "CONTROL_MESSAGE", terutama untuk memeriksa apakah tujuan dapat dijangkau.

owner

ID AWS Akun dari data log asal.

logGroup

Nama grup log dari data log asal.

logStream

Nama pengaliran log dari data log asal.

subscriptionFilters

Daftar nama filter langganan yang cocok dengan data log asal.

logEvents

Data log yang sebenarnya, direpresentasikan sebagai array catatan log acara. Properti "id" adalah pengenal unik untuk setiap log acara.

PolicyLevel

Tingkat di mana kebijakan itu ditegakkan. "ACCOUNT_LEVEL_POLICY" adalah `policyLevel` untuk kebijakan filter langganan tingkat akun.

Ubah keanggotaan tujuan saat runtime

Anda mungkin mengalami situasi ketika Anda harus menambahkan atau menghapus keanggotaan beberapa pengguna dari tujuan yang Anda miliki. Anda dapat menggunakan perintah `put-destination-policy` di tujuan Anda dengan kebijakan akses baru. Dalam contoh berikut, akun 111111111111 yang ditambahkan sudah sebelumnya dihentikan dari mengirim data log lagi, dan akun 222222222222 diaktifkan.

1. Ambil kebijakan yang saat ini terkait dengan `TestDestination` tujuan dan catat: `AccessPolicy`

```
aws logs describe-destinations \
  --destination-name-prefix "testDestination"

{
  "Destinations": [
    {
      "DestinationName": "testDestination",
```



```

    "RoleArn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole",
    "DestinationArn":
    "arn:aws:logs:region:999999999999:destination:testDestination",
    "TargetArn": "arn:aws:kinesis:region:999999999999:stream/RecipientStream",
    "AccessPolicy": "{ \"Version\": \"2012-10-17\", \"Statement\":
    [{ \"Sid\": \"\", \"Effect\": \"Allow\", \"Principal\": { \"AWS\":
    \"111111111111\" }, \"Action\": \"logs:PutSubscriptionFilter\", \"Resource\":
    \"arn:aws:logs:region:999999999999:destination:testDestination\" } ] }"
  }
]
}

```

2. Perbarui kebijakan agar menunjukkan bahwa akun 111111111111 dihentikan, dan akun 222222222222 diaktifkan. Letakkan kebijakan ini di file ~/NewAccessPolicy.json:

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "222222222222"
      },
      "Action": [
        "logs:PutSubscriptionFilter",
        "logs:PutAccountPolicy"
      ],
      "Resource": "arn:aws:logs:us-
east-1:999999999999:destination:testDestination"
    }
  ]
}

```

3. Panggilan PutDestinationPolicy untuk mengaitkan kebijakan yang ditentukan dalam NewAccessPolicyfile.json dengan tujuan:

```

aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json

```

Ini pada akhirnya akan menonaktifkan log acara dari ID akun 111111111111. Log acara dari ID akun 222222222222 mulai mengalir ke tujuan segera setelah pemilik akun 222222222222 membuat filter langganan.

Memperbarui langganan lintas akun yang ada

Jika saat ini Anda memiliki langganan log lintas akun di mana akun tujuan hanya memberikan izin ke akun pengirim tertentu, dan Anda ingin memperbarui langganan ini sehingga akun tujuan memberikan akses ke semua akun di organisasi, ikuti langkah-langkah di bagian ini.

Topik

- [Langkah 1: Perbarui filter berlangganan](#)
- [Langkah 2: Perbarui kebijakan akses tujuan yang ada](#)

Langkah 1: Perbarui filter berlangganan

Note

Langkah ini diperlukan hanya untuk langganan lintas akun untuk log yang dibuat oleh layanan yang tercantum di [Aktifkan logging dari AWS layanan](#). Jika Anda tidak bekerja dengan log yang dibuat oleh salah satu grup log ini, Anda dapat melompat ke [Langkah 2: Perbarui kebijakan akses tujuan yang ada](#).

Dalam kasus tertentu, Anda harus memperbarui filter langganan di semua akun pengirim yang mengirim log ke akun tujuan. Pembaruan menambahkan peran IAM, yang CloudWatch dapat mengasumsikan dan memvalidasi bahwa akun pengirim memiliki izin untuk mengirim log ke akun penerima.

Ikuti langkah-langkah di bagian ini untuk setiap akun pengirim yang ingin Anda perbarui untuk menggunakan ID organisasi untuk izin berlangganan lintas akun.

Dalam contoh di bagian ini, dua akun, 111111111111 dan 222222222222 sudah memiliki filter berlangganan yang dibuat untuk mengirim log ke akun 999999999999. Nilai filter langganan yang ada adalah sebagai berikut:

```
## Existing Subscription Filter parameter values
```

```
{
  "DestinationArn": "arn:aws:logs:region:999999999999:destination:testDestination",
  "FilterPattern": "{$.userIdentity.type = Root}",
  "Distribution": "Random"
}
```

Jika Anda perlu menemukan nilai parameter filter langganan saat ini, masukkan perintah berikut.

```
aws logs describe-account-policies \
--policy-type "SUBSCRIPTION_FILTER_POLICY" \
--policy-name "CrossAccountStreamsExamplePolicy"
```

Untuk memperbarui filter langganan untuk mulai menggunakan organisasi IDs untuk izin log lintas akun

1. Buat kebijakan kepercayaan berikut dalam sebuah file `~/TrustPolicyForCWL.json`. Gunakan editor teks untuk membuat file kebijakan ini; jangan gunakan konsol IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. Buat peran IAM yang menggunakan kebijakan ini. Perhatikan nilai `Arn` nilai yang dikembalikan oleh perintah, Anda akan membutuhkannya nanti dalam prosedur ini. Dalam contoh ini, kita gunakan `CWLtoSubscriptionFilterRole` untuk nama peran yang kita buat.

```
aws iam create-role
\ --role-name CWLtoSubscriptionFilterRole
\ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda.
 - a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `PermissionsForCWLSubscriptionFilter.json`.

```
{
```

```

    "Statement": [
      {
        "Effect": "Allow",
        "Action": "logs:PutLogEvents",
        "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
      }
    ]
  }
}

```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran yang Anda buat di langkah 2.

```

aws iam put-role-policy
  --role-name CWLtoSubscriptionFilterRole
  --policy-name Permissions-Policy-For-CWL-Subscription-filter
  --policy-document file://~/PermissionsForCWLSubscriptionFilter.json

```

4. Masukkan perintah berikut untuk memperbarui kebijakan filter langganan.

```

aws logs put-account-policy \
  --policy-name "CrossAccountStreamsExamplePolicy" \
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \
  --policy-document
'{"DestinationArn":"arn:aws:logs:region:999999999999:destination:testDestination",
"FilterPattern": "${$.userIdentity.type = Root}", "Distribution": "Random"}' \
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",
"LogGroupToExclude2"]' \
  --scope "ALL"

```

Langkah 2: Perbarui kebijakan akses tujuan yang ada

Setelah memperbarui filter langganan di semua akun pengirim, Anda dapat memperbarui kebijakan akses tujuan di akun penerima.

Dalam contoh berikut, akun penerima adalah 999999999999 dan tujuan diberi nama `testDestination`.

Pembaruan memungkinkan semua akun yang merupakan bagian dari organisasi dengan ID `o-1234567890` untuk mengirim log ke akun penerima. Hanya akun yang memiliki filter langganan yang dibuat yang benar-benar akan mengirim log ke akun penerima.

Untuk memperbarui kebijakan akses tujuan di akun penerima untuk mulai menggunakan ID organisasi untuk izin

1. Di akun penerima, gunakan editor teks untuk membuat `~/AccessPolicy.json` file dengan konten berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "logs:PutSubscriptionFilter",
        "logs:PutAccountPolicy"
      ],
      "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": [
            "o-1234567890"
          ]
        }
      }
    }
  ]
}
```

2. Masukkan perintah berikut untuk melampirkan kebijakan yang baru saja Anda buat ke tujuan yang ada. Untuk memperbarui tujuan agar menggunakan kebijakan akses dengan ID organisasi, bukan kebijakan akses yang mencantumkan AWS akun tertentu IDs, sertakan `force` parameternya.

 Warning

Jika Anda bekerja dengan log yang dikirim oleh AWS layanan yang terdaftar di [Aktifkan logging dari AWS layanan](#), maka sebelum melakukan langkah ini, Anda harus terlebih

dahulu memperbarui filter langganan di semua akun pengirim seperti yang dijelaskan di [Langkah 1: Perbarui filter berlangganan](#).

```
aws logs put-destination-policy
\ --destination-name "testDestination"
\ --access-policy file://~/AccessPolicy.json
\ --force
```

Langganan tingkat akun lintas wilayah lintas akun menggunakan Firehose

Untuk berbagi data log lintas akun, Anda perlu membuat pengirim dan penerima data log:

- Pengirim data log —mendapatkan informasi tujuan dari penerima dan memberi tahu CloudWatch Log bahwa ia siap untuk mengirim peristiwa lognya ke tujuan yang ditentukan. Dalam prosedur di bagian lainnya, pengirim data log ditampilkan dengan nomor AWS akun fiksi 111111111111.
- Penerima data log —menyiapkan tujuan yang merangkum aliran Amazon Kinesis Data Streams dan CloudWatch memberi tahu Log bahwa penerima ingin menerima data log. Penerima kemudian membagikan informasi tentang tujuan ini dengan pengirim. Dalam prosedur di bagian lainnya, penerima data log ditampilkan dengan nomor AWS akun fiksi 222222222222.

Contoh di bagian ini menggunakan aliran pengiriman Firehose dengan penyimpanan Amazon S3. Anda juga dapat mengatur aliran pengiriman Firehose dengan pengaturan berbeda. Untuk informasi selengkapnya, lihat [Membuat Aliran Pengiriman Firehose](#).

Note

Grup log dan tujuan harus berada di AWS Wilayah yang sama. Namun, AWS sumber daya yang ditunjuk tujuan dapat ditemukan di Wilayah yang berbeda.

Note

Filter langganan Firehose untuk akun yang sama dan aliran pengiriman lintas wilayah didukung.

Topik

- [Langkah 1: Buat aliran pengiriman Firehose](#)
- [Langkah 2: Buat tujuan](#)
- [Langkah 3: Buat kebijakan filter langganan tingkat akun](#)
- [Memvalidasi alur peristiwa log](#)
- [Memodifikasi keanggotaan tujuan saat runtime](#)

Langkah 1: Buat aliran pengiriman Firehose

Important

Sebelum Anda menyelesaikan langkah-langkah berikut, Anda harus menggunakan kebijakan akses, sehingga Firehose dapat mengakses bucket Amazon S3 Anda. Untuk informasi selengkapnya, lihat [Mengontrol Akses](#) di Panduan Pengembang Amazon Data Firehose. Semua langkah di bagian ini (Langkah 1) harus dilakukan di akun penerima data log. US East (N. Virginia) digunakan dalam contoh perintah berikut. Ganti Wilayah ini dengan Wilayah yang benar untuk penerapan Anda.

Untuk membuat aliran pengiriman Firehose yang akan digunakan sebagai tujuan

1. Buat bucket Amazon S3:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
LocationConstraint=us-east-1
```

2. Buat peran IAM yang memberikan izin Firehose untuk memasukkan data ke dalam bucket.

- a. Pertama, gunakan editor teks untuk membuat kebijakan kepercayaan dalam file `~/TrustPolicyForFirehose.json`.

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
"firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
{ "StringEquals": { "sts:ExternalId": "222222222222" } } } }
```

- b. Buat IAM role dengan menentukan file kebijakan kepercayaan yang baru saja Anda buat.

```
aws iam create-role \
```

```
--role-name FirehoseToS3Role \
--assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

- c. Output perintah ini akan terlihat serupa dengan yang berikut ini. Catat nama peran dan ARN peran.

```
{
  "Role": {
    "Path": "/",
    "RoleName": "FirehoseToS3Role",
    "RoleId": "AR0AR3BXASEKW7K635M53",
    "Arn": "arn:aws:iam::222222222222:role/FirehoseToS3Role",
    "CreateDate": "2021-02-02T07:53:10+00:00",
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Effect": "Allow",
        "Principal": {
          "Service": "firehose.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
          "StringEquals": {
            "sts:ExternalId": "222222222222"
          }
        }
      }
    }
  }
}
```

3. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan Firehose di akun Anda.
- a. Pertama, gunakan editor teks untuk membuat kebijakan izin berikut dalam file bernama `~/PermissionsForFirehose.json`. Bergantung pada kasus penggunaan Anda, Anda mungkin perlu menambahkan lebih banyak izin ke file ini.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "s3:PutObject",
      "s3:PutObjectAcl",
      "s3:ListBucket"
    ]
  }]
```



```

    ],
    "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }]
}

```

- b. Masukkan perintah berikut untuk mengaitkan kebijakan izin yang baru saja Anda buat dengan peran IAM.

```

aws iam put-role-policy --role-name FirehoseToS3Role --policy-name
Permissions-Policy-For-Firehose-To-S3 --policy-document file:///~/
PermissionsForFirehose.json

```

4. Masukkan perintah berikut untuk membuat aliran pengiriman Firehose. Ganti *my-role-arn* dan *amzn-s3-demo-bucket2-arn* dengan nilai yang benar untuk penerapan Anda.

```

aws firehose create-delivery-stream \
  --delivery-stream-name 'my-delivery-stream' \
  --s3-destination-configuration \
  '{"RoleARN": "arn:aws:iam::222222222222:role/FirehoseToS3Role", "BucketARN":
  "arn:aws:s3:::amzn-s3-demo-bucket"}'

```

Outputnya akan serupa dengan yang berikut ini:

```

{
  "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}

```

Langkah 2: Buat tujuan

Important

Semua langkah dalam prosedur ini harus dilakukan di akun penerima data log.

Saat tujuan dibuat, CloudWatch Log mengirimkan pesan pengujian ke tujuan atas nama akun penerima. Saat filter langganan aktif nanti, CloudWatch Log mengirimkan peristiwa log ke tujuan atas nama akun sumber.

Untuk membuat tujuan

1. Tunggu hingga aliran Firehose yang Anda buat [Langkah 1: Buat aliran pengiriman Firehose](#) menjadi aktif. Anda dapat menggunakan perintah berikut untuk memeriksa StreamDescription. StreamStatusproperty.

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
```

Selain itu, perhatikan DeliveryStreamDescription. DeliveryStreamNilai ARN, karena Anda harus menggunakannya di langkah selanjutnya. Contoh output dari perintah ini:

```
{
  "DeliveryStreamDescription": {
    "DeliveryStreamName": "my-delivery-stream",
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-delivery-stream",
    "DeliveryStreamStatus": "ACTIVE",
    "DeliveryStreamEncryptionConfiguration": {
      "Status": "DISABLED"
    },
    "DeliveryStreamType": "DirectPut",
    "VersionId": "1",
    "CreateTimestamp": "2021-02-01T23:59:15.567000-08:00",
    "Destinations": [
      {
        "DestinationId": "destinationId-000000000001",
        "S3DestinationDescription": {
          "RoleARN": "arn:aws:iam::222222222222:role/FirehoseToS3Role",
          "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
          "BufferingHints": {
            "SizeInMBs": 5,
            "IntervalInSeconds": 300
          },
          "CompressionFormat": "UNCOMPRESSED",
          "EncryptionConfiguration": {
            "NoEncryptionConfig": "NoEncryption"
          },
          "CloudWatchLoggingOptions": {
```

```

        "Enabled": false
      },
    },
    "ExtendedS3DestinationDescription": {
      "RoleARN": "arn:aws:iam::222222222222:role/FirehoseToS3Role",
      "BucketARN": "arn:aws:s3:::amzn-s3-demo-bucket",
      "BufferingHints": {
        "SizeInMBs": 5,
        "IntervalInSeconds": 300
      },
      "CompressionFormat": "UNCOMPRESSED",
      "EncryptionConfiguration": {
        "NoEncryptionConfig": "NoEncryption"
      },
      "CloudWatchLoggingOptions": {
        "Enabled": false
      },
      "S3BackupMode": "Disabled"
    }
  ],
  "HasMoreDestinations": false
}

```

Mungkin diperlukan satu atau dua menit bagi aliran pengiriman Anda untuk muncul dalam keadaan aktif.

2. Saat aliran pengiriman aktif, buat peran IAM yang akan memberikan izin kepada CloudWatch Log untuk memasukkan data ke aliran Firehose Anda. Pertama, Anda harus membuat kebijakan kepercayaan dalam file `~/TrustPolicyForCWL.json`. Gunakan editor teks untuk membuat kebijakan ini. Untuk informasi selengkapnya tentang titik akhir CloudWatch Log, lihat [titik akhir dan CloudWatch kuota Amazon Logs](#).

Kebijakan ini mencakup kunci konteks kondisi `aws:SourceArn` global yang menentukan `sourceAccountId` untuk membantu mencegah masalah keamanan wakil yang membingungkan. Jika Anda belum mengetahui ID akun sumber pada panggilan pertama, kami sarankan Anda memasukkan ARN tujuan di bidang ARN sumber. Dalam panggilan berikutnya, Anda harus mengatur ARN sumber menjadi ARN sumber sebenarnya yang Anda kumpulkan dari panggilan pertama. Untuk informasi selengkapnya, lihat [Pencegahan Deputi Bingung](#).

```
{
```

```

    "Statement": {
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringLike": {
          "aws:SourceArn": [
            "arn:aws:logs:region:sourceAccountId:*",
            "arn:aws:logs:region:recipientAccountId:*"
          ]
        }
      }
    }
  }
}

```

- Gunakan perintah `aws iam create-role` untuk membuat IAM role, dengan menentukan file kebijakan kepercayaan yang baru saja Anda buat.

```

aws iam create-role \
  --role-name CWLtoKinesisFirehoseRole \
  --assume-role-policy-document file://~/TrustPolicyForCWL.json

```

Berikut ini adalah contoh output. Perhatikan nilai `Role.Arn` yang dikembalikan, karena Anda akan perlu menggunakannya di langkah berikutnya.

```

{
  "Role": {
    "Path": "/",
    "RoleName": "CWLtoKinesisFirehoseRole",
    "RoleId": "AROAR3BXASEKYJYWF243H",
    "Arn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
    "CreateDate": "2023-02-02T08:10:43+00:00",
    "AssumeRolePolicyDocument": {
      "Statement": {
        "Effect": "Allow",
        "Principal": {
          "Service": "logs.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {

```

```

        "StringLike": {
            "aws:SourceArn": [
                "arn:aws:logs:region:sourceAccountId:*",
                "arn:aws:logs:region:recipientAccountId:"
            ]
        }
    }
}

```

4. Buat kebijakan izin untuk menentukan tindakan yang dapat dilakukan CloudWatch Log di akun Anda. Pertama, gunakan editor teks untuk membuat kebijakan izin dalam file ~/PermissionsForCWL.json:

```

{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["firehose:*"],
      "Resource": ["arn:aws:firehose:region:222222222222:*"]
    }
  ]
}

```

5. Kaitkan kebijakan izin dengan peran tersebut dengan memasukkan perintah berikut:

```

aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json

```

6. Setelah aliran pengiriman Firehose dalam status aktif dan Anda telah membuat peran IAM, Anda dapat membuat tujuan Log. CloudWatch
 - a. Langkah ini tidak akan mengaitkan kebijakan akses dengan tujuan Anda dan hanya merupakan langkah pertama dari dua langkah yang akan menyelesaikan pembuatan tujuan. Catat ARN tujuan baru yang dikembalikan di payload, karena Anda akan menggunakan ini sebagai langkah `destination.arn` selanjutnya.

```

aws logs put-destination \

    --destination-name "testFirehoseDestination" \

```

```
--target-arn "arn:aws:firehose:us-east-1:222222222222:deliverystream/my-
delivery-stream" \
--role-arn "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole"

{
  "destination": {
    "destinationName": "testFirehoseDestination",
    "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
    "roleArn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
    "arn": "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"}
}
```

- b. Setelah langkah sebelumnya selesai, dalam akun penerima data log (222222222222), kaitkan kebijakan akses dengan tujuan. Kebijakan ini memungkinkan akun pengirim data log (111111111111) untuk mengakses tujuan hanya di akun penerima data log (222222222222). Anda dapat menggunakan editor teks untuk memasukkan kebijakan ini ke dalam `~/AccessPolicy.json` file:

JSON

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : ["logs:PutSubscriptionFilter", "logs:PutAccountPolicy"],
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

- c. Ini membuat kebijakan yang menentukan siapa yang memiliki akses menulis ke tujuan. Kebijakan ini harus menentukan `logs:PutSubscriptionFilter` dan `logs:PutAccountPolicy` tindakan untuk mengakses tujuan. Pengguna lintas akun

akan menggunakan PutAccountPolicy tindakan PutSubscriptionFilter dan untuk mengirim peristiwa log ke tujuan.

```
aws logs put-destination-policy \  
  --destination-name "testFirehoseDestination" \  
  --access-policy file://~/AccessPolicy.json
```

Langkah 3: Buat kebijakan filter langganan tingkat akun

Beralihlah ke akun pengiriman, yaitu 111111111111 dalam contoh ini. Anda sekarang akan membuat kebijakan filter langganan tingkat akun di akun pengiriman. Dalam contoh ini, filter menyebabkan setiap peristiwa log yang berisi string ERROR di semua kecuali dua grup log dikirim ke tujuan yang sebelumnya Anda buat.

```
aws logs put-account-policy \  
  --policy-name "CrossAccountFirehoseExamplePolicy" \  
  --policy-type "SUBSCRIPTION_FILTER_POLICY" \  
  --policy-document '{"DestinationArn":"arn:aws:logs:us-  
east-1:222222222222:destination:testFirehoseDestination", "FilterPattern":  
"$$.userIdentity.type = AssumedRole}", "Distribution": "Random"}' \  
  --selection-criteria 'LogGroupName NOT IN ["LogGroupToExclude1",  
"LogGroupToExclude2"]' \  
  --scope "ALL"
```

Grup log akun pengirim dan tujuan harus berada di AWS Wilayah yang sama. Namun, tujuan dapat menunjuk ke AWS sumber daya seperti aliran Firehose yang terletak di Wilayah yang berbeda.

Memvalidasi alur peristiwa log

Setelah Anda membuat filter langganan, CloudWatch Log meneruskan semua peristiwa log masuk yang cocok dengan pola filter dan kriteria pemilihan ke aliran pengiriman Firehose. Data mulai muncul di bucket Amazon S3 Anda berdasarkan interval buffer waktu yang disetel pada aliran pengiriman Firehose. Setelah waktu tertentu berlalu, Anda dapat memverifikasi data dengan memeriksa bucket Amazon S3. Untuk memeriksa bucket, masukkan perintah berikut:

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

Output perintah tersebut akan serupa dengan yang berikut ini:

```
{
  "Contents": [
    {
      "Key": "2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba",
      "LastModified": "2023-02-02T09:00:26+00:00",
      "ETag": "\"EXAMPLEa817fb88fc770b81c8f990d\"",
      "Size": 198,
      "StorageClass": "STANDARD",
      "Owner": {
        "DisplayName": "firehose+2test",
        "ID": "EXAMPLE27fd05889c665d2636218451970ef79400e3d2aecca3adb1930042e0"
      }
    }
  ]
}
```

Anda kemudian dapat mengambil objek tertentu dari bucket dengan memasukkan perintah berikut. Ganti nilai key dengan nilai yang Anda temukan di perintah sebelumnya.

```
aws s3api get-object --bucket 'amzn-s3-demo-bucket' --key '2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

Data dalam objek Amazon S3 dikompresi dengan format gzip. Anda dapat memeriksa data mentah dari baris perintah menggunakan salah satu dari perintah berikut:

Linux:

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

Memodifikasi keanggotaan tujuan saat runtime

Anda mungkin mengalami situasi ketika Anda harus menambahkan atau menghapus pengirim log dari tujuan yang Anda miliki. Anda dapat menggunakan `PutDestinationPolicy` dan `PutAccountPolicy` tindakan di tujuan Anda dengan kebijakan akses baru. Dalam contoh berikut,

akun 111111111111 yang ditambahkan sudah sebelumnya dihentikan dari mengirim data log lagi, dan akun 333333333333 diaktifkan.

1. Ambil kebijakan yang saat ini terkait dengan TestDestination tujuan dan catat: AccessPolicy

```
aws logs describe-destinations \
  --destination-name-prefix "testFirehoseDestination"
```

Data yang dikembalikan mungkin terlihat seperti ini.

```
{
  "destinations": [
    {
      "destinationName": "testFirehoseDestination",
      "targetArn": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream",
      "roleArn": "arn:aws:iam:: 222222222222:role/CWLtoKinesisFirehoseRole",
      "accessPolicy": "{\n  \"Version\" : \"2012-10-17\",\n  \"Statement
\" : [\n    {\n      \"Sid\" : \"\",\n      \"Effect\" : \"Allow\",\n
      \"Principal\" : {\n        \"AWS\" : \"111111111111 \"\n      },\n      \"Action
\" : \"logs:PutSubscriptionFilter\",\n      \"Resource\" : \"arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination\"\n    }\n  ]\n}\n\n",
      "arn": "arn:aws:logs:us-east-1:
222222222222:destination:testFirehoseDestination",
      "creationTime": 1612256124430
    }
  ]
}
```

2. Perbarui kebijakan agar menunjukkan bahwa akun 111111111111 dihentikan, dan akun 333333333333 diaktifkan. Letakkan kebijakan ini di file ~/ NewAccessPolicy .json:

JSON

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
```

```

    "AWS" : "333333333333 "
  },
  "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
  "Resource" : "arn:aws:logs:us-east-1:222222222222:destination:testFirehoseDestination"
}
]
}

```

- Gunakan perintah berikut untuk mengaitkan kebijakan yang ditentukan dalam `NewAccessPolicyfile.json` dengan tujuan:

```

aws logs put-destination-policy \
  --destination-name "testFirehoseDestination" \

  --access-policy file://~/NewAccessPolicy.json

```

Ini akhirnya akan menonaktifkan log acara dari ID akun 111111111111. Log acara dari ID akun 333333333333 mulai mengalir ke tujuan segera setelah pemilik akun 333333333333 membuat filter langganan.

Pencegahan Deputi Bingung

Masalah "confused deputy" adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memengaruhi entitas yang memiliki hak akses lebih tinggi untuk melakukan tindakan. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi data Anda untuk semua layanan dengan prinsip layanan yang telah diberikan akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi

[aws:SourceArns](#), [aws:SourceAccount](#), [aws:SourceOrgID](#), dan [aws:SourceOrgPaths](#) global dalam kebijakan sumber daya untuk membatasi izin yang memberikan layanan lain ke sumber daya. Gunakan `aws:SourceArn` untuk mengaitkan hanya satu sumber daya dengan akses lintas layanan. Gunakan `aws:SourceAccount` untuk membiarkan sumber daya apa pun di akun itu dikaitkan dengan penggunaan lintas layanan. Gunakan `aws:SourceOrgID` untuk memungkinkan

sumber daya apa pun dari akun apa pun dalam suatu organisasi dikaitkan dengan penggunaan lintas layanan. Gunakan `aws:SourceOrgPaths` untuk mengaitkan sumber daya apa pun dari akun dalam AWS Organizations jalur dengan penggunaan lintas layanan. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat [Memahami jalur AWS Organizations entitas](#).

Cara paling efektif untuk melindungi dari masalah "confused deputy" adalah dengan menggunakan kunci konteks kondisi global `aws:SourceArn` dengan ARN lengkap sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci kondisi konteks global `aws:SourceArn` dengan karakter wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, `arn:aws:servicename:*:123456789012:*`.

Jika `aws:SourceArn` nilainya tidak berisi ID akun, seperti ARN bucket Amazon S3, Anda harus menggunakan keduanya `aws:SourceAccount` dan `aws:SourceArn` untuk membatasi izin.

Untuk melindungi dari masalah wakil yang membingungkan dalam skala besar, gunakan kunci konteks kondisi `aws:SourceOrgID` atau `aws:SourceOrgPaths` global dengan ID organisasi atau jalur organisasi sumber daya dalam kebijakan berbasis sumber daya Anda. Kebijakan yang menyertakan `aws:SourceOrgID` atau `aws:SourceOrgPaths` kunci akan secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual saat menambahkan, menghapus, atau memindahkan akun di organisasi Anda.

Kebijakan yang didokumentasikan untuk memberikan akses ke CloudWatch Log untuk menulis data ke Amazon Kinesis Data Streams dan Firehose [Langkah 1: Buat tujuan](#) di [Langkah 2: Buat tujuan](#) dan menunjukkan bagaimana Anda dapat menggunakan kunci konteks kondisi global untuk membantu `aws:SourceArn` mencegah masalah deputi yang membingungkan.

Pencegahan rekursi log

Ada risiko menyebabkan rekursi log tak terbatas dengan filter langganan yang dapat menyebabkan peningkatan besar dalam penagihan konsumsi di CloudWatch Log dan tujuan Anda, jika tidak dicegah. Hal ini dapat terjadi ketika filter langganan dikaitkan dengan grup log yang menerima peristiwa log sebagai hasil dari alur kerja pengiriman langganan Anda. Log yang tertelan ke dalam grup log akan dikirim ke tujuan, menyebabkan grup log menelan lebih banyak log yang kemudian akan diteruskan lagi ke tujuan, membuat loop rekursi.

Misalnya, pertimbangkan filter langganan dengan tujuan sebagai Firehose, yang mengirimkan peristiwa log ke Amazon S3. Selain itu, ada juga fungsi Lambda yang memproses peristiwa baru yang dikirim ke Amazon S3 dan menghasilkan beberapa log itu sendiri. Jika filter langganan

diterapkan ke grup log fungsi Lambda, maka peristiwa log yang dihasilkan oleh fungsi akan diteruskan ke Firehose dan Amazon S3 di tujuan, yang kemudian akan memanggil fungsi tersebut lagi, menyebabkan lebih banyak log diproduksi dan diteruskan ke Firehose dan Amazon S3, menyebabkan pemanggilan fungsi lainnya dan seterusnya. Ini akan terjadi dalam loop tak terbatas, yang mengarah ke peningkatan penagihan tak terduga pada konsumsi log, Firehose, dan Amazon S3.

Jika fungsi Lambda dilampirkan ke VPC dengan log aliran diaktifkan untuk Log, maka grup CloudWatch log VPC dapat menyebabkan rekursi log juga.

Kami menyarankan agar Anda tidak menerapkan filter langganan ke grup log yang merupakan bagian dari alur kerja pengiriman langganan Anda. Untuk filter langganan tingkat akun, gunakan `selectionCriteria` parameter di `PutAccountPolicy` API untuk mengecualikan grup log ini dari kebijakan.

Saat mengecualikan grup log, pertimbangkan AWS layanan berikut yang menghasilkan log dan mungkin menjadi bagian dari alur kerja pengiriman langganan Anda:

- Amazon EC2 dengan Fargate
- Lambda
- AWS Step Functions
- Log aliran VPC Amazon yang diaktifkan untuk Log CloudWatch

Note

Peristiwa log yang dihasilkan oleh grup log tujuan Lambda tidak akan diteruskan kembali ke fungsi Lambda untuk kebijakan filter langganan tingkat akun. Dalam hal ini, tidak termasuk grup log fungsi Lambda tujuan yang `selectionCriteria` digunakan tidak diperlukan untuk kebijakan berlangganan akun.

Filter sintaks pola untuk filter metrik, filter langganan, peristiwa log filter, dan Live Tail

Note

Untuk informasi tentang cara menanyakan grup log Anda dengan bahasa kueri Amazon CloudWatch Logs Insights, lihat [CloudWatch Sintaks kueri bahasa Wawasan Log](#).

Dengan CloudWatch Log, Anda dapat menggunakan [filter metrik](#) untuk mengubah data log menjadi metrik yang dapat ditindaklanjuti, [filter langganan](#) untuk merutekan peristiwa log ke AWS layanan lain, [memfilter peristiwa log](#) untuk mencari peristiwa log, dan [Live Tail](#) untuk secara interaktif melihat log Anda secara real-time saat tertelan.

Pola filter membentuk sintaks yang digunakan filter metrik, filter langganan, peristiwa log, dan Live Tail untuk mencocokkan istilah dalam peristiwa log. Istilah dapat berupa kata, frasa yang tepat, atau nilai numerik. Ekspresi reguler (regex) dapat digunakan untuk membuat pola filter mandiri, atau dapat digabungkan dengan JSON dan pola filter yang dibatasi ruang.

Buat pola filter dengan istilah yang ingin Anda cocokkan. Pola filter hanya mengembalikan peristiwa log yang berisi istilah yang Anda tentukan. Anda dapat menguji pola filter di CloudWatch konsol.

Topik

- [Sintaks ekspresi reguler \(regex\) yang didukung](#)
- [Menggunakan pola filter untuk mencocokkan istilah dengan ekspresi reguler \(regex\)](#)
- [Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log tidak terstruktur](#)
- [Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log JSON](#)
- [Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log yang dibatasi ruang](#)

Sintaks ekspresi reguler (regex) yang didukung

Sintaks regex yang didukung

Saat menggunakan regex untuk mencari dan memfilter data log, Anda harus mengelilingi ekspresi Anda dengan. %

Pola filter dengan regex hanya dapat mencakup yang berikut:

- Karakter alfanumerik — Karakter alfanumerik adalah karakter yang berupa huruf (dari A ke Z atau a hingga z) atau digit (dari 0 hingga 9).
- Karakter simbol yang didukung - Ini termasuk: `'_'`, `'#'`, `'='`, `'@'`, `'/'`, `'.'`, `'.'`, dan `'-'`. Misalnya, `%something!` akan ditolak karena `'!'` tidak didukung.
- Operator yang didukung - Ini termasuk: `^`, `$`, `?`, `[`, `]`, `{`, `}`, `|`, `\`, `*`, `+`, dan `.`.

)Operator (dan tidak didukung. Anda tidak dapat menggunakan tanda kurung untuk mendefinisikan subpola.

Karakter multi-byte tidak didukung.

Note

Kuota

Ada maksimal 5 pola filter yang berisi regex untuk setiap grup log saat membuat filter metrik atau filter langganan.

Ada batas 2 regex untuk setiap pola filter saat membuat pola filter terbatas atau JSON untuk filter metrik dan filter langganan atau saat memfilter peristiwa log atau Live Tail.

Penggunaan operator yang didukung

- `^`: Jangkar pertandingan ke awal string. Misalnya, `%^[hc]at%` cocok dengan “topi” dan “kucing”, tetapi hanya di awal tali.
- `$`: Jangkar korek api ke ujung string. Misalnya, `%[hc]at$%` cocok dengan “topi” dan “kucing”, tetapi hanya di ujung tali.
- `?`: Cocokkan nol atau satu kejadian dari suku sebelumnya. Misalnya, `%colou?r%` dapat mencocokkan “warna” dan “warna”.
- `[]`: Mendefinisikan kelas karakter. Cocokkan daftar karakter atau rentang karakter yang terkandung dalam tanda kurung. Misalnya, `%[abc]%` cocok dengan “a”, “b”, atau “c”; `%[a-z]%` cocok dengan huruf kecil dari “a” ke “z”; dan `%[abcx-z]%` cocok dengan “a”, “b”, “c”, “x”, “y”, atau “z”.
- `{m, n}`: Cocokkan istilah sebelumnya setidaknya m dan tidak lebih dari n kali. Misalnya, hanya `%a{3,5}%` cocok dengan “aaa”, “aaaa”, dan “aaaaa”.

Note

Entah `m` atau `n` dapat dihilangkan jika Anda memilih untuk tidak menentukan minimum atau maksimum.

- `|`: Boolean “Atau”, yang cocok dengan istilah di kedua sisi bilah vertikal. Contoh:
 - `%gra|ey%` dapat mencocokkan “abu-abu” atau “abu-abu”
 - `%^starting|^initializing|^shutting down%` dapat mencocokkan pertandingan “mulai...”, atau “menginisialisasi...”, atau “mematikan”, tetapi tidak akan cocok dengan “melewatkan inisialisasi...”
 - `%abcc|ab[^c]%` dapat mencocokkan “abcc...” dan “aba...” tetapi tidak akan cocok dengan “aac...”
- `\`: Karakter melarikan diri, yang memungkinkan Anda untuk menggunakan arti literal dari operator alih-alih makna khusus. Misalnya, `%\[.\\]%` cocok dengan karakter tunggal yang dikelilingi oleh “[” dan “]” karena tanda kurung diloloskan, seperti “[a]”, “[b]”, “[7]”, “[@]”, “[]”, dan “[]”.

Note

`%10\.10\.0\.1%` adalah cara yang benar untuk membuat regex agar sesuai dengan alamat IP 10.10.0.1.

- `*`: Cocokkan nol atau lebih contoh dari istilah sebelumnya. Misalnya, `%ab*c%` dapat mencocokkan “ac”, “abc”, dan “abbbc”; `%ab[0-9]*%` dapat mencocokkan “ab”, “ab0”, dan “ab129”.
- `+`: Cocokkan satu atau lebih contoh dari istilah sebelumnya. Misalnya, `%ab+c%` dapat mencocokkan “abc”, “abbc”, dan “abbbc”, tetapi tidak “ac”.
- `.`: Cocokkan karakter tunggal apa pun. Misalnya, `%.at%` mencocokkan tiga string karakter yang diakhiri dengan “at”, termasuk “hat”, “cat”, “bat”, “4at”, “#at” dan “at” (dimulai dengan spasi).

Note

Saat membuat regex agar sesuai dengan alamat IP, penting untuk melarikan diri dari operator. . Misalnya, `%10.10.0.1%` dapat mencocokkan “10010,051” yang mungkin bukan tujuan sebenarnya dari ekspresi tersebut.

- `\d,\D`: Cocokkan karakter digit/non-digit. Misalnya, `%\d%` setara dengan `%[0-9]%` dan `%\D%` setara dengan `%[^0-9]%`.

 Note

Operator huruf besar menunjukkan kebalikan dari rekan huruf kecil.

- `\s,\S`: Cocokkan karakter spasi/karakter non-spasi putih.

 Note

Operator huruf besar menunjukkan kebalikan dari rekan huruf kecil. Karakter spasi termasuk karakter tab (`\t`), spasi (), dan baris baru (`\n`).

- `\w,\W`: Cocokkan karakter alfanumerik/karakter non-alfanumerik. Misalnya, `%\w%` setara dengan `^[a-zA-Z_0-9]%` dan `%\W%` setara dengan `^[^a-zA-Z_0-9]%`.

 Note

Operator huruf besar menunjukkan kebalikan dari rekan huruf kecil.

- `\xhh`: Cocokkan pemetaan ASCII untuk karakter heksadesimal dua digit. `\x` adalah urutan escape yang menunjukkan bahwa karakter berikut mewakili nilai heksadesimal untuk ASCII. `hh` menentukan dua digit heksadesimal (0-9 dan A-F) yang menunjuk ke karakter dalam tabel ASCII.

 Note

Anda dapat menggunakan `\xhh` untuk mencocokkan karakter simbol yang tidak didukung oleh pola filter. Misalnya, `%\x3A%` pertandingan `:`; dan `%\x28%` pertandingan `(`.

Menggunakan pola filter untuk mencocokkan istilah dengan ekspresi reguler (regex)

Ketentuan kecocokan menggunakan regex

Anda dapat mencocokkan istilah dalam peristiwa log Anda menggunakan pola regex yang dikelilingi dengan `%` (tanda persentase sebelum dan sesudah pola regex). Cuplikan kode berikut

menunjukkan contoh pola filter yang mengembalikan semua peristiwa log yang terdiri dari kata kunci AUTHORIZED.

Untuk daftar ekspresi reguler yang didukung, lihat [Ekspresi reguler yang didukung](#).

```
%AUTHORIZED%
```

Pola filter ini mengembalikan pesan peristiwa log, seperti berikut ini:

- [ERROR 401] UNAUTHORIZED REQUEST
- [SUCCESS 200] AUTHORIZED REQUEST

Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log tidak terstruktur

Ketentuan kecocokan dalam peristiwa log tidak terstruktur

Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana Anda dapat menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log tidak terstruktur.

Note

Pola filter peka huruf besar/kecil. Lampirkan frasa dan istilah yang tepat yang menyertakan karakter non-alfanumerik dalam tanda kutip ganda ("").

Example: Match a single term

Cuplikan kode berikut menunjukkan contoh pola filter jangka tunggal yang mengembalikan semua peristiwa log di mana pesan berisi kata ERROR.

```
ERROR
```

Pola filter ini cocok dengan pesan peristiwa log, seperti berikut ini:

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST
- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

Example: Match multiple terms

Cuplikan kode berikut menunjukkan contoh pola filter multi-istilah yang mengembalikan semua peristiwa log di mana pesan berisi kata-kata ERROR dan ARGUMENTS.

```
ERROR ARGUMENTS
```

Filter mengembalikan pesan peristiwa log, seperti berikut ini:

- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

Pola filter ini tidak mengembalikan pesan peristiwa log berikut karena tidak berisi kedua istilah yang ditentukan dalam pola filter.

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST

Example: Match optional terms

Anda dapat menggunakan pencocokan pola untuk membuat pola filter yang menampilkan peristiwa log yang berisi istilah opsional. Tempatkan tanda tanya (“?”) sebelum persyaratan yang ingin Anda cocokkan. Cuplikan kode berikut menunjukkan contoh pola filter yang mengembalikan semua peristiwa log di mana pesan berisi kata ERROR atau kata ARGUMENTS.

```
?ERROR ?ARGUMENTS
```

Pola filter ini cocok dengan pesan peristiwa log, seperti berikut ini:

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST
- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

 Note

Anda tidak bisa menggabungkan tanda tanya (“?”) dengan pola filter lainnya, seperti menyertakan dan mengecualikan istilah. Jika Anda menggabungkan “?” dengan pola filter lainnya, semua istilah tanda tanya akan diabaikan.

Misalnya, pola filter berikut cocok dengan semua peristiwa yang mengandung kata REQUEST, tetapi tanda tanya (“?”) istilah filter diabaikan dan tidak berpengaruh.

```
?ERROR ?ARGUMENTS REQUEST
```

Log pertandingan acara

- [INFO] REQUEST FAILED
- [WARN] UNAUTHORIZED REQUEST
- [ERROR] 400 BAD REQUEST

Example: Match exact phrases

Cuplikan kode berikut menunjukkan contoh pola filter yang mengembalikan peristiwa log di mana pesan berisi frase yang tepat INTERNAL SERVER ERROR.

```
"INTERNAL SERVER ERROR"
```

Pola filter ini mengembalikan pesan peristiwa log berikut:

- [ERROR 500] INTERNAL SERVER ERROR

Example: Include and exclude terms

Anda dapat membuat pola filter yang menampilkan peristiwa log di mana pesan menyertakan beberapa istilah dan mengecualikan istilah lain. Tempatkan simbol minus ("-") sebelum istilah yang ingin Anda kecualikan. Cuplikan kode berikut menunjukkan contoh pola filter yang mengembalikan peristiwa log di mana pesan menyertakan istilah ERROR dan mengecualikan istilah ARGUMEN.

```
ERROR -ARGUMENTS
```

Pola filter ini mengembalikan pesan peristiwa log, seperti berikut ini:

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST

Pola filter ini tidak mengembalikan pesan peristiwa log berikut karena mengandung kata ARGUMENTS.

- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

Example: Match everything

Anda dapat mencocokkan semua yang ada di acara log Anda dengan tanda kutip ganda. Cuplikan kode berikut menunjukkan contoh pola filter yang mengembalikan semua peristiwa log.

```
" "
```

Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log JSON

Menulis pola filter untuk peristiwa log JSON

Berikut ini menjelaskan cara menulis sintaks untuk pola filter yang cocok dengan istilah JSON yang berisi string dan nilai numerik.

Writing filter patterns that match strings

Anda dapat membuat pola filter untuk mencocokkan string dalam peristiwa log JSON. Cuplikan kode berikut menunjukkan contoh sintaks untuk pola filter berbasis string.

```
{ PropertySelector EqualityOperator String }
```

Lampirkan pola filter dalam kurung kurawal ("{}"). Pola filter berbasis string harus berisi bagian-bagian berikut:

- **Pemilih properti**

Matikan pemilih properti dengan tanda dolar diikuti dengan titik ("\$."). Penyeleksi properti adalah string alfanumerik yang mendukung karakter tanda hubung ("-") dan garis bawah ("_"). String tidak mendukung notasi ilmiah. Penyeleksi properti menunjuk ke node nilai dalam peristiwa log JSON. Node nilai dapat berupa string atau angka. Tempatkan array setelah pemilih properti. Unsur-unsur dalam array mengikuti sistem penomoran berbasis nol, yang berarti bahwa elemen pertama dalam array adalah elemen 0, elemen kedua adalah elemen 1, dan seterusnya. Lampirkan elemen dalam tanda kurung ("[]"). Jika pemilih properti menunjuk ke array atau objek, pola filter tidak akan cocok dengan format log. Jika properti JSON berisi periode ("."), maka notasi braket dapat digunakan untuk memilih properti itu.



Note

Pemilih wildcard

Anda dapat menggunakan wildcard JSON untuk memilih elemen array atau bidang objek JSON apa pun.

Kuota

Anda hanya dapat menggunakan hingga satu pemilih wildcard di pemilih properti.

- Operator kesetaraan

Matikan operator kesetaraan dengan salah satu simbol berikut: sama ("=") atau tidak sama ("!="). Operator kesetaraan mengembalikan nilai Boolean (benar atau salah).

- Tali

Anda dapat melampirkan string dalam tanda kutip ganda (""). String yang berisi tipe selain karakter alfanumerik dan simbol garis bawah harus ditempatkan dalam tanda kutip ganda. Gunakan tanda bintang ("*") sebagai kartu liar untuk mencocokkan teks.

 Note

Anda dapat menggunakan ekspresi reguler bersyarat apa pun saat membuat pola filter untuk mencocokkan istilah dalam peristiwa log JSON. Untuk daftar ekspresi reguler yang didukung, lihat [Ekspresi reguler yang didukung](#).

Cuplikan kode berikut berisi contoh pola filter yang menunjukkan bagaimana Anda dapat memformat pola filter agar sesuai dengan istilah JSON dengan string.

```
{ $.eventType = "UpdateTrail" }
```

Writing filter patterns that match numeric values

Anda dapat membuat pola filter untuk mencocokkan nilai numerik dalam peristiwa log JSON. Cuplikan kode berikut menunjukkan contoh sintaks untuk pola filter yang cocok dengan nilai numerik.

```
{ PropertySelector NumericOperator Number }
```

Lampirkan pola filter dalam kurung kurawal ("{}"). Pola filter yang cocok dengan nilai numerik harus memiliki bagian berikut:

- **Pemilih properti**

Matikan pemilih properti dengan tanda dolar diikuti dengan titik (“\$.”). Penyeleksi properti adalah string alfanumerik yang mendukung karakter tanda hubung (“-”) dan garis bawah (“_”). String tidak mendukung notasi ilmiah. Penyeleksi properti menunjuk ke node nilai dalam peristiwa log JSON. Node nilai dapat berupa string atau angka. Tempatkan array setelah pemilih properti. Unsur-unsur dalam array mengikuti sistem penomoran berbasis nol, yang berarti bahwa elemen pertama dalam array adalah elemen 0, elemen kedua adalah elemen 1, dan seterusnya. Lampirkan elemen dalam tanda kurung (“[]”). Jika pemilih properti menunjuk ke array atau objek, pola filter tidak akan cocok dengan format log. Jika properti JSON berisi periode (“.”), maka notasi braket dapat digunakan untuk memilih properti itu.

**Note****Pemilih wildcard**

Anda dapat menggunakan wildcard JSON untuk memilih elemen array atau bidang objek JSON apa pun.

Kuota

Anda hanya dapat menggunakan hingga satu pemilih wildcard di pemilih properti.

- **Operator numerik**

Matikan operator numerik dengan salah satu simbol berikut: lebih besar dari (“>”), kurang dari (“<”), sama (“=”), tidak sama (“!=”), lebih besar dari atau sama dengan (“>=”), atau kurang dari atau sama dengan (“<=”).

- **Nomor**

Anda dapat menggunakan bilangan bulat yang berisi simbol plus (“+”) atau minus (“-”) dan mengikuti notasi ilmiah. Gunakan tanda bintang (“*”) sebagai kartu liar untuk mencocokkan angka.

Cuplikan kode berikut berisi contoh yang menunjukkan bagaimana Anda dapat memformat pola filter agar sesuai dengan istilah JSON dengan nilai numerik.

```
// Filter pattern with greater than symbol
{ $.bandwidth > 75 }
// Filter pattern with less than symbol
{ $.latency < 50 }
```

```
// Filter pattern with greater than or equal to symbol
{ $.refreshRate >= 60 }
// Filter pattern with less than or equal to symbol
{ $.responseTime <= 5 }
// Filter pattern with equal sign
{ $.errorCode = 400 }
// Filter pattern with not equal sign
{ $.errorCode != 500 }
// Filter pattern with scientific notation and plus symbol
{ $.number[0] = 1e+3 }
// Filter pattern with scientific notation and minus symbol
{ $.number[0] != 1e-3 }
```

Ketentuan kecocokan dalam peristiwa log JSON menggunakan ekspresi sederhana

Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana pola filter dapat mencocokkan istilah dalam peristiwa log JSON.

Note

Jika Anda menguji pola filter contoh dengan contoh peristiwa log JSON, Anda harus memasukkan log JSON contoh pada satu baris.

Peristiwa log JSON

```
{
  "eventType": "UpdateTrail",
  "sourceIPAddress": "111.111.111.111",
  "arrayKey": [
    "value",
    "another value"
  ],
  "objectList": [
    {
      "name": "a",
      "id": 1
    },
    {
      "name": "b",
      "id": 2
    }
  ]
}
```



```
    }  
  ],  
  "SomeObject": null,  
  "cluster.name": "c"  
}
```

Example: Filter pattern that matches string values

Pola filter ini cocok dengan string "UpdateTrail" di properti "eventType".

```
{ $.eventType = "UpdateTrail" }
```

Example: Filter pattern that matches string values (IP address)

Pola filter ini berisi kartu liar dan cocok dengan properti "sourceIPAddress" karena tidak mengandung angka dengan awalan "123.123.".

```
{ $.sourceIPAddress != 123.123.* }
```

Example: Filter pattern that matches a specific array element with a string value

Pola filter ini cocok dengan elemen "value" dalam array "arrayKey".

```
{ $.arrayKey[0] = "value" }
```

Example: Filter pattern that matches a string using regex

Pola filter ini cocok dengan string "Trail" di properti "eventType".

```
{ $.eventType = %Trail% }
```

Example: Filter pattern that uses a wildcard to match values of any element in the array using regex

Pola filter berisi regex yang cocok dengan elemen "value" dalam array. "arrayKey"

```
{ $.arrayKey[*] = %val.{2}% }
```

Example: Filter pattern that uses a wildcard to match values of any element with a specific prefix and subnet using regex (IP address)

Pola filter ini berisi regex yang cocok dengan elemen "111.111.111.111" dalam properti. "sourceIPAddress"

```
{ $.* = %111\.111\.111\.1[0-9]{1,2}% }
```

 Note

Kuota

Anda hanya dapat menggunakan hingga satu pemilih wildcard di pemilih properti.

Example: Filter pattern that matches a JSON property with a period (.) in the key

```
{ $.['cluster.name'] = "c" }
```

Example: Filter pattern that matches JSON logs using ADALAH

Anda dapat membuat pola filter yang cocok dengan bidang di log JSON dengan IS variabel. ISVariabel dapat mencocokkan bidang yang berisi nilaiNULL,TRUE, atauFALSE. Pola filter berikut mengembalikan log JSON di mana nilai SomeObject adalahNULL.

```
{ $.SomeObject IS NULL }
```

Example: Filter pattern that matches JSON logs using TIDAK ADA

Anda dapat membuat pola filter dengan NOT EXISTS variabel untuk mengembalikan log JSON yang tidak berisi bidang tertentu dalam data log. Pola filter berikut digunakan NOT EXISTS untuk mengembalikan log JSON yang tidak berisi bidangSomeOtherObject.

```
{ $.SomeOtherObject NOT EXISTS }
```

Note

Variabel IS NOT dan EXISTS saat ini tidak didukung.

Cocokkan istilah dalam objek JSON menggunakan ekspresi majemuk

Anda dapat menggunakan operator logika AND (“&&”) dan OR (“||”) dalam pola filter untuk membuat ekspresi gabungan yang cocok dengan peristiwa log di mana dua kondisi atau lebih benar. Ekspresi majemuk mendukung penggunaan tanda kurung (“()”) dan urutan operasi standar berikut: () > && > ||. Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana Anda dapat menggunakan pola filter dengan ekspresi majemuk untuk mencocokkan istilah dalam objek JSON.

Objek JSON

```
{
  "user": {
    "id": 1,
    "email": "John.Stiles@example.com"
  },
  "users": [
    {
      "id": 2,
      "email": "John.Doe@example.com"
    },
    {
      "id": 3,
      "email": "Jane.Doe@example.com"
    }
  ],
}
```

```
    "actions": [
      "GET",
      "PUT",
      "DELETE"
    ],
    "coordinates": [
      [0, 1, 2],
      [4, 5, 6],
      [7, 8, 9]
    ]
  }
}
```

Example: Expression that matches using AND (&&)

Pola filter ini berisi ekspresi majemuk yang cocok "id" "user" dengan nilai numerik 1 dan "email" dalam elemen pertama dari "users" array dengan string "John.Doe@example.com".

```
{ ($.user.id = 1) && ($.users[0].email = "John.Doe@example.com") }
```

Example: Expression that matches using OR (||)

Pola filter ini berisi ekspresi majemuk "email" yang cocok "user" dengan string "John.Stiles@example.com".

```
{ $.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch" &&
  $.actions[2] = "nonmatch" }
```

Example: Expression that doesn't match using AND (&&)

Pola filter ini berisi ekspresi majemuk yang tidak menemukan kecocokan karena ekspresi tidak cocok dengan tindakan ketiga di "actions".

```
{ ($.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch") &&
$.actions[2] = "nonmatch" }
```

Note

Kuota

Anda hanya dapat menggunakan hingga satu pemilih wildcard di pemilih properti, dan hingga tiga pemilih wildcard dalam pola filter dengan ekspresi majemuk.

Example: Expression that doesn't match using OR (||)

Pola filter ini berisi ekspresi majemuk yang tidak menemukan kecocokan karena ekspresi tidak cocok dengan properti pertama "users" atau tindakan ketiga di "actions".

```
{ ($.user.id = 2 && $.users[0].email = "nonmatch") || $.actions[2] = "GET" }
```

Menggunakan pola filter untuk mencocokkan istilah dalam peristiwa log yang dibatasi ruang

Menulis pola filter untuk peristiwa log yang dibatasi ruang

Anda dapat membuat pola filter agar sesuai dengan istilah dalam peristiwa log yang dibatasi ruang. Berikut ini memberikan contoh peristiwa log yang dibatasi ruang dan menjelaskan cara menulis sintaks untuk pola filter yang cocok dengan istilah dalam peristiwa log yang dibatasi ruang.

Note

Anda dapat menggunakan ekspresi reguler bersyarat apa pun saat membuat pola filter untuk mencocokkan istilah dalam peristiwa log yang dibatasi spasi. Untuk daftar ekspresi reguler yang didukung, lihat [Ekspresi reguler yang didukung](#).

Example: Space-delimited log event

Cuplikan kode berikut menunjukkan peristiwa log yang dibatasi spasi yang berisi tujuh bidang: `ip`, `user`, `username` dan `timestamp` `request` `status_code` `bytes`

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404  
1534
```

Note

Karakter antara tanda kurung ("`[]`") dan tanda kutip ganda ("`\"`") dianggap bidang tunggal.

Writing filter patterns that match terms in a space-delimited log event

Untuk membuat pola filter yang cocok dengan istilah dalam peristiwa log yang dibatasi spasi, lampirkan pola filter dalam tanda kurung ("`[]`"), dan tentukan bidang dengan nama yang dipisahkan dengan koma ("`,`"). Pola filter berikut mem-parsing tujuh bidang.

```
[ip=%127\.\0\.\0\[1-9]%, user, username, timestamp, request =*.html*, status_code =  
4*, bytes]
```

Anda dapat menggunakan operator numerik (`>`, `<`, `=`, `!=`, `>=`, atau `<=`) dan tanda bintang (`*`) sebagai wild card atau regex untuk memberikan kondisi pola filter Anda. Dalam contoh pola filter, `ip` menggunakan regex yang cocok dengan rentang alamat IP 127.0.0.1 - 127.0.0.9, `request` berisi wildcard yang menyatakan harus mengekstrak nilai dengan `.html`, dan `status_code` berisi wildcard yang menyatakan harus mengekstrak nilai yang dimulai dengan `4`

Jika Anda tidak mengetahui jumlah bidang yang Anda parsing dalam peristiwa log yang dibatasi spasi, Anda dapat menggunakan elipsis (`...`) untuk mereferensikan bidang yang tidak disebutkan namanya. Elipsis dapat mereferensikan bidang sebanyak yang diperlukan. Contoh

berikut menunjukkan pola filter dengan elipsis yang mewakili empat bidang pertama yang tidak disebutkan namanya yang ditunjukkan pada pola filter contoh sebelumnya.

```
[..., request =*.html*, status_code = 4*, bytes]
```

Anda juga dapat menggunakan operator logika AND (&&) dan OR (||) untuk membuat ekspresi majemuk. Pola filter berikut berisi ekspresi majemuk yang menyatakan nilai `status_code` must be 404 atau 410.

```
[ip, user, username, timestamp, request =*.html*, status_code = 404 || status_code = 410, bytes]
```

Ketentuan kecocokan dalam peristiwa log yang dibatasi ruang menggunakan pencocokan pola

Anda dapat menggunakan pencocokan pola untuk membuat pola filter yang dibatasi ruang yang cocok dengan istilah dalam urutan tertentu. Tentukan urutan persyaratan Anda dengan indikator. Gunakan `w1` untuk mewakili istilah pertama Anda dan `w2` dan seterusnya untuk mewakili urutan persyaratan Anda berikutnya. Tempatkan koma (",") di antara istilah Anda. Contoh berikut berisi cuplikan kode yang menunjukkan bagaimana Anda dapat menggunakan pencocokan pola dengan pola filter yang dibatasi spasi.

Note

Anda dapat menggunakan ekspresi reguler bersyarat apa pun saat membuat pola filter untuk mencocokkan istilah dalam peristiwa log yang dibatasi spasi. Untuk daftar ekspresi reguler yang didukung, lihat [Ekspresi reguler yang didukung](#).

Peristiwa log yang dibatasi ruang

```
INFO 09/25/2014 12:00:00 GET /service/resource/67 1200
INFO 09/25/2014 12:00:01 POST /service/resource/67/part/111 1310
WARNING 09/25/2014 12:00:02 Invalid user request
```

```
ERROR 09/25/2014 12:00:02 Failed to process request
```

Example: Match terms in order

Pola filter yang dibatasi spasi berikut mengembalikan peristiwa log di mana kata pertama dalam peristiwa log adalah ERROR.

```
[w1=ERROR, w2]
```

Note

Saat Anda membuat pola filter yang dibatasi spasi yang menggunakan pencocokan pola, Anda harus menyertakan indikator kosong setelah Anda menentukan urutan istilah Anda. Misalnya, jika Anda membuat pola filter yang mengembalikan peristiwa log di mana kata pertama adalah ERROR, sertakan indikator w2 kosong setelah istilah w1.

Example: Match terms with AND (&&) and OR (||)

Anda dapat menggunakan operator logis AND (“&&”) dan OR (“||”) untuk membuat pola filter yang dibatasi spasi yang berisi kondisi. Pola filter berikut mengembalikan peristiwa log di mana kata pertama dalam peristiwa adalah ERROR atau PERINGATAN.

```
[w1=ERROR || w1=WARNING, w2]
```

Example: Exclude terms from matches

Anda dapat membuat pola filter yang dibatasi spasi yang menampilkan peristiwa log tidak termasuk satu atau beberapa istilah. Tempatkan simbol yang tidak sama (“!=”) sebelum istilah atau istilah yang ingin Anda kecualikan. Cuplikan kode berikut menunjukkan contoh pola filter yang mengembalikan peristiwa log di mana kata-kata pertama tidak ERROR dan PERINGATAN.

```
[w1!=ERROR && w1!=WARNING, w2]
```


Example: Match the top level item in a resource URI

Cuplikan kode berikut menunjukkan contoh pola filter yang cocok dengan item tingkat atas dalam URI sumber daya menggunakan regex.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+$, response_time]
```

Example: Match the child level item in a resource URI

Cuplikan kode berikut menunjukkan contoh pola filter yang cocok dengan item tingkat anak dalam URI sumber daya menggunakan regex.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+/part/[0-9]+$,  
response_time]
```

Aktifkan logging dari AWS layanan

Meskipun banyak layanan mempublikasikan log hanya ke CloudWatch Log, beberapa AWS layanan dapat mempublikasikan log langsung ke Amazon Simple Storage Service atau Amazon Data Firehose. Jika persyaratan utama Anda untuk log adalah penyimpanan atau pemrosesan di salah satu layanan ini, Anda dapat dengan mudah memiliki layanan yang menghasilkan log mengirimkannya langsung ke Amazon S3 atau Firehose tanpa pengaturan tambahan.

Bahkan ketika Anda mempublikasikan log langsung ke Amazon S3 atau Firehose, CloudWatch biaya pengiriman berlaku. Jika Anda mengirim log ke Amazon S3, **AWS_REGION**-S3-Egress-Bytes tagihan akan muncul di Cost Explorer atau di tagihan Anda. Jika Anda mengirim log ke Firehose, maka **AWS_REGION**-FH-Egress-Bytes tagihan akan muncul. Untuk informasi selengkapnya tentang harga log penjual otomatis, lihat tab Log di [CloudWatch Harga Amazon](#).

Beberapa AWS layanan menggunakan infrastruktur umum untuk mengirim log mereka. Untuk mengaktifkan logging dari layanan ini, Anda harus masuk sebagai pengguna yang memiliki izin tertentu. Selain itu, Anda harus memberikan izin AWS untuk mengaktifkan log yang akan dikirim.

Untuk layanan yang memerlukan izin ini, ada dua versi izin yang diperlukan. Layanan yang memerlukan izin tambahan ini dicatat sebagai [Izin V1] yang Didukung dan [Izin V2] yang Didukung dalam tabel. Untuk informasi tentang izin yang diperlukan ini, lihat bagian setelah tabel.

<integration-catalog>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

<integration></integration>

[illegible]

[illegible]

Logging yang membutuhkan izin tambahan [V1]

Beberapa AWS layanan menggunakan infrastruktur umum untuk mengirim log mereka ke CloudWatch Log, Amazon S3, atau Firehose. Untuk mengaktifkan AWS layanan yang tercantum dalam tabel sebelumnya untuk mengirim log mereka ke tujuan ini, Anda harus masuk sebagai pengguna yang memiliki izin tertentu.

Selain itu, izin harus diberikan AWS untuk mengaktifkan log yang akan dikirim. AWS dapat secara otomatis membuat izin tersebut ketika log disiapkan, atau Anda dapat membuatnya sendiri terlebih dahulu sebelum Anda mengatur logging. Untuk pengiriman lintas akun, Anda harus membuat sendiri kebijakan izin secara manual.

Jika Anda memilih untuk AWS secara otomatis mengatur izin dan kebijakan sumber daya yang diperlukan saat Anda atau seseorang di organisasi Anda pertama kali mengatur pengiriman log, maka pengguna yang menyiapkan pengiriman log harus memiliki izin tertentu, seperti yang dijelaskan nanti di bagian ini. Selain itu, Anda dapat membuat kebijakan sumber daya sendiri, dan kemudian pengguna yang mengatur pengiriman log tidak memerlukan banyak izin.

Topik berikut memberikan rincian lebih lanjut untuk masing-masing tujuan ini.

Topik

- [Log dikirim ke CloudWatch Log](#)
- [Log yang dikirim ke Amazon S3](#)
- [Log dikirim ke Firehose](#)

Log dikirim ke CloudWatch Log

Important

Ketika Anda mengatur jenis log dalam daftar berikut untuk dikirim ke CloudWatch Log, AWS membuat atau mengubah kebijakan sumber daya yang terkait dengan grup log yang menerima log, jika diperlukan. Lanjutkan membaca bagian ini untuk melihat detailnya.

Bagian ini berlaku ketika jenis log yang tercantum dalam tabel di bagian sebelumnya dikirim ke CloudWatch Log:

Izin pengguna

Untuk dapat mengatur pengiriman salah satu jenis log ini ke CloudWatch Log untuk pertama kalinya, Anda harus masuk ke akun dengan izin berikut.

- `logs:CreateLogDelivery`
- `logs:PutResourcePolicy`
- `logs:DescribeResourcePolicies`

- `logs:DescribeLogGroups`

Note

Saat Anda menentukan `logs:DescribeLogGroups`, `logs:DescribeResourcePolicies`, atau `logs:PutResourcePolicy` izin, pastikan untuk mengatur ARN Resource barisnya untuk menggunakan * wildcard, alih-alih hanya menentukan satu nama grup log. Sebagai contoh, `"Resource": "arn:aws:logs:us-east-1:111122223333:log-group:*"`.

Jika salah satu jenis log ini sudah dikirim ke grup CloudWatch log di Log, maka untuk mengatur pengiriman salah satu jenis log ini ke grup log yang sama, Anda hanya perlu `logs:CreateLogDelivery` izin.

Kebijakan sumber daya grup log

Grup log tempat log dikirim harus memiliki kebijakan sumber daya yang mencakup izin tertentu. Jika grup log saat ini tidak memiliki kebijakan sumber daya, dan pengguna yang mengatur logging memiliki `logs:PutResourcePolicy`, `logs:DescribeResourcePolicies`, dan `logs:DescribeLogGroups` izin untuk grup log, maka AWS secara otomatis membuat kebijakan berikut untuk itu ketika Anda mulai mengirim CloudWatch log ke Log. Untuk langganan yang baru dibuat, kebijakan sumber daya dikonfigurasi pada tingkat grup log dan memiliki ukuran maksimum 51.200 byte. Jika kebijakan sumber daya tingkat akun yang ada sudah memberikan izin melalui wildcard, kebijakan tingkat grup log terpisah tidak akan dibuat. Untuk memeriksa kebijakan sumber daya tingkat Loggroup untuk grup log tertentu, gunakan `describe-resource-policies` perintah dengan parameter yang disetel ke grup log ARN dan `--resource-arn` parameter yang disetel ke `--policy-scope RESOURCE`

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": [
```

```

        "delivery.logs.amazonaws.com"
    ],
    },
    "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
    ],
    "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group:log-
stream:*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": [
                "0123456789"
            ]
        },
        "ArnLike": {
            "aws:SourceArn": [
                "arn:aws:logs:us-east-1:111122223333:*"
            ]
        }
    }
}

```

Batas kebijakan sumber daya grup log adalah 51.200 byte. Setelah batas ini tercapai, AWS tidak dapat menambahkan izin baru. Ini mengharuskan pelanggan untuk memodifikasi kebijakan secara manual untuk memberikan izin utama `delivery.logs.amazonaws.com` layanan pada `logs:PutLogEvents` tindakan `logs:CreateLogStream` dan tindakan. Pelanggan harus menggunakan awalan nama grup log dengan wildcard seperti `/aws/vendedlogs/*` dan menggunakan nama grup log ini untuk pembuatan Pengiriman di masa mendatang.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite20150319",

```

```

    "Effect": "Allow",
    "Principal": {
      "Service": [
        "delivery.logs.amazonaws.com"
      ]
    },
    "Action": [
      "logs:CreateLogStream",
      "logs:PutLogEvents"
    ],
    "Resource": [
      "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group/aws/
vendedlogs/*"
    ],
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": [
          "0123456789"
        ]
      },
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:logs:us-east-1:111122223333:*"
        ]
      }
    }
  }
}

```

Log yang dikirim ke Amazon S3

Saat Anda menyetel log untuk dikirim ke Amazon S3, AWS buat atau ubah kebijakan sumber daya yang terkait dengan bucket S3 yang menerima log, jika diperlukan.

Log yang diterbitkan langsung ke Amazon S3 diterbitkan ke bucket lama yang Anda tentukan. Satu atau lebih berkas log dibuat setiap lima menit dalam bucket yang ditetapkan.

Ketika Anda mengirimkan log untuk pertama kalinya ke bucket Amazon S3, layanan yang mengirimkan log mencatat pemilik bucket untuk memastikan bahwa log dikirim hanya untuk bucket milik akun ini. Oleh karenanya, untuk mengubah pemilik bucket Amazon S3, Anda harus membuat ulang atau memperbarui langganan log di layanan asal.

 Note

CloudFront menggunakan model izin yang berbeda dari layanan lain yang mengirim log vendek ke S3. Untuk informasi selengkapnya, lihat [Izin yang diperlukan untuk mengonfigurasi pencatatan log standar dan untuk mengakses berkas log Anda](#).

Selain itu, jika Anda menggunakan bucket S3 yang sama untuk log CloudFront akses dan sumber log lainnya, mengaktifkan ACL di bucket CloudFront juga memberikan izin ke semua sumber log lain yang menggunakan bucket ini.

 Important

Jika Anda mengirim log ke bucket Amazon S3 dan kebijakan bucket berisi `NotPrincipal` elemen `NotAction` atau, menambahkan izin pengiriman log ke bucket secara otomatis dan membuat langganan log akan gagal. Agar berhasil membuat langganan log, Anda perlu menambahkan izin pengiriman log secara manual ke kebijakan bucket, lalu membuat langganan log. Untuk informasi lebih lanjut, lihat petunjuk di bagian ini.

Jika bucket memiliki enkripsi sisi server menggunakan AWS KMS kunci yang dikelola pelanggan, Anda juga harus menambahkan kebijakan kunci untuk kunci terkelola pelanggan Anda. Untuk informasi selengkapnya, lihat [Amazon S3](#).

Jika bucket tujuan mengaktifkan SSE-KMS dan Kunci Bucket, kebijakan kunci KMS terkelola pelanggan terlampir tidak lagi berfungsi seperti yang diharapkan untuk semua permintaan. Untuk informasi selengkapnya, lihat [Mengurangi biaya SSE-KMS dengan Amazon S3](#) Bucket Keys.

Jika Anda menggunakan log penjual otomatis dan enkripsi S3 dengan AWS KMS kunci yang dikelola pelanggan, Anda harus menggunakan AWS KMS ARN kunci yang sepenuhnya memenuhi syarat alih-alih ID kunci saat mengonfigurasi bucket. Untuk informasi selengkapnya, lihat [put-bucket-encryption](#).

Izin pengguna

Untuk dapat mengatur pengiriman salah satu jenis log ini ke Amazon S3 untuk pertama kalinya, Anda harus masuk ke akun dengan izin berikut.

- `logs:CreateLogDelivery`
- `S3:GetBucketPolicy`

- S3:PutBucketPolicy

Jika salah satu jenis log ini sudah dikirim ke bucket Amazon S3, untuk mengatur pengiriman dari salah satu jenis log ini ke bucket yang sama Anda hanya perlu memiliki izin `logs:CreateLogDelivery`.

Kebijakan sumber daya bucket S3

Bucket S3 tempat log dikirim harus memiliki kebijakan sumber daya yang mencakup izin tertentu. Jika saat ini bucket tidak memiliki kebijakan sumber daya, dan pengguna yang mengatur pencatatan memiliki izin `S3:GetBucketPolicy` dan `S3:PutBucketPolicy` untuk bucket, maka AWS secara otomatis membuat kebijakan berikut ini saat Anda mulai mengirim log ke Amazon S3.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "AWSLogDeliveryWrite20150319",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryAclCheck",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "0123456789"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:111122223333:*"
          ]
        }
      }
    }
  ],
  {
```

```

    "Sid": "AWSLogDeliveryWrite",
    "Effect": "Allow",
    "Principal": {
        "Service": "delivery.logs.amazonaws.com"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/account-ID/*",
    "Condition": {
        "StringEquals": {
            "s3:x-amz-acl": "bucket-owner-full-control",
            "aws:SourceAccount": [
                "0123456789"
            ]
        },
        "ArnLike": {
            "aws:SourceArn": [
                "arn:aws:logs:us-east-1:111122223333:*"
            ]
        }
    }
}

```

Dalam kebijakan sebelumnya, untuk `aws:SourceAccount`, tentukan daftar ID akun tempat log dikirimkan ke bucket ini. Untuk `aws:SourceArn`, tentukan daftar ARNs sumber daya yang menghasilkan log, dalam formulir `arn:aws:logs:source-region:source-account-id:*`.

Jika bucket memiliki kebijakan sumber daya tetapi kebijakan tersebut tidak berisi pernyataan yang ditampilkan di kebijakan sebelumnya, dan pengguna yang menyiapkan logging memiliki `S3:PutBucketPolicy` izin `S3:GetBucketPolicy` dan untuk bucket, pernyataan tersebut akan ditambahkan ke kebijakan sumber daya bucket.

Note

Dalam beberapa kasus, Anda mungkin melihat `AccessDenied` kesalahan AWS CloudTrail jika `s3:ListBucket` izin belum diberikan ke `delivery.logs.amazonaws.com`. Untuk menghindari kesalahan ini di CloudTrail log Anda, Anda harus memberikan `s3:ListBucket` izin ke `delivery.logs.amazonaws.com` dan Anda harus menyertakan `Condition` parameter yang ditampilkan dengan `s3:GetBucketAcl` izin yang ditetapkan dalam

kebijakan bucket sebelumnya. Untuk membuatnya lebih sederhana, alih-alih membuat yang baruStatement, Anda dapat langsung memperbarui `AWSLogDeliveryAclCheck` to be `"Action": ["s3:GetBucketAcl", "s3:ListBucket"]`

Enkripsi sisi server bucket Amazon S3

Anda dapat melindungi data di bucket Amazon S3 dengan mengaktifkan Enkripsi sisi server dengan kunci yang dikelola Amazon S3 (SSE-S3) atau enkripsi sisi server dengan kunci yang disimpan di (SSE-KMS). AWS KMS AWS Key Management Service Untuk informasi selengkapnya, silakan lihat [Melindungi data menggunakan enkripsi sisi server](#).

Jika Anda memilih SSE-S3, tidak diperlukan konfigurasi tambahan. Amazon S3 menangani kunci enkripsi.

Warning

Jika Anda memilih SSE-KMS, Anda harus menggunakan kunci yang dikelola pelanggan, karena menggunakan kunci AWS terkelola tidak didukung untuk skenario ini. Jika Anda mengatur enkripsi menggunakan kunci AWS terkelola, log akan dikirimkan dalam format yang tidak dapat dibaca.

Saat menggunakan AWS KMS kunci terkelola pelanggan, Anda dapat menentukan Nama Sumber Daya Amazon (ARN) kunci terkelola pelanggan saat mengaktifkan enkripsi bucket. Anda harus menambahkan hal berikut ke kebijakan kunci untuk kunci terkelola pelanggan Anda (bukan ke kebijakan bucket untuk bucket S3 Anda), sehingga akun pengiriman log dapat menulis ke bucket S3 Anda.

Jika Anda memilih SSE-KMS, Anda harus menggunakan kunci yang dikelola pelanggan, karena menggunakan kunci AWS terkelola tidak didukung untuk skenario ini. Saat menggunakan AWS KMS kunci terkelola pelanggan, Anda dapat menentukan Nama Sumber Daya Amazon (ARN) kunci terkelola pelanggan saat mengaktifkan enkripsi bucket. Anda harus menambahkan hal berikut ke kebijakan kunci untuk kunci terkelola pelanggan Anda (bukan ke kebijakan bucket untuk bucket S3 Anda), sehingga akun pengiriman log dapat menulis ke bucket S3 Anda.

```
{  
  "Sid": "Allow Logs Delivery to use the key",
```

```

    "Effect": "Allow",
    "Principal": {
      "Service": [ "delivery.logs.amazonaws.com" ]
    },
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncrypt*",
      "kms:GenerateDataKey*",
      "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": ["0123456789"]
      },
      "ArnLike": {
        "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:*"]
      }
    }
  }
}

```

Untuk `aws:SourceAccount`, tentukan daftar ID akun tempat log dikirimkan ke bucket ini. Untuk `aws:SourceArn`, tentukan daftar ARNs sumber daya yang menghasilkan log, dalam formulir `arn:aws:logs:source-region:source-account-id:*`.

Log dikirim ke Firehose

Bagian ini berlaku ketika jenis log yang tercantum dalam tabel di bagian sebelumnya dikirim ke Firehose:

Izin pengguna

Untuk dapat mengatur pengiriman salah satu jenis log ini ke Firehose untuk pertama kalinya, Anda harus masuk ke akun dengan izin berikut.

- `logs:CreateLogDelivery`
- `firehose:TagDeliveryStream`
- `iam:CreateServiceLinkedRole`

Jika salah satu dari jenis log ini sudah dikirim ke Firehose, maka untuk mengatur pengiriman salah satu dari jenis log ini ke Firehose, Anda hanya perlu memiliki izin `logs:CreateLogDelivery` `firehose:TagDeliveryStream`

Peran IAM yang digunakan untuk izin

Karena Firehose tidak menggunakan kebijakan sumber daya, AWS menggunakan peran IAM saat menyiapkan log ini untuk dikirim ke Firehose. AWS membuat peran terkait layanan bernama `AWSServiceRoleForLogDelivery`. Peran terkait layanan ini mencakup izin berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "firehose:PutRecord",
        "firehose:PutRecordBatch",
        "firehose:ListTagsForDeliveryStream"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/LogDeliveryEnabled": "true"
        }
      },
      "Effect": "Allow"
    }
  ]
}
```

Peran terkait layanan ini memberikan izin untuk semua aliran pengiriman Firehose yang memiliki tag yang disetel ke `LogDeliveryEnabled true`. AWS memberikan tag ini ke aliran pengiriman tujuan saat Anda mengatur logging.

Peran terkait layanan ini juga memiliki kebijakan kepercayaan yang memungkinkan layanan `delivery.logs.amazonaws.com` utama untuk mengasumsikan peran yang terhubung dengan layanan yang diperlukan. Kebijakan kepercayaan tersebut adalah sebagai berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Logging yang membutuhkan izin tambahan [V2]

Beberapa AWS layanan menggunakan metode baru untuk mengirim log mereka. Ini adalah metode fleksibel yang memungkinkan Anda mengatur pengiriman log dari layanan ini ke satu atau beberapa tujuan berikut: CloudWatch Log, Amazon S3, atau Firehose dan X-Ray untuk pengiriman jejak.

Pengiriman log kerja terdiri dari tiga elemen:

- `ADeliverySource`, yang merupakan objek logis yang mewakili sumber daya yang benar-benar mengirim log.
- `ADeliveryDestination`, yang merupakan objek logis yang mewakili tujuan pengiriman yang sebenarnya.
- `ADelivery`, yang menghubungkan sumber pengiriman ke tujuan pengiriman

Untuk mengonfigurasi pengiriman log antara AWS layanan yang didukung dan tujuan, Anda harus melakukan hal berikut:

- Buat sumber pengiriman dengan [PutDeliverySource](#).
- Buat tujuan pengiriman dengan [PutDeliveryDestination](#).
- Jika Anda mengirimkan log lintas akun, Anda harus menggunakan [PutDeliveryDestinationPolicy](#) di akun tujuan untuk menetapkan IAM kebijakan ke tujuan. Kebijakan ini mengotorisasi pembuatan

pengiriman dari sumber pengiriman di akun A ke tujuan pengiriman di akun B. Untuk pengiriman lintas akun, Anda harus membuat sendiri kebijakan izin secara manual.

- Buat pengiriman dengan memasangkan tepat satu sumber pengiriman dan satu tujuan pengiriman, dengan menggunakan [CreateDelivery](#).

Bagian berikut memberikan rincian izin yang perlu Anda miliki saat Anda masuk untuk mengatur pengiriman log ke setiap jenis tujuan, menggunakan proses V2. Izin ini dapat diberikan ke peran IAM yang Anda masuki.

Important

Anda bertanggung jawab untuk menghapus sumber daya pengiriman log setelah menghapus sumber daya penghasil log. Untuk melakukannya, ikuti langkah-langkah ini.

1. Hapus Delivery dengan menggunakan [DeleteDelivery](#) operasi.
2. Hapus DeliverySource dengan menggunakan [DeleteDeliverySource](#) operasi.
3. Jika yang DeliveryDestination terkait dengan DeliverySource yang baru saja Anda hapus hanya digunakan untuk spesifik ini DeliverySource, maka Anda dapat menghapusnya dengan menggunakan [DeleteDeliveryDestinations](#) operasi.

Daftar Isi

- [Log dikirim ke CloudWatch Log](#)
- [Log yang dikirim ke Amazon S3](#)
 - [Enkripsi sisi server bucket Amazon S3](#)
- [Log dikirim ke Firehose](#)
- [Jejak dikirim ke X-Ray](#)

Log dikirim ke CloudWatch Log

Izin pengguna

Untuk mengaktifkan pengiriman CloudWatch log ke Log, Anda harus masuk dengan izin berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadWriteAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:GetDelivery",
        "logs:GetDeliverySource",
        "logs:PutDeliveryDestination",
        "logs:GetDeliveryDestinationPolicy",
        "logs>DeleteDeliverySource",
        "logs:PutDeliveryDestinationPolicy",
        "logs>CreateDelivery",
        "logs:GetDeliveryDestination",
        "logs:PutDeliverySource",
        "logs>DeleteDeliveryDestination",
        "logs>DeleteDeliveryDestinationPolicy",
        "logs>DeleteDelivery",
        "logs:UpdateDeliveryConfiguration"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:delivery:*",
        "arn:aws:logs:us-east-1:444455556666:delivery-source:*",
        "arn:aws:logs:us-east-1:777788889999:delivery-destination:*"
      ]
    },
    {
      "Sid": "ListAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeDeliveryDestinations",
        "logs:DescribeDeliverySources",
        "logs:DescribeDeliveries",
        "logs:DescribeConfigurationTemplates"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowUpdatesToResourcePolicyCWL",
      "Effect": "Allow",
```

```

        "Action": [
            "logs:PutResourcePolicy",
            "logs:DescribeResourcePolicies",
            "logs:DescribeLogGroups"
        ],
        "Resource": [
            "arn:aws:logs:us-east-1:123456789012:*"
        ]
    }
]
}

```

Kebijakan sumber daya grup log

Grup log tempat log dikirim harus memiliki kebijakan sumber daya yang mencakup izin tertentu. Jika grup log saat ini tidak memiliki kebijakan sumber daya, dan pengguna yang mengatur logging memiliki `logs:PutResourcePolicy`, `logs:DescribeResourcePolicies`, dan `logs:DescribeLogGroups` izin untuk grup log, maka AWS secara otomatis membuat kebijakan berikut untuk itu ketika Anda mulai mengirim CloudWatch log ke Log. Untuk langganan yang baru dibuat, kebijakan sumber daya dikonfigurasi pada tingkat grup log dan memiliki ukuran maksimum 51.200 byte. Jika kebijakan sumber daya tingkat akun yang ada sudah memberikan izin melalui wildcard, kebijakan tingkat grup log terpisah tidak akan dibuat. Untuk memeriksa kebijakan sumber daya tingkat Loggroup untuk grup log tertentu, gunakan `describe-resource-policies` perintah dengan parameter yang disetel ke grup log ARN dan `--resource-arn` parameter yang disetel ke `--policy-scope RESOURCE`

JSON

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "delivery.logs.amazonaws.com"
                ]
            },
            "Action": [

```

```

        "logs:CreateLogStream",
        "logs:PutLogEvents"
    ],
    "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group:log-
stream:*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": [
                "0123456789"
            ]
        },
        "ArnLike": {
            "aws:SourceArn": [
                "arn:aws:logs:us-east-1:111122223333:*"
            ]
        }
    }
}

```

Batas kebijakan sumber daya grup log adalah 51.200 byte. Setelah batas ini tercapai, AWS tidak dapat menambahkan izin baru. Ini mengharuskan pelanggan untuk memodifikasi kebijakan secara manual untuk memberikan izin utama `delivery.logs.amazonaws.com` layanan pada `logs:PutLogEvents` tindakan `logs:CreateLogStream` dan tindakan. Pelanggan harus menggunakan awalan nama grup log dengan wildcard seperti `/aws/vendedlogs/*` dan menggunakan nama grup log ini untuk pembuatan Pengiriman di masa mendatang.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "delivery.logs.amazonaws.com"
        ]
      }
    }
  ]
}

```

```

    ],
    "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
    ],
    "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:my-log-group/aws/
vendedlogs/*"
    ],
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": [
                "0123456789"
            ]
        },
        "ArnLike": {
            "aws:SourceArn": [
                "arn:aws:logs:us-east-1:111122223333:*"
            ]
        }
    }
}
]
}

```

Log yang dikirim ke Amazon S3

Izin pengguna

Untuk mengaktifkan pengiriman log ke Amazon S3, Anda harus masuk dengan izin berikut.

JSON

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:GetDelivery",
            ]
        }
    ]
}

```

```

        "logs:GetDeliverySource",
        "logs:PutDeliveryDestination",
        "logs:GetDeliveryDestinationPolicy",
        "logs>DeleteDeliverySource",
        "logs:PutDeliveryDestinationPolicy",
        "logs:CreateDelivery",
        "logs:GetDeliveryDestination",
        "logs:PutDeliverySource",
        "logs>DeleteDeliveryDestination",
        "logs>DeleteDeliveryDestinationPolicy",
        "logs>DeleteDelivery",
        "logs:UpdateDeliveryConfiguration"
    ],
    "Resource": [
        "arn:aws:logs:us-east-1:111122223333:delivery:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
    ]
},
{
    "Sid": "ListAccessForLogDeliveryActions",
    "Effect": "Allow",
    "Action": [
        "logs:DescribeDeliveryDestinations",
        "logs:DescribeDeliverySources",
        "logs:DescribeDeliveries",
        "logs:DescribeConfigurationTemplates"
    ],
    "Resource": "*"
},
{
    "Sid": "AllowUpdatesToResourcePolicyS3",
    "Effect": "Allow",
    "Action": [
        "s3:PutBucketPolicy",
        "s3:GetBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::bucket-name"
}
]
}

```

Bucket S3 tempat log dikirim harus memiliki kebijakan sumber daya yang mencakup izin tertentu. Jika saat ini bucket tidak memiliki kebijakan sumber daya, dan pengguna yang mengatur pencatatan memiliki izin `S3:GetBucketPolicy` dan `S3:PutBucketPolicy` untuk bucket, maka AWS secara otomatis membuat kebijakan berikut ini saat Anda mulai mengirim log ke Amazon S3.

JSON

```
{
  "Version": "2012-10-17",
  "Id": "AWSLogDeliveryWrite20150319",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/account-ID/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control",
          "aws:SourceAccount": [
            "0123456789"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:111122223333:delivery-source:*"
          ]
        }
      }
    }
  ]
}
```

Dalam kebijakan sebelumnya, untuk `aws:SourceAccount`, tentukan daftar ID akun tempat log dikirimkan ke bucket ini. Untuk `aws:SourceArn`, tentukan daftar ARNs sumber daya yang menghasilkan log, dalam formulir `arn:aws:logs:source-region:source-account-id:*`.

Jika bucket memiliki kebijakan sumber daya tetapi kebijakan tersebut tidak berisi pernyataan yang ditampilkan di kebijakan sebelumnya, dan pengguna yang menyiapkan logging memiliki `S3:PutBucketPolicy` izin `S3:GetBucketPolicy` dan untuk bucket, pernyataan tersebut akan ditambahkan ke kebijakan sumber daya bucket.

Note

Dalam beberapa kasus, Anda mungkin melihat `AccessDenied` kesalahan AWS CloudTrail jika `s3:ListBucket` izin belum diberikandelivery.logs.amazonaws.com. Untuk menghindari kesalahan ini di CloudTrail log Anda, Anda harus memberikan `s3:ListBucket` izin delivery.logs.amazonaws.com dan Anda harus menyertakan `Condition` parameter yang ditampilkan dengan `s3:GetBucketAcl` izin yang ditetapkan dalam kebijakan bucket sebelumnya. Untuk membuatnya lebih sederhana, alih-alih membuat yang baruStatement, Anda dapat langsung memperbarui `AWSLogDeliveryAclCheck` to be `"Action": ["s3:GetBucketAcl", "s3:ListBucket"]`

Enkripsi sisi server bucket Amazon S3

Anda dapat melindungi data di bucket Amazon S3 dengan mengaktifkan Enkripsi sisi server dengan kunci yang dikelola Amazon S3 (SSE-S3) atau enkripsi sisi server dengan kunci yang disimpan di (SSE-KMS). AWS KMS AWS Key Management Service Untuk informasi selengkapnya, silakan lihat [Melindungi data menggunakan enkripsi sisi server](#).

Jika Anda memilih SSE-S3, tidak diperlukan konfigurasi tambahan. Amazon S3 menangani kunci enkripsi.

Warning

Jika Anda memilih SSE-KMS, Anda harus menggunakan kunci yang dikelola pelanggan, karena menggunakan kunci AWS terkelola tidak didukung untuk skenario ini. Jika Anda mengatur enkripsi menggunakan kunci AWS terkelola, log akan dikirimkan dalam format yang tidak dapat dibaca.

Saat menggunakan AWS KMS kunci terkelola pelanggan, Anda dapat menentukan Nama Sumber Daya Amazon (ARN) kunci terkelola pelanggan saat mengaktifkan enkripsi bucket. Anda harus menambahkan hal berikut ke kebijakan kunci untuk kunci terkelola pelanggan Anda (bukan ke

kebijakan bucket untuk bucket S3 Anda), sehingga akun pengiriman log dapat menulis ke bucket S3 Anda.

Jika Anda memilih SSE-KMS, Anda harus menggunakan kunci yang dikelola pelanggan, karena menggunakan kunci AWS terkelola tidak didukung untuk skenario ini. Saat menggunakan AWS KMS kunci terkelola pelanggan, Anda dapat menentukan Nama Sumber Daya Amazon (ARN) kunci terkelola pelanggan saat mengaktifkan enkripsi bucket. Anda harus menambahkan hal berikut ke kebijakan kunci untuk kunci terkelola pelanggan Anda (bukan ke kebijakan bucket untuk bucket S3 Anda), sehingga akun pengiriman log dapat menulis ke bucket S3 Anda.

```
{
  "Sid": "Allow Logs Delivery to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": [ "delivery.logs.amazonaws.com" ]
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": ["0123456789"]
    },
    "ArnLike": {
      "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:delivery-source:*"]
    }
  }
}
```

Untuk `aws:SourceAccount`, tentukan daftar ID akun tempat log dikirimkan ke bucket ini.

Untuk `aws:SourceArn`, tentukan daftar ARNs sumber daya yang menghasilkan log, dalam formulir `arn:aws:logs:source-region:source-account-id:*`.

Log dikirim ke Firehose

Izin pengguna

Untuk mengaktifkan pengiriman log ke Firehose, Anda harus masuk dengan izin berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadWriteAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:GetDelivery",
        "logs:GetDeliverySource",
        "logs:PutDeliveryDestination",
        "logs:GetDeliveryDestinationPolicy",
        "logs>DeleteDeliverySource",
        "logs:PutDeliveryDestinationPolicy",
        "logs>CreateDelivery",
        "logs:GetDeliveryDestination",
        "logs:PutDeliverySource",
        "logs>DeleteDeliveryDestination",
        "logs>DeleteDeliveryDestinationPolicy",
        "logs>DeleteDelivery",
        "logs:UpdateDeliveryConfiguration"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:delivery:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
      ]
    },
    {
      "Sid": "ListAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeDeliveryDestinations",
        "logs:DescribeDeliverySources",
        "logs:DescribeDeliveries",
        "logs:DescribeConfigurationTemplates"
      ],
      "Resource": "*"
    }
  ]
}
```

```

        "Sid": "AllowUpdatesToResourcePolicyFH",
        "Effect": "Allow",
        "Action": [
            "firehose:TagDeliveryStream"
        ],
        "Resource": [
            "arn:aws:firehose:us-east-1:111122223333:deliverystream/*"
        ]
    },
    {
        "Sid": "CreateServiceLinkedRole",
        "Effect": "Allow",
        "Action": [
            "iam:CreateServiceLinkedRole"
        ],
        "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery"
    }
]
}

```

Peran IAM yang digunakan untuk izin sumber daya

Karena Firehose tidak menggunakan kebijakan sumber daya, AWS menggunakan peran IAM saat menyiapkan log ini untuk dikirim ke Firehose. AWS membuat peran terkait layanan bernama. `AWSServiceRoleForLogDelivery` Peran terkait layanan ini mencakup izin berikut.

JSON

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {

```

```

        "aws:ResourceTag/LogDeliveryEnabled": "true"
      },
    },
    "Effect": "Allow"
  }
]
}

```

Peran terkait layanan ini memberikan izin untuk semua aliran pengiriman Firehose yang memiliki tag yang disetel ke `LogDeliveryEnabled true`. AWS memberikan tag ini ke aliran pengiriman tujuan saat Anda mengatur logging.

Peran terkait layanan ini juga memiliki kebijakan kepercayaan yang memungkinkan layanan `delivery.logs.amazonaws.com` utama untuk mengasumsikan peran yang terhubung dengan layanan yang diperlukan. Kebijakan kepercayaan tersebut adalah sebagai berikut:

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

Jejak dikirim ke X-Ray

Izin pengguna

Untuk mengaktifkan pengiriman jejak AWS X-Ray, Anda harus masuk dengan izin berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadWriteAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:GetDelivery",
        "logs:GetDeliverySource",
        "logs:PutDeliveryDestination",
        "logs:GetDeliveryDestinationPolicy",
        "logs>DeleteDeliverySource",
        "logs:PutDeliveryDestinationPolicy",
        "logs>CreateDelivery",
        "logs:GetDeliveryDestination",
        "logs:PutDeliverySource",
        "logs>DeleteDeliveryDestination",
        "logs>DeleteDeliveryDestinationPolicy",
        "logs>DeleteDelivery",
        "logs:UpdateDeliveryConfiguration"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:delivery:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-source:*",
        "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
      ]
    },
    {
      "Sid": "ListAccessForLogDeliveryActions",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeDeliveryDestinations",
        "logs:DescribeDeliverySources",
        "logs:DescribeDeliveries",
        "logs:DescribeConfigurationTemplates"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowUpdatesToResourcePolicyXRay",
      "Effect": "Allow",
```

```

        "Action": [
            "xray:PutResourcePolicy",
            "xray:ListResourcePolicies",
            "xray:GetTraceSegmentDestination"
        ],
        "Resource": "*"
    }
}

```

Kebijakan sumber daya X-Ray

Akun tujuan tempat jejak dikirim harus memiliki kebijakan sumber daya yang menyertakan izin tertentu. Saat pengguna mengatur pelacakan memiliki `xray:PutResourcePolicy` dan `xray:ListResourcePolicies` izin di akun, AWS secara otomatis membuat kebijakan sumber daya saat Anda mulai mengirim jejak ke X-Ray. Kebijakan yang dibuat tergantung pada layanan sumber:

Amazon Bedrock AgentCore sumber daya

AWS membuat satu kebijakan sumber daya per jenis sumber daya. Kebijakan ini menggunakan pola wildcard yang tercakup pada batas akun, yang mencakup semua sumber daya dari jenis sumber daya yang sama Amazon Bedrock AgentCore di akun. Misalnya, jika sumber daya Amazon Bedrock AgentCoreMemori diaktifkan untuk pengiriman jejak, kebijakan tersebut mencakup semua sumber daya memori di akun tersebut — termasuk sumber daya memori apa pun yang dibuat di masa mendatang.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "xray:PutTraceSegments",
      "Resource": "*",
      "Condition": {

```

```

    "StringEquals": {
      "aws:SourceAccount": "123456789012"
    },
    "ForAllValues:ArnLike": {
      "logs:LogGeneratingResourceArns": "arn:aws:bedrock-agentcore:us-
east-1:123456789012:memory/*"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:logs:us-east-1:123456789012:delivery-
source:*"
    }
  }
}
]
}

```

AWS Layanan lainnya

Untuk layanan lain yang mendukung pengiriman jejak, AWS buat kebijakan sumber daya yang dicakup ke sumber daya tertentu.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "xray:PutTraceSegments",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ForAllValues:ArnLike": {
          "logs:LogGeneratingResourceArns": "arn:aws:bedrock:us-
east-1:123456789012:knowledge-base/KnowledgeBaseId"
        },
        "ArnLike": {

```

```

      "aws:SourceArn": "arn:aws:logs:us-east-1:123456789012:delivery-
source:xray-test"
    }
  }
}
]
}

```

Aktifkan pencarian transaksi

Untuk mengaktifkan pengiriman jejak ke X-Ray, Anda harus mengaktifkan [pencarian transaksi](#).

Izin khusus layanan

Selain izin khusus tujuan yang tercantum di bagian sebelumnya, beberapa layanan memerlukan otorisasi eksplisit bahwa pelanggan diizinkan mengirim log dari sumber daya mereka, sebagai lapisan keamanan tambahan. Ini mengotorisasi `AllowVendedLogDeliveryForResource` tindakan untuk sumber daya yang menjual log dalam layanan itu. Untuk layanan ini, gunakan kebijakan berikut dan ganti *service* dan *resource-type* dengan nilai yang sesuai. Untuk nilai khusus layanan untuk bidang ini, lihat halaman dokumentasi layanan tersebut untuk log penjual. Dalam contoh berikut, kebijakan telah diperbarui untuk mengaktifkan log vended dari Amazon SES.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ServiceLevelAccessForLogDelivery",
      "Effect": "Allow",
      "Action": [
        "ses:AllowVendedLogDeliveryForResource"
      ],
      "Resource": "arn:aws:ses:us-east-1:123456789012:resource-type/*"
    }
  ]
}

```

Izin khusus konsol

Selain izin yang tercantum di bagian sebelumnya, jika Anda menyiapkan pengiriman log menggunakan konsol alih-alih APIs, Anda juga memerlukan izin tambahan berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowLogDeliveryActionsConsoleCWL",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups",
        "logs:CreateLogGroup"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:*"
      ]
    },
    {
      "Sid": "AllowLogDeliveryActionsConsoleS3",
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    },
    {
      "Sid": "AllowLogDeliveryActionsConsoleFH",
      "Effect": "Allow",
      "Action": [
        "firehose:ListDeliveryStreams",
        "firehose:DescribeDeliveryStream"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```



```

    ]
  }
]
}

```

Contoh pengiriman lintas akun

Dalam contoh ini, dua akun terlibat. Akun dengan sumber daya penghasil log adalah Akun A, ID: **123456789012**, dan akun dengan sumber daya yang memakan log adalah Akun B, ID: **111122223333**

Akun A ingin mengirimkan log dari basis Amazon Bedrock pengetahuan di akun mereka dengan ARN `arn:aws:bedrock::knowledge-base/. us-east-1 123456789012 kb-12345678`

Untuk contoh ini, akun A memerlukan izin berikut:

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowVendedLogDeliveryForKnowledgeBase",
      "Effect": "Allow",
      "Action": [
        "bedrock:AllowVendedLogDeliveryForResource"
      ],
      "Resource": "arn:aws:bedrock:us-east-1:123456789012:knowledge-  
base/XXXXXXXXXX"
    },
    {
      "Sid": "CreateLogDeliveryPermissions",
      "Effect": "Allow",
      "Action": [
        "logs:PutDeliverySource",
        "logs:CreateDelivery"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
        "arn:aws:logs:us-east-1:123456789012:delivery:*",
        "arn:aws:logs:us-east-1:444455556666:delivery-destination:*"
      ]
    }
  ]
}

```

```
    ]
  }
]
}
```

Buat sumber pengiriman

Untuk memulai, akun A membuat sumber pengiriman dengan basis pengetahuan dasar mereka:

```
aws logs put-delivery-source --name my-delivery-source --log-type APPLICATION_LOGS --
resource-arn arn:aws:bedrock:region:AAAAAAAAAAAA:knowledge-base/XXXXXXXXXXXX
```

Selanjutnya, akun B harus membuat tujuan pengiriman menggunakan salah satu alur di bawah ini:

- [Konfigurasi pengiriman ke bucket Amazon S3](#)
- [Konfigurasi pengiriman ke aliran Firehose](#)

Konfigurasi pengiriman ke bucket Amazon S3

Akun B ingin menerima log ke bucket S3 mereka dengan ARN `arn:aws:s3:::amzn-s3-demo-bucket`

Untuk contoh ini, akun B akan memerlukan izin berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PutLogDestinationPermissions",
      "Effect": "Allow",
      "Action": [
        "logs:PutDeliveryDestination",
        "logs:PutDeliveryDestinationPolicy"
      ],
      "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-
destination:*"
    }
  ]
}
```

Bucket akan memerlukan izin berikut dalam kebijakan bucket:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSLogsDeliveryWrite",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/AWSLogs/123456789012/
*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control",
          "aws:SourceAccount": [
            "123456789012"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:delivery-source:my-
delivery-source"
          ]
        }
      }
    }
  ]
}
```

Jika bucket dienkripsi dengan SSE-KMS, pastikan kebijakan AWS KMS kunci memiliki izin yang sesuai. Misalnya, jika kunci KMS `arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`, gunakan yang berikut ini:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowLogsGenerateDataKey",
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey"
      ],
      "Resource": "arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "123456789012"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:delivery-source:my-delivery-source"
          ]
        }
      }
    }
  ]
}
```

Akun B kemudian dapat membuat tujuan pengiriman dengan bucket S3 sebagai sumber daya tujuan:

```
aws logs put-delivery-destination --name my-s3-delivery-destination --delivery-destination-configuration "destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket"
```

Selanjutnya, Akun B membuat kebijakan tujuan pengiriman pada tujuan pengiriman yang baru dibuat, yang akan memberikan izin bagi Akun A untuk membuat pengiriman log. Kebijakan yang akan ditambahkan ke tujuan pengiriman yang baru dibuat adalah sebagai berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateDelivery",
      "Effect": "Allow",
      "Principal": {
        "AWS": "123456789012"
      },
      "Action": [
        "logs:CreateDelivery"
      ],
      "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-destination:amzn-s3-demo-bucket"
    }
  ]
}
```

Kebijakan ini akan disimpan di komputer Akun B sebagai `destination-policy-s3.json` Untuk melampirkan sumber daya ini, Akun B akan menjalankan perintah berikut:

```
aws logs put-delivery-destination-policy --delivery-destination-name my-s3-delivery-destination --delivery-destination-policy file://destination-policy-s3.json
```

Terakhir, Akun A membuat pengiriman, yang menautkan sumber pengiriman di Akun A ke tujuan pengiriman di Akun B.

```
aws logs create-delivery --delivery-source-name my-delivery-source --delivery-destination-arn arn:aws:logs:region:BBBBBBBBBBBB:delivery-destination:my-s3-delivery-destination
```

Konfigurasi pengiriman ke aliran Firehose

Dalam contoh ini, Akun B ingin menerima log ke aliran Firehose mereka. Aliran Firehose memiliki ARN berikut dan dikonfigurasi untuk menggunakan jenis aliran pengiriman: `DirectPut`

```
arn:aws:firehose:us-east-1:111122223333:deliverystream/log-delivery-stream
```

Untuk contoh ini, Akun B memerlukan izin berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowFirehoseCreateSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "arn:aws:iam::111122223333:role/aws-service-role/delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery"
    },
    {
      "Sid": "AllowFirehoseTagging",
      "Effect": "Allow",
      "Action": [
        "firehose:TagDeliveryStream"
      ],
      "Resource": "arn:aws:firehose:us-east-1:111122223333:deliverystream/X"
    },
    {
      "Sid": "AllowFirehoseDeliveryDestination",
      "Effect": "Allow",
      "Action": [
        "logs:PutDeliveryDestination",
        "logs:PutDeliveryDestinationPolicy"
      ],
      "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-destination:*"
    }
  ]
}
```

Aliran Firehose harus memiliki tag yang LogDeliveryEnabled disetel ke true

Akun B kemudian akan membuat tujuan pengiriman dengan aliran Firehose sebagai sumber daya tujuan:

```
aws logs put-delivery-destination --name my-fh-delivery-destination --delivery-destination-configuration "destinationResourceArn=arn:aws:firehose:region:BBBBBBBBBBBB:deliverystream/X"
```

Selanjutnya, Akun B membuat kebijakan tujuan pengiriman pada tujuan pengiriman yang baru dibuat, yang akan memberikan izin bagi Akun A untuk membuat pengiriman log. Kebijakan yang akan ditambahkan ke tujuan pengiriman yang baru dibuat adalah sebagai berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreateDelivery",
      "Effect": "Allow",
      "Principal": {
        "AWS": "123456789012"
      },
      "Action": [
        "logs:CreateDelivery"
      ],
      "Resource": "arn:aws:logs:us-east-1:111122223333:delivery-destination:amzn-s3-demo-bucket"
    }
  ]
}
```

Kebijakan ini akan disimpan di komputer Akun B sebagai `destination-policy-fh.json`. Untuk melampirkan sumber daya ini, Akun B menjalankan perintah berikut:

```
aws logs put-delivery-destination-policy --delivery-destination-name my-fh-delivery-destination --delivery-destination-policy file://destination-policy-fh.json
```

Terakhir, Akun A membuat pengiriman, yang menautkan sumber pengiriman di Akun A ke tujuan pengiriman di Akun B.

```
aws logs create-delivery --delivery-source-name my-delivery-source --delivery-destination-arn arn:aws:logs:region:BBBBBBBBBBBB:delivery-destination:my-fh-delivery-destination
```

Pencegahan "confused deputy" lintas layanan

Masalah "confused deputy" adalah masalah keamanan di mana entitas yang tidak memiliki izin untuk melakukan tindakan dapat memengaruhi entitas yang memiliki hak akses lebih tinggi untuk melakukan tindakan. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS sediakan alat yang membantu Anda melindungi data Anda untuk semua layanan dengan prinsip layanan yang telah diberikan akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi

[aws:SourceArn](#), [aws:SourceAccount](#), [aws:SourceOrgID](#), dan [aws:SourceOrgPaths](#) global dalam kebijakan sumber daya untuk membatasi izin yang diberikan CloudWatch Log kepada layanan lain ke sumber daya. Gunakan [aws:SourceArn](#) untuk mengaitkan hanya satu sumber daya dengan akses lintas layanan. Gunakan [aws:SourceAccount](#) untuk membiarkan sumber daya apa pun di akun itu dikaitkan dengan penggunaan lintas layanan. Gunakan [aws:SourceOrgID](#) untuk memungkinkan sumber daya apa pun dari akun apa pun dalam suatu organisasi dikaitkan dengan penggunaan lintas layanan. Gunakan [aws:SourceOrgPaths](#) untuk mengaitkan sumber daya apa pun dari akun dalam AWS Organizations jalur dengan penggunaan lintas layanan. Untuk informasi selengkapnya tentang menggunakan dan memahami jalur, lihat [Memahami jalur AWS Organizations entitas](#).

Cara paling efektif untuk melindungi dari masalah "confused deputy" adalah dengan menggunakan kunci konteks kondisi global [aws:SourceArn](#) dengan ARN lengkap sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci kondisi konteks global [aws:SourceArn](#) dengan karakter wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, `arn:aws:servicename:*:123456789012:*`.

Jika [aws:SourceArn](#) nilainya tidak berisi ID akun, seperti ARN bucket Amazon S3, Anda harus menggunakan keduanya [aws:SourceAccount](#) dan [aws:SourceArn](#) untuk membatasi izin.

Untuk melindungi dari masalah wakil yang membingungkan dalam skala besar, gunakan `aws:SourceOrgID` atau kunci konteks kondisi `aws:SourceOrgPaths` global dengan ID organisasi atau jalur organisasi sumber daya dalam kebijakan berbasis sumber daya Anda. Kebijakan yang menyertakan `aws:SourceOrgID` atau `aws:SourceOrgPaths` kunci akan secara otomatis menyertakan akun yang benar dan Anda tidak perlu memperbarui kebijakan secara manual saat menambahkan, menghapus, atau memindahkan akun di organisasi Anda.

Kebijakan yang didokumentasikan untuk memberikan akses ke CloudWatch Log untuk menulis data ke Kinesis Data Streams dan Firehose [Langkah 1: Buat tujuan](#) di [Langkah 2: Buat tujuan](#) dan menunjukkan bagaimana Anda dapat menggunakan kunci konteks kondisi global untuk membantu `aws:SourceArn` mencegah masalah deputi yang membingungkan.

Otomatiskan pengaktifan log dengan aturan pengaktifan telemetri

Anda dapat menggunakan aturan pengaktifan telemetri untuk mengonfigurasi pengumpulan log secara otomatis untuk sumber daya Anda. AWS Aturan membantu Anda menstandarisasi pengumpulan log di seluruh organisasi atau akun Anda dan memastikan cakupan pemantauan yang konsisten.

Aturan pemberdayaan memungkinkan Anda:

- Secara otomatis mengaktifkan pencatatan untuk sumber daya baru dan yang sudah ada yang cocok dengan cakupan aturan Anda
- Menerapkan aturan di organisasi, unit organisasi (OU), atau tingkat akun individu
- Filter sumber daya mana yang memengaruhi aturan menggunakan tag

Aturan pengaktifan saat ini mendukung sumber AWS telemetri berikut: Log Aliran VPC Amazon, Log, Log Kueri Penyelesai Rute 53, AWS WAF Log Akses NLB, Log Pesawat Kontrol Amazon EKS, Acara Data CloudTrail dan Manajemen, Log Batuan Dasar Amazon, Metrik Terperinci Amazon AgentCore EC2, Security Hub, Amazon Bedrock Gateway, Memori Batuan Dasar Amazon, dan AWS Distribusi. AgentCore AgentCore CloudFront

Untuk detail lengkap tentang aturan pemberdayaan termasuk perilaku aturan, mengelola aturan, pemecahan masalah, dan pertimbangan khusus layanan, lihat Aturan [pemberdayaan telemetri](#) di Panduan Pengguna Amazon. CloudWatch

Hirarki evaluasi aturan

Aturan pemberdayaan dievaluasi secara hierarkis: aturan organisasi terlebih dahulu, kemudian aturan tingkat OU, kemudian aturan tingkat akun. Aturan di tingkat yang lebih tinggi menyediakan telemetri dasar. Aturan tingkat rendah dapat menambahkan telemetri tambahan tetapi tidak dapat mengurangnya. Jika aturan yang bertentangan ada pada ruang lingkup yang sama, tidak ada yang diterapkan sampai konflik diselesaikan.

Membuat aturan pemberdayaan

Untuk membuat aturan pemberdayaan telemetri

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Di panel navigasi, pilih konfigurasi Telemetri.
3. Pilih tab Aturan Pengaktifan.
4. Pilih Tambahkan aturan.
5. Tentukan nama aturan, ruang lingkup (organisasi, OU, atau akun), sumber data, dan jenis telemetri.
6. Secara opsional tambahkan tag untuk memfilter sumber daya mana yang dipengaruhi aturan.
7. Pilih Buat aturan.

Aktifkan logging dari sumber pihak ketiga

CloudWatch Log mendukung pengambilan log dari sumber pihak ketiga melalui integrasi API langsung dan integrasi bucket Amazon S3. Anda juga dapat menerima temuan keamanan pihak ketiga tambahan melalui AWS Security Hub CSPM.

Integrasi pihak ketiga langsung

CloudWatch Log menyediakan integrasi langsung dengan sumber pihak ketiga berikut. Integrasi ini menggunakan koneksi API langsung atau integrasi bucket Amazon S3 untuk memasukkan log ke dalam Log: CloudWatch

- CrowdStrike Falcon
- Microsoft Office 365
- Okta Auth0
- ID Microsoft Entra
- Jaringan Palo Alto NGFW
- Acara Microsoft Windows
- Wiz
- Akses Internet Zscaler
- Okta SSO
- SentinelOne Keamanan Endpoint
- GitHub Log Audit
- ServiceNow CMDB
- Payung Cisco

Untuk petunjuk penyiapan, lihat [panduan penyiapan integrasi pihak ketiga](#) di Panduan CloudWatch Pengguna Amazon.

Sumber tambahan melalui AWS Security Hub CSPM

Selain integrasi langsung, mitra keamanan pihak ketiga mengirim temuan ke AWS Security Hub CSPM, yang kemudian tersedia sebagai sumber data di Log. CloudWatch Tabel berikut mencantumkan integrasi mitra CSPM Security Hub dan jenis integrasinya.

Untuk mengaktifkan temuan CSPM Security Hub sebagai sumber data di CloudWatch Log, buat aturan pengaktifan telemetri untuk Security Hub AWS di konsol. CloudWatch Aturan pemberdayaan dikonfigurasi CloudWatch untuk secara otomatis mencerna temuan dari Security Hub CSPM ke dalam grup log terkelola. Untuk step-by-step petunjuknya, lihat [Aturan pengaktifan telemetri](#) di Panduan Pengguna Amazon CloudWatch .

Mitra	Integrasi
3 CORESec — NTA	Mengirimkan temuan melalui Security Hub CSPM
Logika Peringatan - Manajemen SIEMless Ancaman	Mengirimkan temuan melalui Security Hub CSPM
Aqua Security - Platform Keamanan Asli Cloud	Mengirimkan temuan melalui Security Hub CSPM
Aqua Security — Bangku Kube	Mengirimkan temuan melalui Security Hub CSPM
Armor - Armor Dimana Saja	Mengirimkan temuan melalui Security Hub CSPM
AttackIQ	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Barracuda - Penjaga Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
BiGid — BiGid Perusahaan	Mengirimkan temuan melalui Security Hub CSPM
Hexagon Biru	Mengirimkan temuan melalui Security Hub CSPM
Titik Periksa - CloudGuard IaaS	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
Check Point - Manajemen CloudGuard Postur	Mengirimkan temuan melalui Security Hub CSPM
Claroty — xDome	Mengirimkan temuan melalui Security Hub CSPM
Keamanan Penyimpanan Cloud - Antivirus untuk Amazon S3	Mengirimkan temuan melalui Security Hub CSPM
Keamanan Kontras - Penilaian Kontras	Mengirimkan temuan melalui Security Hub CSPM
CrowdStrike — CrowdStrike Elang	Mengirimkan temuan melalui Security Hub CSPM
CyberArk — Analisis Ancaman Istimewa	Mengirimkan temuan melalui Security Hub CSPM
Teorema Data	Mengirimkan temuan melalui Security Hub CSPM
Drata	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - CASB	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - Gerbang Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - DLP	Mengirimkan temuan melalui Security Hub CSPM
Forcepoint - NGFW	Mengirimkan temuan melalui Security Hub CSPM
Fugue	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
Guardicore — Pusat	Mengirimkan temuan melalui Security Hub CSPM
HackerOne — Kecerdasan Kerentanan	Mengirimkan temuan melalui Security Hub CSPM
JFrog — Xray	Mengirimkan temuan melalui Security Hub CSPM
Juniper Networks — Firewall Generasi Berikutnya VsRx	Mengirimkan temuan melalui Security Hub CSPM
k9 Keamanan - Penganalisis Akses	Mengirimkan temuan melalui Security Hub CSPM
Lacework	Mengirimkan temuan melalui Security Hub CSPM
McAfee — MVISION CNAPP	Mengirimkan temuan melalui Security Hub CSPM
NETSCOUT — Penyelidik Cyber	Mengirimkan temuan melalui Security Hub CSPM
Orca - Platform Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Palo Alto - Komputasi Awan Prisma	Mengirimkan temuan melalui Security Hub CSPM
Jaringan Palo Alto - Perusahaan Awan Prisma	Mengirimkan temuan melalui Security Hub CSPM
Plerion - Platform Keamanan Cloud	Mengirimkan temuan melalui Security Hub CSPM
Prowler	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
Qualys - Manajemen Kerentanan	Mengirimkan temuan melalui Security Hub CSPM
Rapid7 - InsightVM	Mengirimkan temuan melalui Security Hub CSPM
SentinelOne	Mengirimkan temuan melalui Security Hub CSPM
Snyk	Mengirimkan temuan melalui Security Hub CSPM
Keamanan Sonrai - Sonrai Dig	Mengirimkan temuan melalui Security Hub CSPM
Sophos - Perlindungan Server	Mengirimkan temuan melalui Security Hub CSPM
StackRox — Keamanan Kubernetes	Mengirimkan temuan melalui Security Hub CSPM
Logika Sumo - Analisis Data Mesin	Mengirimkan temuan melalui Security Hub CSPM
Symantec - Perlindungan Beban Kerja Cloud	Mengirimkan temuan melalui Security Hub CSPM
Tenable.io	Mengirimkan temuan melalui Security Hub CSPM
Trend Micro - Cloud One	Mengirimkan temuan melalui Security Hub CSPM
Vectra - Deteksi Cognito	Mengirimkan temuan melalui Security Hub CSPM
Wiz	Mengirimkan temuan melalui Security Hub CSPM

Mitra	Integrasi
Caveonix - Awan Caveonix	Mengirim dan menerima temuan melalui Security Hub CSPM
Penjaga Awan	Mengirim dan menerima temuan melalui Security Hub CSPM
DisruptOps	Mengirim dan menerima temuan melalui Security Hub CSPM
Kion	Mengirim dan menerima temuan melalui Security Hub CSPM
Turbot	Mengirim dan menerima temuan melalui Security Hub CSPM

 Note

Daftar ini mencerminkan integrasi mitra Security Hub yang mengirimkan temuan pada saat penulisan. Karena AWS Security Hub secara teratur menambahkan integrasi mitra baru, lihat [integrasi produk Pihak Ketiga dengan Security Hub CSPM](#) di Panduan Pengguna AWS Security Hub untuk up-to-date daftar mitra terbanyak yang tersedia.

Mengekspor data log ke Amazon S3

Bab ini memberi Anda informasi, sehingga Anda dapat mengekspor data log dari grup log ke bucket Amazon S3 untuk pemrosesan dan analisis kustom, atau memuat ke sistem lain. Anda dapat mengekspor ke bucket S3 di akun yang sama atau akun lain.

Anda dapat melakukan tindakan berikut:

- Eksport data log ke bucket S3 yang dienkripsi oleh SSE-KMS di () AWS Key Management Service AWS KMS
- Eksport data log ke bucket S3 yang mengaktifkan Kunci Objek S3 dengan periode retensi

Kami menyarankan agar Anda tidak secara teratur mengekspor ke Amazon S3 sebagai cara untuk terus mengarsipkan log Anda. Untuk kasus penggunaan itu, kami menyarankan Anda menggunakan langganan. Untuk informasi selengkapnya tentang langganan, lihat [Pemrosesan data log secara real-time dengan langganan](#).

Untuk memulai proses ekspor, Anda harus membuat bucket S3 untuk menyimpan data log yang diekspor. Anda dapat menyimpan file yang diekspor di bucket S3 dan menentukan aturan siklus hidup Amazon S3 untuk mengarsipkan atau menghapus file yang diekspor secara otomatis.

Anda dapat mengekspor ke bucket S3 yang dienkripsi dengan AES-256 atau dengan SSE-KMS. Mengekspor ke bucket yang dienkripsi dengan DSSE-KMS tidak didukung.

Anda dapat mengekspor log dari beberapa grup log atau beberapa rentang waktu ke bucket S3 yang sama. Untuk mengatur data yang diekspor, tentukan awalan untuk setiap tugas ekspor yang akan digunakan sebagai awalan key Amazon S3 untuk semua objek yang diekspor. Misalnya `prod/app-logs/2026-01-03/` atau `log-group-name/backup/`

Note

Penyortiran berbasis waktu pada potongan data log di dalam file yang diekspor tidak dijamin. Anda dapat mengurutkan data bidang log yang diekspor dengan menggunakan utilitas Linux. Misalnya, perintah utilitas berikut mengurutkan peristiwa di semua `.gz` file dalam satu folder.

```
find . -exec zcat {} + | sed -r 's/^[0-9]+\x0&/' | sort -z
```

Perintah utilitas berikut mengurutkan file.gz dari beberapa subfolder.

```
find ./* -type f -exec zcat {} + | sed -r 's/^[0-9]+\x0&/' | sort -z
```

Selain itu, Anda dapat menggunakan `stdout` perintah lain untuk menyalurkan output yang diurutkan ke file lain untuk menyimpannya.

Data log dapat memakan waktu hingga 12 jam agar tersedia untuk diekspor. Waktu tugas ekspor habis setelah 24 jam. Jika tugas ekspor Anda habis waktu, kurangi rentang waktu saat Anda membuat tugas ekspor.

Untuk analisis data log secara hampir waktu nyata, lihat [Menganalisis data log dengan Wawasan CloudWatch Log](#) atau [Pemrosesan data log secara real-time dengan langganan](#).

Daftar Isi

- [Konsep](#)
- [Ekspor data log ke Amazon S3 menggunakan konsol](#)
- [Ekspor data log ke Amazon S3 menggunakan AWS CLI](#)
- [Jelaskan tugas ekspor \(CLI\)](#)
- [Batalkan tugas ekspor \(CLI\)](#)

Konsep

Sebelum Anda mulai, pahami konsep ekspor berikut:

nama grup log

Nama grup log yang terkait dengan tugas ekspor. Data log dalam grup log ini akan diekspor ke bucket S3 yang ditentukan.

dari (stempel waktu)

Stempel waktu yang diperlukan dan dinyatakan sebagai angka milidetik sejak 1 Jan 1970 00:00:00 UTC. Semua peristiwa log dalam grup log yang tertelan pada atau setelah waktu ini akan diekspor.

ke (stempel waktu)

Stempel waktu yang diperlukan dan dinyatakan sebagai angka milidetik sejak 1 Jan 1970 00:00:00 UTC. Semua log acara dalam grup log yang diserap sebelum waktu ini akan diekspor.

bucket tujuan

Nama bucket S3 yang terkait dengan tugas ekspor. Bucket ini digunakan untuk mengekspor data log dari grup log yang ditentukan.

prefiks tujuan

Atribut opsional yang digunakan sebagai key prefix Amazon S3 untuk semua objek yang diekspor. Ini membantu membuat organisasi mirip folder di bucket Anda.

Ekspor data log ke Amazon S3 menggunakan konsol

Dalam contoh berikut, Anda menggunakan CloudWatch konsol Amazon untuk mengekspor semua data dari grup CloudWatch log Amazon Logs yang diberi nama `my-log-group` ke bucket Amazon S3 bernama `amzn-s3-demo-bucket`

Mengekspor data log ke bucket S3 yang dienkripsi oleh SSE-KMS didukung. Mengekspor ke bucket yang dienkripsi dengan DSSE-KMS tidak didukung.

Detail cara Anda mengatur ekspor tergantung pada apakah bucket Amazon S3 yang ingin Anda ekspor berada di akun yang sama dengan log Anda yang sedang diekspor, atau di akun lain.

Topik

- [Ekspor akun yang sama \(konsol\)](#)
- [Ekspor lintas akun \(konsol\)](#)

Ekspor akun yang sama (konsol)

Jika bucket Amazon S3 berada di akun yang sama dengan log yang sedang diekspor, gunakan instruksi di bagian ini.

Topik

- [Buat bucket Amazon S3 \(konsol\)](#)
- [Mengatur izin akses \(konsol\)](#)

- [Setel izin pada bucket Amazon S3 \(konsol\)](#)
- [\(Opsional\) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS \(konsol\)](#)
- [Buat tugas ekspor \(konsol\)](#)

Buat bucket Amazon S3 (konsol)

Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang sudah ada, Anda dapat melompat ke langkah 2.

Note

Bucket Amazon S3 harus berada di Wilayah yang sama dengan data log yang akan diekspor. CloudWatch Log tidak mendukung ekspor data ke bucket Amazon S3 di Wilayah lain.

Untuk membuat bucket Amazon S3

1. Buka konsol Amazon S3 di <https://console.aws.amazon.com/s3/>
2. Jika perlu, ubah Wilayah. Dari bilah navigasi, pilih Wilayah tempat CloudWatch Log Anda berada.
3. Pilih Create Bucket (Buat Bucket).
4. Untuk Bucket Name (Nama Bucket), masukkan nama untuk bucket.
5. Untuk Wilayah, pilih Wilayah tempat data CloudWatch Log Anda berada.
6. Pilih Buat.

Mengatur izin akses (konsol)

Untuk membuat tugas ekspor, Anda harus masuk dengan peran AmazonS3ReadOnlyAccess IAM dan dengan izin berikut:

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Setel izin pada bucket Amazon S3 (konsol)

Semua objek dan bucket Amazon S3 secara default bersifat privat. Hanya pemilik sumber daya, Akun AWS yang membuat ember, yang dapat mengakses ember dan objek apa pun yang dikandungnya. Namun, pemilik sumber daya dapat memilih untuk memberikan izin akses kepada sumber daya dan pengguna lain dengan menulis kebijakan akses.

Ketika Anda menetapkan kebijakan, sebaiknya Anda menyertakan string yang dihasilkan secara acak sebagai prefiks untuk bucket sehingga hanya pengaliran log yang dimaksud yang diekspor ke bucket tersebut.

Important

Untuk membuat ekspor ke bucket Amazon S3 lebih aman, kami sekarang meminta Anda untuk menentukan daftar akun sumber yang diizinkan untuk mengekspor data log ke bucket S3 Anda.

Dalam contoh berikut, daftar akun IDs di `aws:SourceAccount` kunci adalah akun tempat pengguna dapat mengekspor data log ke bucket Amazon S3 Anda. `aws:SourceArn` kuncinya adalah sumber daya tempat tindakan diambil. Anda dapat membatasi ini ke grup log tertentu, atau menggunakan wildcard seperti yang ditunjukkan dalam contoh ini.

Kami menyarankan Anda juga menyertakan ID akun tempat bucket S3 dibuat, untuk memungkinkan ekspor dalam akun yang sama.

Untuk mengatur izin bucket Amazon S3

1. Di konsol Amazon S3, pilih bucket yang Anda buat.
2. Pilih Permissions (Izin), Bucket policy (Kebijakan bucket).
3. Di Editor Kebijakan Bucket, tambahkan kebijakan berikut. Ubah `amzn-s3-demo-bucket` nama bucket S3 Anda. Pastikan untuk menentukan titik akhir Wilayah yang benar, seperti `us-east-1`, untuk Principal.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCloudWatchLogsGetBucketAcl",
      "Action": "s3:GetBucketAcl",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "123456789012",
            "111122223333"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:log-group:*",
            "arn:aws:logs:us-east-1:111122223333:log-group:*"
          ]
        }
      }
    },
    {
      "Sid": "AllowCloudWatchLogsPutObject",
```

```

    "Action": "s3:PutObject",
    "Effect": "Allow",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
    "Condition": {
      "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control",
        "aws:SourceAccount": [
          "123456789012",
          "111122223333"
        ]
      },
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:logs:us-east-1:123456789012:log-group:*",
          "arn:aws:logs:us-east-1:111122223333:log-group:*"
        ]
      }
    }
  }
}

```

4. Pilih Save (Simpan) untuk menetapkan kebijakan yang baru saja ditambahkan sebagai kebijakan akses di bucket Anda. Kebijakan ini memungkinkan CloudWatch Log untuk mengekspor data log ke bucket Amazon S3 Anda. Pemilik bucket memiliki izin penuh atas semua objek yang diekspor.

Warning

Jika bucket yang ada sudah memiliki satu atau beberapa kebijakan yang dilampirkan padanya, tambahkan pernyataan untuk akses CloudWatch Log ke kebijakan atau kebijakan tersebut. Sebaiknya Anda mengevaluasi hasil rangkaian izin untuk memastikan bahwa itu sesuai untuk pengguna yang akan mengakses bucket.

(Opsional) Mengekspor ke bucket Amazon S3 tujuan yang dienkrpsi dengan SSE-KMS (konsol)

Langkah ini diperlukan hanya jika Anda mengekspor ke bucket Amazon S3 yang menggunakan enkripsi sisi server. AWS KMS keys Enkripsi ini dikenal sebagai SSE-KMS.

Untuk mengekspor ke bucket yang dienkripsi dengan SSE-KMS

1. Buka AWS KMS konsol di <https://console.aws.amazon.com/kms>.
2. Untuk mengubah Wilayah AWS, gunakan pemilih Wilayah di sudut kanan atas halaman.
3. Di bilah navigasi kiri, pilih Kunci yang dikelola pelanggan.

Pilih Buat Kunci.
4. Untuk Tipe Kunci, pilih Simetris.
5. Untuk penggunaan Kunci, pilih Enkripsi dan dekripsi dan kemudian pilih Berikutnya.
6. Di bawah Tambahkan label, masukkan alias untuk kunci dan secara opsional tambahkan deskripsi atau tag. Lalu pilih Selanjutnya.
7. Di bawah Administrator kunci, pilih siapa yang dapat mengelola kunci ini, lalu pilih Berikutnya.
8. Di bawah Tentukan izin penggunaan kunci, jangan buat perubahan dan pilih Berikutnya.
9. Tinjau pengaturan dan pilih Selesai.
10. Kembali ke halaman kunci yang dikelola Pelanggan, pilih nama kunci yang baru saja Anda buat.
11. Pilih tab Kebijakan kunci dan pilih Beralih ke tampilan kebijakan.
12. Di bagian Kebijakan kunci, pilih Edit.
13. Tambahkan pernyataan berikut ke daftar pernyataan kebijakan kunci. Ketika Anda melakukannya, ganti *Region* dengan Wilayah log Anda dan ganti *account-ARN* dengan ARN dari akun yang memiliki kunci KMS.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow CWL Service Principal usage",
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.Region.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```



```

    },
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "account-ARN"
      },
      "Action": [
        "kms:GetKeyPolicy*",
        "kms:PutKeyPolicy*",
        "kms:DescribeKey*",
        "kms:CreateAlias*",
        "kms:ScheduleKeyDeletion*",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}

```

14. Pilih Simpan perubahan.
15. Buka konsol Amazon S3 di <https://console.aws.amazon.com/s3/>
16. Temukan bucket yang Anda buat [Buat ember S3 \(CLI\)](#) dan pilih nama bucket.
17. Pilih tab Properti. Kemudian, di bawah Enkripsi Default, pilih Edit.
18. Di bawah Enkripsi sisi server, pilih Aktifkan.
19. Di bawah Tipe enkripsi memilih Kunci (SSE-KMS)AWS Key Management Service .
20. Pilih Pilih dari AWS KMS kunci Anda dan temukan kunci yang Anda buat.
21. Untuk kunci Bucket, pilih Aktifkan.
22. Pilih Simpan perubahan.

Buat tugas ekspor (konsol)

Dalam prosedur ini, Anda membuat tugas ekspor untuk mengekspor log dari grup log.

Untuk mengekspor data ke Amazon S3 menggunakan konsol CloudWatch

1. Masuk dengan izin yang memadai seperti yang didokumentasikan[Mengatur izin akses \(konsol\)](#).
2. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
3. Pada panel navigasi, pilih Grup log.

4. Di layar Log Groups (Grup Log), pilih nama grup log.
5. Pilih Actions (Tindakan), Export data to Amazon S3 (Ekspor data ke Amazon S3).
6. Di layar Export data to Amazon S3 (Ekspor data ke Amazon S3), di Define data export (Tentukan ekspor data), atur rentang waktu untuk data yang akan diekspor menggunakan From (Dari) dan To (Sampai).
7. Jika grup log Anda memiliki beberapa pengaliran log, Anda dapat memberikan prefiks pengaliran log untuk membatasi data grup log ke pengaliran tertentu. Pilih Advanced (Lanjutan), lalu untuk Stream prefix (Prefiks pengaliran), masukkan prefiks pengaliran log.
8. Di bawah bucket Pilih S3, pilih akun yang terkait dengan bucket S3.
9. Untuk nama bucket S3, pilih bucket &S3;.
10. Untuk S3 Bucket prefix (Prefiks bucket S3), masukkan string yang dihasilkan secara acak yang Anda tentukan dalam kebijakan bucket.
11. Pilih Export (Ekspor) untuk mengeksport data log ke Amazon S3.
12. Untuk melihat status data log yang diekspor ke Amazon S3, pilih Actions (Tindakan), lalu View all exports to Amazon S3 (Lihat semua ekspor ke Amazon S3).

Ekspor lintas akun (konsol)

Jika bucket Amazon S3 berada di akun yang berbeda dari log yang sedang diekspor, gunakan petunjuk di bagian ini.

Topik

- [Buat bucket Amazon S3 untuk ekspor lintas akun \(konsol\)](#)
- [Menyiapkan izin akses untuk ekspor lintas akun \(konsol\)](#)
- [Tetapkan izin pada bucket S3 untuk ekspor lintas akun \(konsol\)](#)
- [\(Opsional\) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS untuk ekspor lintas akun \(konsol\)](#)
- [Buat tugas ekspor untuk ekspor lintas akun \(konsol\)](#)

Buat bucket Amazon S3 untuk ekspor lintas akun (konsol)

Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang ada, Anda dapat melewati prosedur ini.

 Note

Bucket Amazon S3 harus berada di Wilayah yang sama dengan data log yang akan diekspor. CloudWatch Log tidak mendukung ekspor data ke bucket Amazon S3 di Wilayah lain.

Untuk membuat bucket Amazon S3

1. Buka konsol Amazon S3 di <https://console.aws.amazon.com/s3/>
2. Jika perlu, ubah Wilayah. Dari bilah navigasi, pilih Wilayah tempat CloudWatch Log Anda berada.
3. Pilih Create Bucket (Buat Bucket).
4. Untuk Bucket Name (Nama Bucket), masukkan nama untuk bucket.
5. Untuk Wilayah, pilih Wilayah tempat data CloudWatch Log Anda berada.
6. Pilih Buat.

Menyiapkan izin akses untuk ekspor lintas akun (konsol)

Pertama, Anda harus membuat kebijakan IAM baru untuk mengaktifkan CloudWatch Log agar memiliki `s3:PutObject` tindakan untuk bucket Amazon S3 tujuan di akun tujuan.

Seiring dengan `s3:PutObject` tindakan, tindakan tambahan yang disertakan dalam kebijakan bergantung pada apakah bucket tujuan menggunakan AWS KMS enkripsi atau telah ACLs diaktifkan menggunakan setelan [Kepemilikan Objek S3](#).

- Jika menggunakan enkripsi KMS, tambahkan `kms:GenerateDataKey` dan `kms:Decrypt` tindakan untuk sumber daya utama
- Jika ACLs diaktifkan di bucket, tambahkan `s3:PutObjectAcl` tindakan untuk sumber daya bucket

Ubah `amzn-s3-demo-bucket` nama bucket S3 tujuan Anda dalam kebijakan berikut.

Untuk membuat kebijakan IAM untuk mengekspor log ke bucket Amazon S3

1. Buka konsol IAM di <https://console.aws.amazon.com/iam/>.
2. Di panel navigasi sebelah kiri, pilih Kebijakan.

3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih JSON.
5. Jika bucket tujuan tidak menggunakan AWS KMS enkripsi, tempelkan kebijakan berikut ke editor.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}
```

Jika bucket tujuan menggunakan AWS KMS enkripsi, tempelkan kebijakan berikut ke editor.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:us-east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}
```

```
]
}
```

Jika ACLs diaktifkan pada bucket tujuan, tambahkan s3: PutObjectAcl ke s3: PutObject Action block pada kebijakan di atas.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}
```

6. Pilih Berikutnya.
7. Masukkan nama kebijakan. Anda akan menggunakan nama ini untuk melampirkan kebijakan ke peran IAM Anda.
8. Pilih Buat kebijakan untuk menyimpan kebijakan baru.

Untuk membuat tugas ekspor, Anda harus masuk dengan peran IAM yang memiliki kebijakan AmazonS3ReadOnlyAccess terkelola yang dilampirkan, kebijakan IAM yang dibuat di atas, dan juga dengan izin berikut:

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Tetapkan izin pada bucket S3 untuk ekspor lintas akun (konsol)

Secara default, semua bucket dan objek S3 bersifat pribadi. Hanya pemilik sumber daya, Akun AWS yang membuat ember, yang dapat mengakses ember dan objek apa pun yang dikandungnya. Namun, pemilik sumber daya dapat memilih untuk memberikan izin akses kepada sumber daya dan pengguna lain dengan menulis kebijakan akses.

Ketika Anda menetapkan kebijakan, sebaiknya Anda menyertakan string yang dihasilkan secara acak sebagai prefiks untuk bucket sehingga hanya pengaliran log yang dimaksud yang diekspor ke bucket tersebut.

Important

Untuk membuat ekspor ke bucket S3 lebih aman, kami sekarang meminta Anda untuk menentukan daftar akun sumber yang diizinkan untuk mengekspor data log ke bucket S3 Anda.

Dalam contoh berikut, daftar akun IDs dalam `aws:SourceAccount` kunci akan menjadi akun dari mana pengguna dapat mengekspor data log ke bucket S3 Anda. `aws:SourceArn` kuncinya adalah sumber daya tempat tindakan diambil. Anda dapat membatasi ini ke grup log tertentu, atau menggunakan wildcard seperti yang ditunjukkan dalam contoh ini.

Kami menyarankan Anda juga menyertakan ID akun tempat bucket S3 dibuat, untuk memungkinkan ekspor dalam akun yang sama.

Untuk mengatur izin bucket Amazon S3

1. Di konsol Amazon S3, pilih bucket yang Anda buat.
2. Pilih Permissions (Izin), Bucket policy (Kebijakan bucket).
3. Di Editor Kebijakan Bucket, tambahkan kebijakan berikut. Ubah `amzn-s3-demo-bucket` nama bucket S3 Anda. Pastikan untuk menentukan titik akhir Wilayah yang benar, seperti `us-east-1`, untuk Principal.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "s3:GetBucketAcl",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "123456789012",
            "111122223333"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:log-group:*",
            "arn:aws:logs:us-east-1:111122223333:log-group:*"
          ]
        }
      }
    },
    {
      "Action": "s3:PutObject",
      "Effect": "Allow",
```

```

    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
    "Condition": {
      "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control",
        "aws:SourceAccount": [
          "123456789012",
          "111122223333"
        ]
      },
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:logs:us-east-1:123456789012:log-group:*",
          "arn:aws:logs:us-east-1:111122223333:log-group:*"
        ]
      }
    },
  },
  {
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:role/role_name"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control"
      }
    }
  }
]
}

```

4. Pilih Save (Simpan) untuk menetapkan kebijakan yang baru saja ditambahkan sebagai kebijakan akses di bucket Anda. Kebijakan ini memungkinkan CloudWatch Log untuk mengekspor data log ke bucket S3 Anda. Pemilik bucket memiliki izin penuh atas semua objek yang diekspor.

Warning

Jika bucket yang ada sudah memiliki satu atau beberapa kebijakan yang dilampirkan padanya, tambahkan pernyataan untuk akses CloudWatch Log ke kebijakan atau

kebijakan tersebut. Sebaiknya Anda mengevaluasi hasil rangkaian izin untuk memastikan bahwa itu sesuai untuk pengguna yang akan mengakses bucket.

(Opsional) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS untuk ekspor lintas akun (konsol)

Prosedur ini diperlukan hanya jika Anda mengekspor ke bucket S3 yang menggunakan enkripsi sisi server. AWS KMS keys Enkripsi ini dikenal sebagai SSE-KMS.

Untuk mengekspor ke bucket yang dienkripsi dengan SSE-KMS

1. Buka AWS KMS konsol di <https://console.aws.amazon.com/kms>.
2. Untuk mengubah Wilayah AWS, gunakan pemilih Wilayah di sudut kanan atas halaman.
3. Di bilah navigasi kiri, pilih Kunci yang dikelola pelanggan.

Pilih Buat Kunci.
4. Untuk Tipe Kunci, pilih Simetris.
5. Untuk penggunaan Kunci, pilih Enkripsi dan dekripsi dan kemudian pilih Berikutnya.
6. Di bawah Tambahkan label, masukkan alias untuk kunci dan secara opsional tambahkan deskripsi atau tag. Lalu pilih Selanjutnya.
7. Di bawah Administrator kunci, pilih siapa yang dapat mengelola kunci ini, lalu pilih Berikutnya.
8. Di bawah Tentukan izin penggunaan kunci, jangan buat perubahan dan pilih Berikutnya.
9. Tinjau pengaturan dan pilih Selesai.
10. Kembali ke halaman kunci yang dikelola Pelanggan, pilih nama kunci yang baru saja Anda buat.
11. Pilih tab Kebijakan kunci dan pilih Beralih ke tampilan kebijakan.
12. Di bagian Kebijakan kunci, pilih Edit.
13. Tambahkan pernyataan berikut ke daftar pernyataan kebijakan kunci. Ketika Anda melakukannya, ganti *us-east-1* dengan Wilayah log Anda, *account-ARN* dengan ARN akun yang memiliki kunci KMS, dengan nomor akun yang memiliki kunci KMS, *123456789012* dengan ID kms-key dan *key_id role_name* dengan peran yang digunakan untuk membuat tugas ekspor.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow CWL Service Principal usage",
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "account-ARN"
      },
      "Action": [
        "kms:GetKeyPolicy*",
        "kms:PutKeyPolicy*",
        "kms:DescribeKey*",
        "kms:CreateAlias*",
        "kms:ScheduleKeyDeletion*",
        "kms:Decrypt"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Enable IAM Role Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role_name"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ]
    }
  ]
}
```

```
    ],  
    "Resource": "arn:aws:kms:us-east-1:123456789012:key/key-id"  
  }  
]  
}
```

14. Pilih Simpan perubahan.
15. Buka konsol Amazon S3 di <https://console.aws.amazon.com/s3/>
16. Temukan bucket yang Anda buat [Buat ember S3 \(CLI\)](#) dan pilih nama bucket.
17. Pilih tab Properti. Kemudian, di bawah Enkripsi Default, pilih Edit.
18. Di bawah Enkripsi sisi server, pilih Aktifkan.
19. Di bawah Tipe enkripsi memilih Kunci (SSE-KMS)AWS Key Management Service .
20. Pilih Pilih dari AWS KMS kunci Anda dan temukan kunci yang Anda buat.
21. Untuk kunci Bucket, pilih Aktifkan.
22. Pilih Simpan perubahan.

Buat tugas ekspor untuk ekspor lintas akun (konsol)

Dalam prosedur ini, Anda membuat tugas ekspor untuk mengekspor log dari grup log.

Untuk mengekspor data ke Amazon S3 menggunakan konsol CloudWatch

1. Masuk dengan izin yang memadai seperti yang didokumentasikan[Mengatur izin akses \(konsol\)](#).
2. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
3. Pada panel navigasi, pilih Grup log.
4. Di layar Log Groups (Grup Log), pilih nama grup log.
5. Pilih Actions (Tindakan), Export data to Amazon S3 (Ekspor data ke Amazon S3).
6. Di layar Export data to Amazon S3 (Ekspor data ke Amazon S3), di Define data export (Tentukan ekspor data), atur rentang waktu untuk data yang akan diekspor menggunakan From (Dari) dan To (Sampai).
7. Jika grup log Anda memiliki beberapa pengaliran log, Anda dapat memberikan prefiks pengaliran log untuk membatasi data grup log ke pengaliran tertentu. Pilih Advanced (Lanjutan), lalu untuk Stream prefix (Prefiks pengaliran), masukkan prefiks pengaliran log.
8. Di bawah bucket Pilih S3, pilih akun yang terkait dengan bucket S3.

9. Untuk nama bucket S3, pilih bucket S3.
10. Untuk S3 Bucket prefix (Prefiks bucket S3), masukkan string yang dihasilkan secara acak yang Anda tentukan dalam kebijakan bucket.
11. Pilih Export (Ekspor) untuk mengeksport data log ke Amazon S3.
12. Untuk melihat status data log yang diekspor ke Amazon S3, pilih Actions (Tindakan), lalu View all exports to Amazon S3 (Lihat semua ekspor ke Amazon S3).

Ekspor data log ke Amazon S3 menggunakan AWS CLI

Dalam contoh berikut, Anda menggunakan tugas ekspor untuk mengeksport semua data dari grup CloudWatch log Log yang diberi nama `my-log-group` ke bucket Amazon S3 bernama `amzn-s3-demo-bucket`. Contoh ini mengasumsikan bahwa Anda telah membuat grup log bernama `my-log-group`.

Mengekspor data log ke bucket S3 yang dienkripsi oleh didukung. AWS KMS Mengekspor ke bucket yang dienkripsi dengan DSSE-KMS tidak didukung.

Detail cara Anda mengatur ekspor tergantung pada apakah bucket Amazon S3 yang ingin Anda ekspor berada di akun yang sama dengan log Anda yang sedang diekspor, atau di akun lain.

Topik

- [Ekspor akun yang sama \(CLI\)](#)
- [Ekspor lintas akun \(CLI\)](#)

Ekspor akun yang sama (CLI)

Jika bucket Amazon S3 berada di akun yang sama dengan log yang sedang diekspor, gunakan instruksi di bagian ini.

Topik

- [Buat ember S3 \(CLI\)](#)
- [Mengatur izin akses \(CLI\)](#)
- [Setel izin pada bucket S3 \(CLI\)](#)
- [\(Opsional\) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS \(CLI\)](#)

- [Buat tugas ekspor \(CLI\)](#)

Buat ember S3 (CLI)

Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang ada, Anda dapat melewati prosedur ini.

Note

Bucket S3 harus berada di Region yang sama dengan data log yang akan diekspor. CloudWatch Log tidak mendukung ekspor data ke bucket S3 di Wilayah lain.

Untuk membuat bucket S3 menggunakan AWS CLI

Di jendela perintah, jalankan perintah [create-bucket](#) berikut, di mana LocationConstraint adalah Wilayah tempat Anda mengeksport data log.

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration  
LocationConstraint=us-east-2
```

Berikut ini adalah output contoh.

```
{  
  "Location": "/amzn-s3-demo-bucket"  
}
```

Mengatur izin akses (CLI)

Untuk membuat tugas ekspor nanti, Anda harus masuk dengan peran AmazonS3ReadOnlyAccess IAM dan dengan izin berikut:

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Setel izin pada bucket S3 (CLI)

Secara default, semua bucket dan objek S3 bersifat pribadi. Hanya pemilik sumber daya, akun yang membuat bucket, yang dapat mengakses bucket dan objek yang ada di dalamnya. Namun, pemilik sumber daya dapat memilih untuk memberikan izin akses kepada sumber daya dan pengguna lain dengan menulis kebijakan akses.

Important

Untuk membuat ekspor ke bucket S3 lebih aman, kami sekarang meminta Anda untuk menentukan daftar akun sumber yang diizinkan untuk mengekspor data log ke bucket S3 Anda.

Dalam contoh berikut, daftar akun IDs dalam `aws:SourceAccount` kunci akan menjadi akun dari mana pengguna dapat mengekspor data log ke bucket S3 Anda. `aws:SourceArn` kuncinya adalah sumber daya tempat tindakan diambil. Anda dapat membatasi ini ke grup log tertentu, atau menggunakan wildcard seperti yang ditunjukkan dalam contoh ini.

Kami menyarankan Anda juga menyertakan ID akun tempat bucket S3 dibuat, untuk memungkinkan ekspor dalam akun yang sama.

Untuk menyetel izin pada bucket S3

1. Buat file bernama `policy.json` dan tambahkan kebijakan akses berikut, ubah `amzn-s3-demo-bucket` nama bucket S3 Anda dan `Principal` ke titik akhir Wilayah tempat Anda mengeksport data log, seperti `us-east-1`. Gunakan editor teks untuk membuat file kebijakan ini. Jangan gunakan konsol IAM.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGetBucketAcl",
      "Action": "s3:GetBucketAcl",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "123456789012",
            "111122223333"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:log-group:*",
            "arn:aws:logs:us-east-1:111122223333:log-group:*"
          ]
        }
      }
    },
    {
      "Sid": "AllowPutObject",
      "Action": "s3:PutObject",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
```

```

        "s3:x-amz-acl": "bucket-owner-full-control",
        "aws:SourceAccount": [
            "123456789012",
            "111122223333"
        ],
    },
    "ArnLike": {
        "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:log-group:*",
            "arn:aws:logs:us-east-1:111122223333:log-group:*"
        ]
    }
}
]
}
}

```

2. Tetapkan kebijakan yang baru saja ditambahkan sebagai kebijakan akses di bucket Anda dengan menggunakan [put-bucket-policy](#) perintah. Kebijakan ini memungkinkan CloudWatch Log untuk mengekspor data log ke bucket S3 Anda. Pemilik bucket akan memiliki izin penuh atas semua objek yang diekspor.

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json
```

Warning

Jika bucket yang ada sudah memiliki satu atau beberapa kebijakan yang dilampirkan padanya, tambahkan pernyataan untuk akses CloudWatch Log ke kebijakan atau kebijakan tersebut. Sebaiknya Anda mengevaluasi hasil rangkaian izin untuk memastikan bahwa itu sesuai untuk pengguna yang akan mengakses bucket.

(Opsional) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS (CLI)

Prosedur ini diperlukan hanya jika Anda mengekspor ke bucket S3 yang menggunakan enkripsi sisi server. AWS KMS keys Enkripsi ini dikenal sebagai SSE-KMS.

Untuk mengekspor ke bucket yang dienkripsi dengan SSE-KMS

1. Gunakan editor teks untuk membuat file bernama `key_policy.json` dan menambahkan kebijakan akses berikut. Saat Anda menambahkan kebijakan, lakukan perubahan berikut:

- Ganti *Region* dengan Wilayah log Anda.
- Ganti *account-ARN* dengan ARN akun yang memiliki kunci KMS.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow CWL Service Principal usage",
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.Region.amazonaws.com"
      },
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "account-ARN"
      },
      "Action": [
        "kms:GetKeyPolicy*",
        "kms:PutKeyPolicy*",
        "kms:DescribeKey*",
        "kms:CreateAlias*",
        "kms:ScheduleKeyDeletion*",
        "kms:Decrypt"
      ],
      "Resource": "*"
    }
  ]
}
```

```
]
}
```

2. Masukkan perintah berikut:

```
aws kms create-key --policy file:///key_policy.json
```

Berikut ini adalah contoh output dari perintah ini:

```
{
  "KeyMetadata": {
    "AWSAccountId": "account_id",
    "KeyId": "key_id",
    "Arn": "arn:aws:kms:us-east-2:account-ARN:key/key_id",
    "CreationDate": "time",
    "Enabled": true,
    "Description": "",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegion": false
  }
}
```

3. Gunakan editor teks untuk membuat file yang disebut `bucketencryption.json` dengan konten berikut.

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

```
}
```

4. Masukkan perintah berikut, ganti *amzn-s3-demo-bucket* dengan nama bucket tempat Anda mengeksport log.

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-encryption-configuration file://bucketencryption.json
```

Jika perintah tidak mengembalikan kesalahan, prosesnya berhasil.

Buat tugas ekspor (CLI)

Gunakan perintah berikut untuk membuat tugas ekspor. Setelah Anda membuatnya, tugas ekspor mungkin memakan waktu mulai dari beberapa detik hingga beberapa jam, tergantung pada ukuran data yang akan diekspor.

Untuk mengeksport data ke Amazon S3 menggunakan AWS CLI

1. Masuk dengan izin yang memadai seperti yang didokumentasikan [Mengatur izin akses \(CLI\)](#).
2. Pada prompt perintah, gunakan [create-export-task](#) perintah berikut untuk membuat tugas ekspor.

```
aws logs create-export-task --profile CWLExportUser --task-name "my-log-group-09-10-2015" --log-group-name "my-log-group" --from 1441490400000 --to 1441494000000 --destination "amzn-s3-demo-bucket" --destination-prefix "export-task-output"
```

Berikut ini adalah output contoh.

```
{  
  "taskId": "cda45419-90ea-4db5-9833-aade86253e66"  
}
```

Ekspor lintas akun (CLI)

Jika bucket Amazon S3 berada di akun yang berbeda dari log yang sedang diekspor, gunakan petunjuk di bagian ini.

Topik

- [Buat bucket S3 untuk ekspor lintas akun \(CLI\)](#)
- [Mengatur izin akses untuk ekspor lintas akun \(CLI\)](#)
- [Tetapkan izin pada bucket S3 untuk ekspor lintas akun \(CLI\)](#)
- [\(Opsional\) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS untuk ekspor lintas akun \(CLI\)](#)
- [Buat tugas ekspor untuk ekspor lintas akun \(CLI\)](#)

Buat bucket S3 untuk ekspor lintas akun (CLI)

Kami menyarankan Anda menggunakan bucket yang dibuat khusus untuk CloudWatch Log. Namun, jika Anda ingin menggunakan bucket yang sudah ada, Anda dapat melompat ke langkah 2.

Note

Bucket S3 harus berada di Region yang sama dengan data log yang akan diekspor. CloudWatch Log tidak mendukung ekspor data ke bucket S3 di Wilayah lain.

Untuk membuat bucket S3 menggunakan AWS CLI

Di jendela perintah, jalankan perintah [create-bucket](#) berikut, di mana LocationConstraint adalah Wilayah tempat Anda mengekspor data log.

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration  
LocationConstraint=us-east-2
```

Berikut ini adalah output contoh.

```
{  
  "Location": "/amzn-s3-demo-bucket"  
}
```

Mengatur izin akses untuk ekspor lintas akun (CLI)

Pertama, Anda harus membuat kebijakan IAM baru untuk mengaktifkan CloudWatch Log agar memiliki `s3:PutObject` tindakan untuk bucket Amazon S3 tujuan di akun tujuan.

Seiring dengan `s3:PutObject` tindakan, tindakan tambahan yang disertakan dalam kebijakan bergantung pada apakah bucket tujuan menggunakan AWS KMS enkripsi atau telah ACLs diaktifkan menggunakan setelan [Kepemilikan Objek S3](#).

- Jika menggunakan enkripsi KMS, tambahkan `kms:GenerateDataKey` dan `kms:Decrypt` tindakan untuk sumber daya utama
- Jika ACLs diaktifkan di bucket, tambahkan `s3:PutObjectAcl` tindakan untuk sumber daya bucket

Ubah `amzn-s3-demo-bucket` nama bucket S3 tujuan Anda dalam kebijakan berikut.

Kebijakan yang Anda buat bergantung pada apakah bucket tujuan menggunakan AWS KMS enkripsi. Jika tidak menggunakan AWS KMS enkripsi, buat kebijakan dengan konten berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}
```

Jika bucket tujuan menggunakan AWS KMS enkripsi, buat kebijakan dengan konten berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  },
```

```

    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:us-  
east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}

```

Jika ACLs diaktifkan pada bucket tujuan, tambahkan s3: PutObjectAcl ke s3: PutObject Action block pada kebijakan di atas.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}

```

Untuk membuat tugas ekspor, Anda harus masuk dengan peran IAM yang memiliki kebijakan AmazonS3ReadOnlyAccess terkelola yang dilampirkan, kebijakan IAM yang dibuat di atas, dan juga dengan izin berikut:

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks

- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Tetapkan izin pada bucket S3 untuk ekspor lintas akun (CLI)

Secara default, semua bucket dan objek S3 bersifat pribadi. Hanya pemilik sumber daya, akun yang membuat bucket, yang dapat mengakses bucket dan objek yang ada di dalamnya. Namun, pemilik sumber daya dapat memilih untuk memberikan izin akses kepada sumber daya dan pengguna lain dengan menulis kebijakan akses.

Important

Untuk membuat ekspor ke bucket S3 lebih aman, kami sekarang meminta Anda untuk menentukan daftar akun sumber yang diizinkan untuk mengekspor data log ke bucket S3 Anda.

Dalam contoh berikut, daftar akun IDs dalam `aws:SourceAccount` kunci akan menjadi akun dari mana pengguna dapat mengekspor data log ke bucket S3 Anda. `aws:SourceArn` kuncinya adalah sumber daya tempat tindakan diambil. Anda dapat membatasi ini ke grup log tertentu, atau menggunakan wildcard seperti yang ditunjukkan dalam contoh ini.

Kami menyarankan Anda juga menyertakan ID akun tempat bucket S3 dibuat, untuk memungkinkan ekspor dalam akun yang sama.

Untuk menyetel izin pada bucket S3

1. Buat file bernama `policy.json` dan tambahkan kebijakan akses berikut, ubah `amzn-s3-demo-bucket` nama bucket S3 tujuan Anda, `Principal` ke titik akhir Wilayah tempat Anda mengeksport data log, seperti `us-west-1`. Gunakan editor teks untuk membuat file kebijakan ini. Jangan gunakan konsol IAM.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "s3:GetBucketAcl",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": [
            "123456789012",
            "111122223333"
          ]
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:logs:us-east-1:123456789012:log-group:*",
            "arn:aws:logs:us-east-1:111122223333:log-group:*"
          ]
        }
      }
    },
    {
      "Action": "s3:PutObject",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
```



```

    "Principal": { "Service": "logs.us-east-1.amazonaws.com" },
    "Condition": {
      "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control",
        "aws:SourceAccount": [
          "123456789012",
          "111122223333"
        ]
      },
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:logs:us-east-1:123456789012:log-group:*",
          "arn:aws:logs:us-east-1:111122223333:log-group:*"
        ]
      }
    },
  },
  {
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:role/role_name"
    },
    "Action": "s3:PutObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
      "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control"
      }
    }
  }
]
}

```

2. Tetapkan kebijakan yang baru saja ditambahkan sebagai kebijakan akses di bucket Anda dengan menggunakan [put-bucket-policy](#) perintah. Kebijakan ini memungkinkan CloudWatch Log untuk mengeksport data log ke bucket S3 Anda. Pemilik bucket akan memiliki izin penuh atas semua objek yang diekspor.

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://policy.json
```

⚠ Warning

Jika bucket yang ada sudah memiliki satu atau beberapa kebijakan yang dilampirkan padanya, tambahkan pernyataan untuk akses CloudWatch Log ke kebijakan atau kebijakan tersebut. Sebaiknya Anda mengevaluasi hasil rangkaian izin untuk memastikan bahwa itu sesuai untuk pengguna yang akan mengakses bucket.

(Opsional) Mengekspor ke bucket Amazon S3 tujuan yang dienkripsi dengan SSE-KMS untuk ekspor lintas akun (CLI)

Prosedur ini diperlukan hanya jika Anda mengekspor ke bucket S3 yang menggunakan enkripsi sisi server. AWS KMS keys Enkripsi ini dikenal sebagai SSE-KMS.

Untuk mengekspor ke bucket yang dienkripsi dengan SSE-KMS

1. Gunakan editor teks untuk membuat file bernama `key_policy.json` dan menambahkan kebijakan akses berikut. Saat Anda menambahkan kebijakan, lakukan perubahan berikut:
 - Ganti `us-east-1` dengan Wilayah log Anda.
 - Ganti `account-ARN` dengan ARN akun yang memiliki kunci KMS.
 - Ganti `123456789012` dengan nomor akun yang memiliki kunci KMS.
 - `key_id` dengan Id kms-key.
 - `role_name` dengan peran yang digunakan untuk membuat tugas ekspor.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCWServicePrincipalUsage",
      "Effect": "Allow",
      "Principal": {
        "Service": "logs.us-east-1.amazonaws.com"
      },
    },
  ],
}
```

```

        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "*"
    },
    {
        "Sid": "EnableIAMUserPermissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "account-ARN"
        },
        "Action": [
            "kms:GetKeyPolicy*",
            "kms:PutKeyPolicy*",
            "kms:DescribeKey*",
            "kms:CreateAlias*",
            "kms:ScheduleKeyDeletion*",
            "kms:Decrypt"
        ],
        "Resource": "*"
    },
    {
        "Sid": "EnableIAMRolePermissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/role_name"
        },
        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
        ],
        "Resource": "arn:aws:kms:us-east-1:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
]
}

```

2. Masukkan perintah berikut:

```
aws kms create-key --policy file://key_policy.json
```

Berikut ini adalah contoh output dari perintah ini:

```
{
  "KeyMetadata": {
    "AWSAccountId": "account_id",
    "KeyId": "key_id",
    "Arn": "arn:aws:kms:us-east-1:123456789012:key/key_id",
    "CreationDate": "time",
    "Enabled": true,
    "Description": "",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegion": false
  }
}
```

- Gunakan editor teks untuk membuat file yang disebut `bucketencryption.json` dengan konten berikut.

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyId": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

- Masukkan perintah berikut, ganti *amzn-s3-demo-bucket* dengan nama bucket tempat Anda mengekspor log.

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-encryption-configuration file://bucketencryption.json
```

Jika perintah tidak mengembalikan kesalahan, prosesnya berhasil.

Buat tugas ekspor untuk ekspor lintas akun (CLI)

Gunakan perintah berikut untuk membuat tugas ekspor. Setelah Anda membuatnya, tugas ekspor mungkin memakan waktu mulai dari beberapa detik hingga beberapa jam, tergantung pada ukuran data yang akan diekspor.

Untuk mengekspor data ke Amazon S3 menggunakan AWS CLI

1. Masuk dengan izin yang memadai seperti yang didokumentasikan [Mengatur izin akses \(CLI\)](#).
2. Pada prompt perintah, gunakan [create-export-task](#) perintah berikut untuk membuat tugas ekspor.

```
aws logs create-export-task --profile CWLEXPOTUser --task-name "my-log-group-09-10-2015" --log-group-name "my-log-group" --from 1441490400000 --to 1441494000000 --destination "amzn-s3-demo-bucket" --destination-prefix "export-task-output"
```

Berikut ini adalah output contoh.

```
{
  "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

Jelaskan tugas ekspor (CLI)

Setelah Anda membuat tugas ekspor, Anda bisa mendapatkan status tugas saat ini.

Untuk menggambarkan tugas ekspor menggunakan AWS CLI

Pada prompt perintah, gunakan [describe-export-tasks](#) perintah berikut.

```
aws logs --profile CWLEXPOTUser describe-export-tasks --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

Berikut ini adalah output contoh.

```
{
```

```

"exportTasks": [
{
  "destination": "my-exported-logs",
  "destinationPrefix": "export-task-output",
  "executionInfo": {
    "creationTime": 1441495400000
  },
  "from": 1441490400000,
  "logGroupName": "my-log-group",
  "status": {
    "code": "RUNNING",
    "message": "Started Successfully"
  },
  "taskId": "cda45419-90ea-4db5-9833-aade86253e66",
  "taskName": "my-log-group-09-10-2015",
  "tTo": 1441494000000
}]
}

```

Anda dapat menggunakan perintah `describe-export-tasks` dalam tiga cara yang berbeda:

- Tanpa filter apa pun - Daftar semua tugas ekspor Anda, dalam urutan pembuatan terbalik.
- Filter pada ID tugas - Daftar tugas ekspor, jika ada, dengan ID yang ditentukan.
- Filter pada status tugas - Daftar tugas ekspor dengan status yang ditentukan.

Misalnya, gunakan perintah berikut untuk memfilter dengan status `FAILED`.

```
aws logs --profile CWLEXPOTUser describe-export-tasks --status-code "FAILED"
```

Berikut ini adalah output contoh.

```

{
  "exportTasks": [
    {
      "destination": "amzn-s3-demo-bucket",
      "destinationPrefix": "export-task-output",
      "executionInfo": {
        "completionTime": 1441498600000
        "creationTime": 1441495400000
      },
      "from": 1441490400000,

```

```
"logGroupName": "my-log-group",
"status": {
  "code": "FAILED",
  "message": "FAILED"
},
"taskId": "cda45419-90ea-4db5-9833-aade86253e66",
"taskName": "my-log-group-09-10-2015",
"to": 1441494000000
}]
}
```

Batalkan tugas ekspor (CLI)

Anda dapat membatalkan tugas ekspor jika dalam RUNNING status PENDING atau.

Untuk membatalkan tugas ekspor menggunakan AWS CLI

Pada prompt perintah, gunakan [cancel-export-task](#) perintah berikut:

```
aws logs --profile CWExportUser cancel-export-task --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

Anda dapat menggunakan [describe-export-tasks](#) perintah untuk memverifikasi bahwa tugas telah dibatalkan dengan sukses.

Streaming data CloudWatch Log ke OpenSearch Layanan Amazon

Anda dapat mengonfigurasi grup log di Amazon CloudWatch Logs, sehingga Anda dapat mengalirkan data ke kluster OpenSearch Layanan Amazon dalam waktu dekat. Untuk informasi selengkapnya, lihat [Pemrosesan data log secara real-time dengan langganan](#).

Note

Streaming ke OpenSearch Layanan hanya didukung untuk grup log di kelas log Standar. Untuk informasi selengkapnya tentang kelas log, lihat [Kelas log](#).

Bergantung pada jumlah data log yang dialirkan, pertimbangkan untuk menetapkan batas konkurensi tingkat fungsi. Untuk informasi selengkapnya, lihat Penskalaan [fungsi Lambda](#).

Note

Karena streaming data CloudWatch Log dalam jumlah besar ke OpenSearch Layanan dapat mengakibatkan biaya penggunaan yang tinggi, sebaiknya Anda membuat anggaran di AWS Manajemen Penagihan dan Biaya konsol. Untuk informasi selengkapnya, lihat [Mengelola biaya Anda dengan AWS Anggaran](#).

Bagian ini menjelaskan prasyarat yang harus Anda selesaikan sebelum berlangganan grup log ke Layanan. OpenSearch Ini juga menjelaskan cara berlangganan grup log ke OpenSearch Layanan.

Prasyarat

Sebelum memulai, buat domain OpenSearch Layanan. Domain dapat memiliki akses publik atau akses VPC, tetapi Anda tidak dapat mengubah jenis akses setelah domain dibuat. Anda mungkin ingin meninjau setelan domain OpenSearch Layanan nanti, dan memodifikasi konfigurasi kluster berdasarkan jumlah data yang akan diproses kluster Anda. Untuk petunjuk membuat domain, lihat [Membuat domain OpenSearch Layanan](#).

Untuk informasi selengkapnya tentang OpenSearch Layanan, lihat [Panduan Pengembang OpenSearch Layanan Amazon](#).

Berlangganan grup log ke OpenSearch Layanan

Anda dapat menggunakan CloudWatch konsol untuk berlangganan grup log ke OpenSearch Layanan.

Untuk berlangganan grup log ke OpenSearch Layanan

1. Buka CloudWatch konsol di <https://console.aws.amazon.com/cloudwatch/>.
2. Pada panel navigasi, pilih Grup log.
3. Pilih nama grup log.
4. Pilih Tindakan, Filter langganan, Buat filter langganan OpenSearch Layanan Amazon.
5. Pilih apakah Anda ingin melakukan pengaliran ke klaster di akun ini atau akun lain.
 - Jika Anda memilih akun ini, pilih domain yang Anda buat pada langkah sebelumnya.
 - Jika Anda memilih akun lain, berikan domain ARN dan endpoint.
6. Untuk Peran Eksekusi IAM Lambda, pilih peran IAM yang harus digunakan Lambda saat menjalankan panggilan. OpenSearch

IAM role yang Anda pilih harus memenuhi persyaratan berikut:

- Harus memiliki `lambda.amazonaws.com` dalam hubungan kepercayaan.
- Harus mencakup kebijakan berikut:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowOpenSearchStreamingAccess",
      "Action": [
        "es:*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:es:us-east-1:123456789012:domain/cloudwatch-logs/*"
    }
  ]
}
```

- Jika domain OpenSearch Layanan target menggunakan akses VPC, peran harus memiliki AWSLambdaVPCLambdaAccessExecutionRole kebijakan yang dilampirkan. Kebijakan yang dikelola Amazon ini memberi Lambda akses ke VPC pelanggan, memungkinkan Lambda untuk menulis ke titik akhir di VPC. OpenSearch
7. Untuk format Log, pilih format log.
 8. Untuk pola filter Langganan, ketikkan istilah atau pola yang akan ditemukan di peristiwa log Anda. Ini memastikan bahwa Anda hanya mengirim data yang Anda minati ke OpenSearch cluster Anda. Untuk informasi selengkapnya, lihat [Membuat metrik dari peristiwa log dengan menggunakan filter](#).
 9. (Opsional) Untuk Pilih data log yang akan diuji, pilih aliran log lalu pilih Pola uji untuk memverifikasi bahwa filter pencarian Anda mengembalikan hasil yang Anda harapkan.
 10. Pilih Mulai streaming.

Contoh kode untuk CloudWatch Log menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan CloudWatch Log dengan kit pengembangan AWS perangkat lunak (SDK).

Tindakan merupakan kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh kode

- [Contoh dasar untuk menggunakan CloudWatch Log AWS SDKs](#)
 - [Tindakan untuk CloudWatch Log menggunakan AWS SDKs](#)
 - [Gunakan AssociateKmsKey dengan AWS SDK](#)
 - [Gunakan CancelExportTask dengan AWS SDK](#)
 - [Gunakan CreateExportTask dengan AWS SDK](#)
 - [Gunakan CreateLogGroup dengan AWS SDK atau CLI](#)
 - [Gunakan CreateLogStream dengan AWS SDK atau CLI](#)
 - [Gunakan DeleteLogGroup dengan AWS SDK atau CLI](#)
 - [Gunakan DeleteSubscriptionFilter dengan AWS SDK](#)
 - [Gunakan DescribeExportTasks dengan AWS SDK](#)
 - [Gunakan DescribeLogGroups dengan AWS SDK atau CLI](#)
 - [Gunakan DescribeLogStreams dengan AWS SDK atau CLI](#)
 - [Gunakan DescribeSubscriptionFilters dengan AWS SDK](#)
 - [Gunakan GetLogEvents dengan AWS SDK atau CLI](#)
 - [Gunakan GetQueryResults dengan AWS SDK](#)

- [Gunakan PutSubscriptionFilter dengan AWS SDK](#)
- [Gunakan StartLiveTail dengan AWS SDK](#)
- [Gunakan StartQuery dengan AWS SDK](#)
- [Skenario untuk CloudWatch Log menggunakan AWS SDKs](#)
 - [Konfigurasi Amazon ECS Service Connect](#)
 - [Membuat fungsi Lambda pertama Anda](#)
 - [Gunakan CloudWatch Log untuk menjalankan kueri besar](#)
 - [Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda](#)

Contoh dasar untuk menggunakan CloudWatch Log AWS SDKs

Contoh kode berikut menunjukkan cara menggunakan dasar-dasar Amazon CloudWatch Logs dengan AWS SDKs.

Contoh

- [Tindakan untuk CloudWatch Log menggunakan AWS SDKs](#)
 - [Gunakan AssociateKmsKey dengan AWS SDK](#)
 - [Gunakan CancelExportTask dengan AWS SDK](#)
 - [Gunakan CreateExportTask dengan AWS SDK](#)
 - [Gunakan CreateLogGroup dengan AWS SDK atau CLI](#)
 - [Gunakan CreateLogStream dengan AWS SDK atau CLI](#)
 - [Gunakan DeleteLogGroup dengan AWS SDK atau CLI](#)
 - [Gunakan DeleteSubscriptionFilter dengan AWS SDK](#)
 - [Gunakan DescribeExportTasks dengan AWS SDK](#)
 - [Gunakan DescribeLogGroups dengan AWS SDK atau CLI](#)
 - [Gunakan DescribeLogStreams dengan AWS SDK atau CLI](#)
 - [Gunakan DescribeSubscriptionFilters dengan AWS SDK](#)
 - [Gunakan GetLogEvents dengan AWS SDK atau CLI](#)
 - [Gunakan GetQueryResults dengan AWS SDK](#)
 - [Gunakan PutSubscriptionFilter dengan AWS SDK](#)
 - [Gunakan StartLiveTail dengan AWS SDK](#)

- [Gunakan StartQuery dengan AWS SDK](#)

Tindakan untuk CloudWatch Log menggunakan AWS SDKs

Contoh kode berikut menunjukkan cara melakukan tindakan CloudWatch Log individual dengan AWS SDKs. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Kutipan ini memanggil CloudWatch Logs API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Anda dapat melihat tindakan dalam konteks di [Skenario untuk CloudWatch Log menggunakan AWS SDKs](#).

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat [Referensi API Amazon CloudWatch Logs](#).

Contoh

- [Gunakan AssociateKmsKey dengan AWS SDK](#)
- [Gunakan CancelExportTask dengan AWS SDK](#)
- [Gunakan CreateExportTask dengan AWS SDK](#)
- [Gunakan CreateLogGroup dengan AWS SDK atau CLI](#)
- [Gunakan CreateLogStream dengan AWS SDK atau CLI](#)
- [Gunakan DeleteLogGroup dengan AWS SDK atau CLI](#)
- [Gunakan DeleteSubscriptionFilter dengan AWS SDK](#)
- [Gunakan DescribeExportTasks dengan AWS SDK](#)
- [Gunakan DescribeLogGroups dengan AWS SDK atau CLI](#)
- [Gunakan DescribeLogStreams dengan AWS SDK atau CLI](#)
- [Gunakan DescribeSubscriptionFilters dengan AWS SDK](#)
- [Gunakan GetLogEvents dengan AWS SDK atau CLI](#)
- [Gunakan GetQueryResults dengan AWS SDK](#)
- [Gunakan PutSubscriptionFilter dengan AWS SDK](#)
- [Gunakan StartLiveTail dengan AWS SDK](#)
- [Gunakan StartQuery dengan AWS SDK](#)

Gunakan **AssociateKmsKey** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `AssociateKmsKey`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to associate an AWS Key Management Service (AWS KMS) key with
/// an Amazon CloudWatch Logs log group.
/// </summary>
public class AssociateKmsKey
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();

        string kmsKeyId = "arn:aws:kms:us-west-2:<account-
number>:key/7c9eccc2-38cb-4c4f-9db3-766ee8dd3ad4";
        string groupName = "cloudwatchlogs-example-loggroup";

        var request = new AssociateKmsKeyRequest
        {
            KmsKeyId = kmsKeyId,
            LogGroupName = groupName,
        };
    }
}
```

```
var response = await client.AssociateKmsKeyAsync(request);

if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
{
    Console.WriteLine($"Successfully associated KMS key ID:
{kmsKeyId} with log group: {groupName}.");
}
else
{
    Console.WriteLine("Could not make the association between:
{kmsKeyId} and {groupName}.");
}
}
```

- Untuk detail API, lihat [AssociateKmsKey](#) di Referensi AWS SDK untuk .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **CancelExportTask** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `CancelExportTask`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
```

```
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to cancel an Amazon CloudWatch Logs export task.
/// </summary>
public class CancelExportTask
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();
        string taskId = "exampleTaskId";

        var request = new CancelExportTaskRequest
        {
            TaskId = taskId,
        };

        var response = await client.CancelExportTaskAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"{taskId} successfully canceled.");
        }
        else
        {
            Console.WriteLine($"{taskId} could not be canceled.");
        }
    }
}
```

- Untuk detail API, lihat [CancelExportTask](#) di Referensi AWS SDK untuk .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **CreateExportTask** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `CreateExportTask`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to create an Export Task to export the contents of the Amazon
/// CloudWatch Logs to the specified Amazon Simple Storage Service (Amazon
S3)
/// bucket.
/// </summary>
public class CreateExportTask
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();
        string taskName = "export-task-example";
        string logGroupName = "cloudwatchlogs-example-loggroup";
        string destination = "amzn-s3-demo-bucket";
        var fromTime = 1437584472382;
        var toTime = 1437584472833;

        var request = new CreateExportTaskRequest
        {
```

```
        From = fromTime,
        To = toTime,
        TaskName = taskName,
        LogGroupName = logGroupName,
        Destination = destination,
    };

    var response = await client.CreateExportTaskAsync(request);

    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        Console.WriteLine($"The task, {taskName} with ID: " +
                           $"{response.TaskId} has been created
successfully.");
    }
}
```

- Untuk detail API, lihat [CreateExportTask](#) di Referensi AWS SDK untuk .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **CreateLogGroup** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `CreateLogGroup`.

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:

- [Konfigurasi Amazon ECS Service Connect](#)

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to create an Amazon CloudWatch Logs log group.
/// </summary>
public class CreateLogGroup
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();

        string logGroupName = "cloudwatchlogs-example-loggroup";

        var request = new CreateLogGroupRequest
        {
            LogGroupName = logGroupName,
        };

        var response = await client.CreateLogGroupAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"Successfully create log group with ID:
{logGroupName}.");
        }
    }
}
```

```
        else
        {
            Console.WriteLine("Could not create log group.");
        }
    }
}
```

- Untuk detail API, lihat [CreateLogGroup](#) di Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Perintah berikut membuat grup log bernama `my-logs`:

```
aws logs create-log-group --log-group-name my-logs
```

- Untuk detail API, lihat [CreateLogGroup](#) di Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import { CreateLogGroupCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
    const command = new CreateLogGroupCommand({
        // The name of the log group.
        logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
    });
```

```
try {
    return await client.send(command);
} catch (err) {
    console.error(err);
}
};

export default run();
```

- Untuk detail API, lihat [CreateLogGroup](#) di Referensi AWS SDK untuk JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **CreateLogStream** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `CreateLogStream`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to create an Amazon CloudWatch Logs stream for a CloudWatch
/// log group.
/// </summary>
public class CreateLogStream
{

```

```

public static async Task Main()
{
    // This client object will be associated with the same AWS Region
    // as the default user on this system. If you need to use a
    // different AWS Region, pass it as a parameter to the client
    // constructor.
    var client = new AmazonCloudWatchLogsClient();
    string logGroupName = "cloudwatchlogs-example-loggroup";
    string logStreamName = "cloudwatchlogs-example-logstream";

    var request = new CreateLogStreamRequest
    {
        LogGroupName = logGroupName,
        LogStreamName = logStreamName,
    };

    var response = await client.CreateLogStreamAsync(request);

    if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
    {
        Console.WriteLine($"{logStreamName} successfully created for
{logGroupName}.");
    }
    else
    {
        Console.WriteLine("Could not create stream.");
    }
}
}

```

- Untuk detail API, lihat [CreateLogStream](#) di Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Perintah berikut membuat aliran log bernama 20150601 dalam grup logmy-logs:

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- Untuk detail API, lihat [CreateLogStream](#) di Referensi AWS CLI Perintah.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DeleteLogGroup** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DeleteLogGroup`.

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:

- [Konfigurasi Amazon ECS Service Connect](#)
- [Membuat fungsi Lambda pertama Anda](#)

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Uses the Amazon CloudWatch Logs Service to delete an existing
/// CloudWatch Logs log group.
/// </summary>
public class DeleteLogGroup
{
    public static async Task Main()
    {
        var client = new AmazonCloudWatchLogsClient();
        string logGroupName = "cloudwatchlogs-example-loggroup";

        var request = new DeleteLogGroupRequest
```

```
{
    LogGroupName = logGroupName,
};

var response = await client.DeleteLogGroupAsync(request);

if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
{
    Console.WriteLine($"Successfully deleted CloudWatch log group,
{logGroupName}.");
}
}
```

- Untuk detail API, lihat [DeleteLogGroup](#) di Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Perintah berikut menghapus grup log bernama `my-logs`:

```
aws logs delete-log-group --log-group-name my-logs
```

- Untuk detail API, lihat [DeleteLogGroup](#) di Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import { DeleteLogGroupCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";
```



```
const run = async () => {
  const command = new DeleteLogGroupCommand({
    // The name of the log group.
    logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  });

  try {
    return await client.send(command);
  } catch (err) {
    console.error(err);
  }
};

export default run();
```

- Untuk detail API, lihat [DeleteLogGroup](#) di Referensi AWS SDK untuk JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DeleteSubscriptionFilter** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `DeleteSubscriptionFilter`.

C++

SDK untuk C++

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Sertakan file-file yang diperlukan.

```
#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>
```

```
#include <aws/logs/model/DeleteSubscriptionFilterRequest.h>
#include <iostream>
```

Hapus filter langganan.

```
Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
Aws::CloudWatchLogs::Model::DeleteSubscriptionFilterRequest request;
request.SetFilterName(filter_name);
request.SetLogGroupName(log_group);

auto outcome = cwl.DeleteSubscriptionFilter(request);
if (!outcome.IsSuccess()) {
    std::cout << "Failed to delete CloudWatch log subscription filter "
              << filter_name << ": " << outcome.GetError().GetMessage() <<
              std::endl;
} else {
    std::cout << "Successfully deleted CloudWatch logs subscription " <<
              "filter " << filter_name << std::endl;
}
```

- Untuk detail API, lihat [DeleteSubscriptionFilter](#) di Referensi AWS SDK untuk C++ API.

Java

SDK untuk Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.DeleteSubscriptionFilterRequest;

/**
```

```

* 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 DeleteSubscriptionFilter {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <filter> <logGroup>

            Where:
                filter - The name of the subscription filter (for example,
MyFilter).
                logGroup - The name of the log group. (for example, testgroup).
            "";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String filter = args[0];
        String logGroup = args[1];
        CloudWatchLogsClient logs = CloudWatchLogsClient.builder()
            .build();

        deleteSubFilter(logs, filter, logGroup);
        logs.close();
    }

    public static void deleteSubFilter(CloudWatchLogsClient logs, String filter,
String logGroup) {
        try {
            DeleteSubscriptionFilterRequest request =
DeleteSubscriptionFilterRequest.builder()
                .filterName(filter)
                .logGroupName(logGroup)
                .build();

```

```
        logs.deleteSubscriptionFilter(request);
        System.out.printf("Successfully deleted CloudWatch logs subscription
filter %s", filter);

    } catch (CloudWatchException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- Untuk detail API, lihat [DeleteSubscriptionFilter](#) di 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 mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import { DeleteSubscriptionFilterCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
    const command = new DeleteSubscriptionFilterCommand({
        // The name of the filter.
        filterName: process.env.CLOUDWATCH_LOGS_FILTER_NAME,
        // The name of the log group.
        logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
    });

    try {
        return await client.send(command);
    } catch (err) {
        console.error(err);
    }
};
```

```
export default run();
```

- Untuk detail API, lihat [DeleteSubscriptionFilter](#) di Referensi AWS SDK untuk 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 the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
  filterName: "FILTER",
  logGroupName: "LOG_GROUP",
};

cwl.deleteSubscriptionFilter(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK untuk JavaScript](#).
- Untuk detail API, lihat [DeleteSubscriptionFilter](#) di Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
suspend fun deleteSubFilter(  
    filter: String?,  
    logGroup: String?,  
) {  
    val request =  
        DeleteSubscriptionFilterRequest {  
            filterName = filter  
            logGroupName = logGroup  
        }  
  
    CloudWatchLogsClient.fromEnvironment { region = "us-west-2" }.use { logs ->  
        logs.deleteSubscriptionFilter(request)  
        println("Successfully deleted CloudWatch logs subscription filter named  
$filter")  
    }  
}
```

- Untuk detail API, lihat [DeleteSubscriptionFilter](#) di AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DescribeExportTasks** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `DescribeExportTasks`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Shows how to retrieve a list of information about Amazon CloudWatch
/// Logs export tasks.
/// </summary>
public class DescribeExportTasks
{
    public static async Task Main()
    {
        // This client object will be associated with the same AWS Region
        // as the default user on this system. If you need to use a
        // different AWS Region, pass it as a parameter to the client
        // constructor.
        var client = new AmazonCloudWatchLogsClient();

        var request = new DescribeExportTasksRequest
        {
            Limit = 5,
        };

        var response = new DescribeExportTasksResponse();

        do
        {
            response = await client.DescribeExportTasksAsync(request);
            response.ExportTasks.ForEach(t =>
            {
```

```

        Console.WriteLine($"{t.TaskName} with ID: {t.TaskId} has
status: {t.Status}");
    });
}
while (response.NextToken is not null);
}
}

```

- Untuk detail API, lihat [DescribeExportTasks](#) di Referensi AWS SDK untuk .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DescribeLogGroups** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DescribeLogGroups`.

.NET

SDK untuk .NET

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

using System;
using System.Threading.Tasks;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;

/// <summary>
/// Retrieves information about existing Amazon CloudWatch Logs log groups
/// and displays the information on the console.
/// </summary>
public class DescribeLogGroups
{

```



```
public static async Task Main()
{
    // Creates a CloudWatch Logs client using the default
    // user. If you need to work with resources in another
    // AWS Region than the one defined for the default user,
    // pass the AWS Region as a parameter to the client constructor.
    var client = new AmazonCloudWatchLogsClient();

    bool done = false;
    string newToken = null;

    var request = new DescribeLogGroupsRequest
    {
        Limit = 5,
    };

    DescribeLogGroupsResponse response;

    do
    {
        if (newToken is not null)
        {
            request.NextToken = newToken;
        }

        response = await client.DescribeLogGroupsAsync(request);

        response.LogGroups.ForEach(lg =>
        {
            Console.WriteLine($"{lg.LogGroupName} is associated with the
key: {lg.KmsKeyId}.");
            Console.WriteLine($"Created on:
{lg.CreationTime.Date.Date}");
            Console.WriteLine($"Date for this group will be stored for:
{lg.RetentionInDays} days.\n");
        });

        if (response.NextToken is null)
        {
            done = true;
        }
        else
        {
            newToken = response.NextToken;
        }
    }
}
```

```
        }  
    }  
    while (!done);  
}  
}
```

- Untuk detail API, lihat [DescribeLogGroups](#) di Referensi AWS SDK untuk .NET API.

CLI

AWS CLI

Perintah berikut menjelaskan grup log bernama `my-logs`:

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

Output:

```
{  
  "logGroups": [  
    {  
      "storedBytes": 0,  
      "metricFilterCount": 0,  
      "creationTime": 1433189500783,  
      "logGroupName": "my-logs",  
      "retentionInDays": 5,  
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"  
    }  
  ]  
}
```

- Untuk detail API, lihat [DescribeLogGroups](#) di Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import {
  paginateDescribeLogGroups,
  CloudWatchLogsClient,
} from "@aws-sdk/client-cloudwatch-logs";

const client = new CloudWatchLogsClient({});

export const main = async () => {
  const paginatedLogGroups = paginateDescribeLogGroups({ client }, {});
  const logGroups = [];

  for await (const page of paginatedLogGroups) {
    if (page.logGroups?.every((lg) => !!lg)) {
      logGroups.push(...page.logGroups);
    }
  }

  console.log(logGroups);
  return logGroups;
};
```

- Untuk detail API, lihat [DescribeLogGroups](#) di Referensi AWS SDK untuk JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DescribeLogStreams** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `DescribeLogStreams`.

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:

- [Membuat fungsi Lambda pertama Anda](#)

CLI

AWS CLI

Perintah berikut menunjukkan semua aliran log yang dimulai dengan awalan 2015 di grup log: `my-logs`

```
aws logs describe-log-streams --log-group-name my-logs --log-stream-name-prefix 2015
```

Output:

```
{
  "logStreams": [
    {
      "creationTime": 1433189871774,
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-stream:20150531",
      "logStreamName": "20150531",
      "storedBytes": 0
    },
    {
      "creationTime": 1433189873898,
      "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:log-stream:20150601",
      "logStreamName": "20150601",
      "storedBytes": 0
    }
  ]
}
```

- Untuk detail API, lihat [DescribeLogStreams](#) di Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Mencari aliran log dalam grup log tertentu yang cocok dengan awalan yang diberikan.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CloudWatchLogsSearch {

    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <logGroupName> <logStreamName>

            Where:
                logGroupName - The name of the log group (for example,
                WeathertopJavaContainerLogs).
                logStreamName - The name of the log stream (for example,
                weathertop-java-stream).
                pattern - the pattern to use (for example, INFO)

            """;

        if (args.length != 3) {
            System.out.print(usage);
            System.exit(1);
        }
    }
}
```

```

String logGroupName = args[0] ;
String logStreamName = args[1] ;
String pattern = args[2] ;

CloudWatchLogsClient cwlClient = CloudWatchLogsClient.builder()
    .region(Region.US_EAST_1)
    .build();

    searchLogStreamsAndFilterEvents(cwlClient, logGroupName, logStreamName,
pattern);
}

/**
 * Searches for log streams with a specific prefix within a log group and
filters log events based on a specified pattern.
 *
 * @param cwlClient      the CloudWatchLogsClient used to interact with AWS
CloudWatch Logs
 * @param logGroupName  the name of the log group to search within
 * @param logStreamPrefix the prefix of the log streams to search for
 * @param pattern       the pattern to filter log events by
 */
public static void searchLogStreamsAndFilterEvents(CloudWatchLogsClient
cwlClient, String logGroupName, String logStreamPrefix, String pattern) {
    DescribeLogStreamsRequest describeLogStreamsRequest =
DescribeLogStreamsRequest.builder()
        .logGroupName(logGroupName)
        .logStreamNamePrefix(logStreamPrefix)
        .build();

    DescribeLogStreamsResponse describeLogStreamsResponse =
cwlClient.describeLogStreams(describeLogStreamsRequest);
    List<LogStream> logStreams = describeLogStreamsResponse.logStreams();

    for (LogStream logStream : logStreams) {
        String logStreamName = logStream.logStreamName();
        System.out.println("Searching in log stream: " + logStreamName);

        FilterLogEventsRequest filterLogEventsRequest =
FilterLogEventsRequest.builder()
            .logGroupName(logGroupName)
            .logStreamNames(logStreamName)
            .filterPattern(pattern)

```

```

        .build();

        FilterLogEventsResponse filterLogEventsResponse =
        cwlClient.filterLogEvents(filterLogEventsRequest);

        for (FilteredLogEvent event : filterLogEventsResponse.events()) {
            System.out.println(event.message());
        }

        System.out.println("-----"); //
        Separator for better readability
    }
}
}

```

Mencetak metadata tentang aliran log terbaru dalam grup log tertentu.

```

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 * <p>
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CloudWatchLogQuery {
    public static void main(final String[] args) {
        final String usage = ""
            Usage:
                <logGroupName>

            Where:
                logGroupName - The name of the log group (for example, /aws/
                lambda/ChatAIHandler).
            "";

        if (args.length != 1) {
            System.out.print(usage);
            System.exit(1);
        }
    }
}

```

```

    }

    String logGroupName = "/aws/lambda/ChatAIHandler" ; //args[0];
    CloudWatchLogsClient logsClient = CloudWatchLogsClient.builder()
        .region(Region.US_EAST_1)
        .build();

    describeMostRecentLogStream(logsClient, logGroupName);
}

/**
 * Describes and prints metadata about the most recent log stream in the
 * specified log group.
 *
 * @param logsClient the CloudWatchLogsClient used to interact with AWS
 * CloudWatch Logs
 * @param logGroupName the name of the log group
 */
public static void describeMostRecentLogStream(CloudWatchLogsClient
logsClient, String logGroupName) {
    DescribeLogStreamsRequest streamsRequest =
DescribeLogStreamsRequest.builder()
        .logGroupName(logGroupName)
        .orderBy(OrderBy.LAST_EVENT_TIME)
        .descending(true)
        .limit(1)
        .build();

    try {
        DescribeLogStreamsResponse streamsResponse =
logsClient.describeLogStreams(streamsRequest);
        List<LogStream> logStreams = streamsResponse.logStreams();

        if (logStreams.isEmpty()) {
            System.out.println("No log streams found for log group: " +
logGroupName);
            return;
        }

        LogStream stream = logStreams.get(0);
        System.out.println("Most Recent Log Stream:");
        System.out.println("  Name: " + stream.logStreamName());
        System.out.println("  ARN: " + stream.arn());
        System.out.println("  Creation Time: " + stream.creationTime());
    }
}

```



```

        System.out.println("  First Event Time: " +
stream.firstEventTimestamp());
        System.out.println("  Last Event Time: " +
stream.lastEventTimestamp());
        System.out.println("  Stored Bytes: " + stream.storedBytes());
        System.out.println("  Upload Sequence Token: " +
stream.uploadSequenceToken());

    } catch (CloudWatchLogsException e) {
        System.err.println("Failed to describe log stream: " +
e.awsErrorDetails().errorMessage());
    }
}
}

```

- Untuk detail API, lihat [DescribeLogStreams](#) di Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **DescribeSubscriptionFilters** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `DescribeSubscriptionFilters`.

C++

SDK untuk C++

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Sertakan file-file yang diperlukan.

```

#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>

```

```
#include <aws/logs/model/DescribeSubscriptionFiltersRequest.h>
#include <aws/logs/model/DescribeSubscriptionFiltersResult.h>
#include <iostream>
#include <iomanip>
```

Buat daftar filter berlangganan.

```
Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
Aws::CloudWatchLogs::Model::DescribeSubscriptionFiltersRequest request;
request.SetLogGroupName(log_group);
request.SetLimit(1);

bool done = false;
bool header = false;
while (!done) {
    auto outcome = cwl.DescribeSubscriptionFilters(
        request);
    if (!outcome.IsSuccess()) {
        std::cout << "Failed to describe CloudWatch subscription filters
"
        << "for log group " << log_group << ": " <<
        outcome.GetError().GetMessage() << std::endl;
        break;
    }

    if (!header) {
        std::cout << std::left << std::setw(32) << "Name" <<
        std::setw(64) << "FilterPattern" << std::setw(64) <<
        "DestinationArn" << std::endl;
        header = true;
    }

    const auto &filters = outcome.GetResult().GetSubscriptionFilters();
    for (const auto &filter : filters) {
        std::cout << std::left << std::setw(32) <<
        filter.GetFilterName() << std::setw(64) <<
        filter.GetFilterPattern() << std::setw(64) <<
        filter.GetDestinationArn() << std::endl;
    }

    const auto &next_token = outcome.GetResult().GetNextToken();
    request.SetNextToken(next_token);
```

```
done = next_token.empty();
}
```

- Untuk detail API, lihat [DescribeSubscriptionFilters](#) di Referensi AWS SDK untuk C++ API.

Java

SDK untuk Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.DescribeSubscriptionFiltersRequest;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.DescribeSubscriptionFiltersResponse;
import software.amazon.awssdk.services.cloudwatchlogs.model.SubscriptionFilter;

/**
 * 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 DescribeSubscriptionFilters {
    public static void main(String[] args) {

        final String usage = ""

        Usage:
        <logGroup>
```

```

        Where:
            logGroup - A log group name (for example, myloggroup).
        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String logGroup = args[0];
    CloudWatchLogsClient logs = CloudWatchLogsClient.builder()
        .credentialsProvider(ProfileCredentialsProvider.create())
        .build();

    describeFilters(logs, logGroup);
    logs.close();
}

public static void describeFilters(CloudWatchLogsClient logs, String
logGroup) {
    try {
        boolean done = false;
        String newToken = null;

        while (!done) {
            DescribeSubscriptionFiltersResponse response;
            if (newToken == null) {
                DescribeSubscriptionFiltersRequest request =
DescribeSubscriptionFiltersRequest.builder()
                    .logGroupName(logGroup)
                    .limit(1).build();

                response = logs.describeSubscriptionFilters(request);
            } else {
                DescribeSubscriptionFiltersRequest request =
DescribeSubscriptionFiltersRequest.builder()
                    .nextToken(newToken)
                    .logGroupName(logGroup)
                    .limit(1).build();
                response = logs.describeSubscriptionFilters(request);
            }

            for (SubscriptionFilter filter : response.subscriptionFilters())
{

```

```

        System.out.printf("Retrieved filter with name %s, " +
            "pattern %s " + "and destination arn %s",
                filter.filterName(),
                filter.filterPattern(),
                filter.destinationArn());
    }

    if (response.nextToken() == null) {
        done = true;
    } else {
        newToken = response.nextToken();
    }
}

} catch (CloudWatchException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
System.out.printf("Done");
}
}

```

- Untuk detail API, lihat [DescribeSubscriptionFilters](#) di 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 mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

import { DescribeSubscriptionFiltersCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
    // This will return a list of all subscription filters in your account
    // matching the log group name.

```

```
const command = new DescribeSubscriptionFiltersCommand({
  logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
  limit: 1,
});

try {
  return await client.send(command);
} catch (err) {
  console.error(err);
}

};

export default run();
```

- Untuk detail API, lihat [DescribeSubscriptionFilters](#) di Referensi AWS SDK untuk 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 the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
  logGroupName: "GROUP_NAME",
  limit: 5,
};

cwl.describeSubscriptionFilters(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
```

```
console.log("Success", data.subscriptionFilters);
}
});
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK untuk JavaScript](#).
- Untuk detail API, lihat [DescribeSubscriptionFilters](#) di Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
suspend fun describeFilters(logGroup: String) {
    val request =
        DescribeSubscriptionFiltersRequest {
            logGroupName = logGroup
            limit = 1
        }

    CloudWatchLogsClient.fromEnvironment { region = "us-west-2" }.use { cwlClient
->
        val response = cwlClient.describeSubscriptionFilters(request)
        response.subscriptionFilters?.forEach { filter ->
            println("Retrieved filter with name ${filter.filterName} pattern
${filter.filterPattern} and destination ${filter.destinationArn}")
        }
    }
}
```

- Untuk detail API, lihat [DescribeSubscriptionFilters](#) di AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **GetLogEvents** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan `GetLogEvents`.

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:

- [Membuat fungsi Lambda pertama Anda](#)

CLI

AWS CLI

Perintah berikut mengambil peristiwa log dari aliran log bernama `20150601` dalam grup `my-logs` log:

```
aws logs get-log-events --log-group-name my-logs --log-stream-name 20150601
```

Output:

```
{
  "nextForwardToken":
  "f/31961209122447488583055879464742346735121166569214640130",
  "events": [
    {
      "ingestionTime": 1433190494190,
      "timestamp": 1433190184356,
      "message": "Example Event 1"
    },
    {
      "ingestionTime": 1433190516679,
      "timestamp": 1433190184356,
      "message": "Example Event 1"
    },
    {
      "ingestionTime": 1433190494190,
      "timestamp": 1433190184358,
      "message": "Example Event 2"
    }
  ]
}
```



```

    }
  ],
  "nextBackwardToken":
  "b/31961209122358285602261756944988674324553373268216709120"
}

```

- Untuk detail API, lihat [GetLogEvents](#) di Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatch.model.CloudWatchException;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
  software.amazon.awssdk.services.cloudwatchlogs.model.DescribeLogStreamsRequest;
import
  software.amazon.awssdk.services.cloudwatchlogs.model.DescribeLogStreamsResponse;
import software.amazon.awssdk.services.cloudwatchlogs.model.GetLogEventsRequest;
import software.amazon.awssdk.services.cloudwatchlogs.model.GetLogEventsResponse;

import java.time.Instant;
import java.time.temporal.ChronoUnit;

/**
 * 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 GetLogEvents {

```

```

public static void main(String[] args) {

    final String usage = ""

        Usage:
            <logGroupName> <logStreamName>

        Where:
            logGroupName - The name of the log group (for example,
myloggroup).
            logStreamName - The name of the log stream (for example,
mystream).

        """;

    // if (args.length != 2) {
    //     System.out.print(usage);
    //     System.exit(1);
//     }

    String logGroupName = "WeathertopJavaContainerLogs" ; //args[0];
    String logStreamName = "weathertop-java-stream" ; //args[1];

    Region region = Region.US_EAST_1 ;
    CloudWatchLogsClient cloudWatchLogsClient =
CloudWatchLogsClient.builder()
        .region(region)
        .build();

    getCWLogEvents(cloudWatchLogsClient, logGroupName, logStreamName);
    cloudWatchLogsClient.close();
}

public static void getCWLogEvents(CloudWatchLogsClient cloudWatchLogsClient,
                                String logGroupName,
                                String logStreamPrefix) {

    try {
        // First, find the exact log stream name
        DescribeLogStreamsRequest describeRequest =
DescribeLogStreamsRequest.builder()
            .logGroupName(logGroupName)
            .logStreamNamePrefix(logStreamPrefix)
            .limit(1) // get the first matching stream
            .build();

```

```

        DescribeLogStreamsResponse describeResponse =
cloudWatchLogsClient.describeLogStreams(describeRequest);

        if (describeResponse.logStreams().isEmpty()) {
            System.out.println("No matching log streams found for prefix: " +
logStreamPrefix);
            return;
        }

        String exactLogStreamName =
describeResponse.logStreams().get(0).logStreamName();
        System.out.println("Using exact log stream: " + exactLogStreamName);

        long startTime = Instant.now().minus(7,
ChronoUnit.DAYS).toEpochMilli();
        long endTime = Instant.now().toEpochMilli();

        GetLogEventsRequest getLogEventsRequest =
GetLogEventsRequest.builder()
            .logGroupName(logGroupName)
            .logStreamName(exactLogStreamName) // <-- exact name, not
prefix
            .startTime(startTime)
            .endTime(endTime)
            .startFromHead(true)
            .build();

        GetLogEventsResponse response =
cloudWatchLogsClient.getLogEvents(getLogEventsRequest);

        if (response.events().isEmpty()) {
            System.out.println("No log events found in the past 7 days.");
        } else {
            response.events().forEach(e -> System.out.println(e.message()));
        }

    } catch (CloudWatchException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}

```

- Untuk detail API, lihat [GetLogEvents](#) di Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **GetQueryResults** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `GetQueryResults`.

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:

- [Jalankan kueri besar](#)

.NET

SDK untuk .NET (v4)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// <summary>
/// Gets the results of a CloudWatch Logs Insights query.
/// </summary>
/// <param name="queryId">The ID of the query.</param>
/// <returns>The query results response.</returns>
public async Task<GetQueryResultsResponse?> GetQueryResultsAsync(string
queryId)
{
    try
    {
        var request = new GetQueryResultsRequest
        {
            QueryId = queryId
        }
    }
}
```

```

        };

        var response = await
        _amazonCloudWatchLogs.GetQueryResultsAsync(request);
        return response;
    }
    catch (ResourceNotFoundException ex)
    {
        _logger.LogError($"Query not found: {ex.Message}");
        return null;
    }
    catch (Exception ex)
    {
        _logger.LogError($"An error occurred while getting query results:
{ex.Message}");
        return null;
    }
}

```

- Untuk detail API, lihat [GetQueryResults](#) di Referensi AWS SDK untuk .NET API.

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
    return this.client.send(new GetQueryResultsCommand({ queryId }));
}

```

- Untuk detail API, lihat [GetQueryResults](#) di Referensi AWS SDK untuk JavaScript API.

Python

SDK untuk Python (Boto3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
def _wait_for_query_results(self, client, query_id):
    """
    Waits for the query to complete and retrieves the results.

    :param query_id: The ID of the initiated query.
    :type query_id: str
    :return: A list containing the results of the query.
    :rtype: list
    """
    while True:
        time.sleep(1)
        results = client.get_query_results(queryId=query_id)
        if results["status"] in [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ]:
            return results.get("results", [])
```

- Untuk detail API, lihat [GetQueryResults](#) di AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
TRY.  
    oo_result = lo_cwl->getqueryresults(  
        iv_queryid = iv_query_id ).  
  
    " Display query status and result count  
    DATA(lv_status) = oo_result->get_status( ).  
    DATA(lt_results) = oo_result->get_results( ).  
    DATA(lv_result_count) = lines( lt_results ).  
  
    MESSAGE |Query status: { lv_status }. Retrieved { lv_result_count } log  
event(s).| TYPE 'I'.  
    CATCH /aws1/cx_cwlinvalidparameterex.  
        MESSAGE 'Invalid parameter.' TYPE 'E'.  
    CATCH /aws1/cx_cwlresourcenotfoundex.  
        MESSAGE 'Resource not found.' TYPE 'E'.  
    CATCH /aws1/cx_cwlserviceunavailex.  
        MESSAGE 'Service unavailable.' TYPE 'E'.  
ENDTRY.
```

- Untuk detail API, lihat [GetQueryResults](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **PutSubscriptionFilter** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `PutSubscriptionFilter`.

C++

SDK untuk C++

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Sertakan file-file yang diperlukan.

```
#include <aws/core/Aws.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/PutSubscriptionFilterRequest.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Buat filter langganan.

```
Aws::CloudWatchLogs::CloudWatchLogsClient cwl;
Aws::CloudWatchLogs::Model::PutSubscriptionFilterRequest request;
request.SetFilterName(filter_name);
request.SetFilterPattern(filter_pattern);
request.SetLogGroupName(log_group);
request.SetDestinationArn(dest_arn);
auto outcome = cwl.PutSubscriptionFilter(request);
if (!outcome.IsSuccess())
{
    std::cout << "Failed to create CloudWatch logs subscription filter "
              << filter_name << ": " << outcome.GetError().GetMessage() <<
              std::endl;
}
else
{
    std::cout << "Successfully created CloudWatch logs subscription " <<
              "filter " << filter_name << std::endl;
}
```

- Untuk detail API, lihat [PutSubscriptionFilter](#) di Referensi AWS SDK untuk C++ API.

Java

SDK untuk Java 2.x

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsClient;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.CloudWatchLogsException;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.PutSubscriptionFilterRequest;

/**
 * Before running this code example, you need to grant permission to CloudWatch
 * Logs the right to execute your Lambda function.
 * To perform this task, you can use this CLI command:
 *
 * aws lambda add-permission --function-name "lamda1" --statement-id "lamda1"
 * --principal "logs.us-west-2.amazonaws.com" --action "lambda:InvokeFunction"
 * --source-arn "arn:aws:logs:us-west-2:111111111111:log-group:testgroup:*"
 * --source-account "111111111111"
 *
 * Make sure you replace the function name with your function name and replace
 * '111111111111' with your account details.
 * For more information, see "Subscription Filters with AWS Lambda" in the
 * Amazon CloudWatch Logs Guide.
 *
 * Also, 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 PutSubscriptionFilter {
    public static void main(String[] args) {
        final String usage = ""

            Usage:
                <filter> <pattern> <logGroup> <functionArn>\s

            Where:
                filter - A filter name (for example, myfilter).
                pattern - A filter pattern (for example, ERROR).
                logGroup - A log group name (testgroup).
                functionArn - An AWS Lambda function ARN (for example,
arn:aws:lambda:us-west-2:111111111111:function:lambda1) .
                "";

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }

        String filter = args[0];
        String pattern = args[1];
        String logGroup = args[2];
        String functionArn = args[3];
        Region region = Region.US_WEST_2;
        CloudWatchLogsClient cwl = CloudWatchLogsClient.builder()
            .region(region)
            .build();

        putSubFilters(cwl, filter, pattern, logGroup, functionArn);
        cwl.close();
    }

    public static void putSubFilters(CloudWatchLogsClient cwl,
        String filter,
        String pattern,
        String logGroup,
        String functionArn) {

        try {
            PutSubscriptionFilterRequest request =
PutSubscriptionFilterRequest.builder()
                .filterName(filter)

```

```

        .filterPattern(pattern)
        .logGroupName(logGroup)
        .destinationArn(functionArn)
        .build();

    cwL.putSubscriptionFilter(request);
    System.out.printf(
        "%s",
        "Successfully created CloudWatch logs subscription filter
        filter");

    } catch (CloudWatchLogsException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

```

- Untuk detail API, lihat [PutSubscriptionFilter](#) di 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 mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

import { PutSubscriptionFilterCommand } from "@aws-sdk/client-cloudwatch-logs";
import { client } from "../libs/client.js";

const run = async () => {
    const command = new PutSubscriptionFilterCommand({
        // An ARN of a same-account Kinesis stream, Kinesis Firehose
        // delivery stream, or Lambda function.
        // https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/
        SubscriptionFilters.html
        destinationArn: process.env.CLOUDWATCH_LOGS_DESTINATION_ARN,
    });

```

```
// A name for the filter.
filterName: process.env.CLOUDWATCH_LOGS_FILTER_NAME,

// A filter pattern for subscribing to a filtered stream of log events.
// https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/
FilterAndPatternSyntax.html
filterPattern: process.env.CLOUDWATCH_LOGS_FILTER_PATTERN,

// The name of the log group. Messages in this group matching the filter
pattern
// will be sent to the destination ARN.
logGroupName: process.env.CLOUDWATCH_LOGS_LOG_GROUP,
});

try {
  return await client.send(command);
} catch (err) {
  console.error(err);
}
};

export default run();
```

- Untuk detail API, lihat [PutSubscriptionFilter](#) di Referensi AWS SDK untuk 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 the CloudWatchLogs service object
var cwl = new AWS.CloudWatchLogs({ apiVersion: "2014-03-28" });

var params = {
```

```
destinationArn: "LAMBDA_FUNCTION_ARN",
filterName: "FILTER_NAME",
filterPattern: "ERROR",
logGroupName: "LOG_GROUP",
};

cwl.putSubscriptionFilter(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK untuk JavaScript](#).
- Untuk detail API, lihat [PutSubscriptionFilter](#) di Referensi AWS SDK untuk JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **StartLiveTail** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `StartLiveTail`.

.NET

SDK untuk .NET

Sertakan file-file yang diperlukan.

```
using Amazon;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
```

Mulai sesi Live Tail.

```
var client = new AmazonCloudWatchLogsClient();
```

```

var request = new StartLiveTailRequest
{
    LogGroupIdentifiers = logGroupIdentifiers,
    LogStreamNames = logStreamNames,
    LogEventFilterPattern = filterPattern,
};

var response = await client.StartLiveTailAsync(request);

// Catch if request fails
if (response.HttpStatusCode != System.Net.HttpStatusCode.OK)
{
    Console.WriteLine("Failed to start live tail session");
    return;
}

```

Anda dapat menangani acara dari sesi Live Tail dengan dua cara:

```

/* Method 1
 * 1). Asynchronously loop through the event stream
 * 2). Set a timer to dispose the stream and stop the Live Tail
session at the end.
*/
var eventStream = response.ResponseStream;
var task = Task.Run(() =>
{
    foreach (var item in eventStream)
    {
        if (item is LiveTailSessionUpdate liveTailSessionUpdate)
        {
            foreach (var sessionResult in
liveTailSessionUpdate.SessionResults)
            {
                Console.WriteLine("Message : {0}",
sessionResult.Message);
            }
        }
        if (item is LiveTailSessionStart)
        {
            Console.WriteLine("Live Tail session started");
        }
    }
    // On-stream exceptions are processed here

```

```

        if (item is CloudWatchLogsEventStreamException)
        {
            Console.WriteLine($"ERROR: {item}");
        }
    }
});
// Close the stream to stop the session after a timeout
if (!task.Wait(TimeSpan.FromSeconds(10))){
    eventStream.Dispose();
    Console.WriteLine("End of line");
}

```

```

/* Method 2
 * 1). Add event handlers to each event variable
 * 2). Start processing the stream and wait for a timeout using
AutoResetEvent
*/
AutoResetEvent endEvent = new AutoResetEvent(false);
var eventStream = response.ResponseStream;
using (eventStream) // automatically disposes the stream to stop the
session after execution finishes
{
    eventStream.SessionStartReceived += (sender, e) =>
    {
        Console.WriteLine("LiveTail session started");
    };
    eventStream.SessionUpdateReceived += (sender, e) =>
    {
        foreach (LiveTailSessionLogEvent logEvent in
e.EventStreamEvent.SessionResults){
            Console.WriteLine("Message: {0}", logEvent.Message);
        }
    };
    // On-stream exceptions are captured here
    eventStream.ExceptionReceived += (sender, e) =>
    {
        Console.WriteLine($"ERROR:
{e.EventStreamException.Message}");
    };

    eventStream.StartProcessing();
    // Stream events for this amount of time.

```

```
        endEvent.WaitOne(TimeSpan.FromSeconds(10));
        Console.WriteLine("End of line");
    }
}
```

- Untuk detail API, lihat [StartLiveTail](#) di Referensi AWS SDK untuk .NET API.

Go

SDK untuk Go V2

Sertakan file-file yang diperlukan.

```
import (
    "context"
    "log"
    "time"

    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
    "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs/types"
)
```

Tangani acara dari sesi Live Tail.

```
func handleEventStreamAsync(stream *cloudwatchlogs.StartLiveTailEventStream) {
    eventsChan := stream.Events()
    for {
        event := <-eventsChan
        switch e := event.(type) {
        case *types.StartLiveTailResponseStreamMemberSessionStart:
            log.Println("Received SessionStart event")
        case *types.StartLiveTailResponseStreamMemberSessionUpdate:
            for _, logEvent := range e.Value.SessionResults {
                log.Println(*logEvent.Message)
            }
        default:
            // Handle on-stream exceptions
            if err := stream.Err(); err != nil {
                log.Fatalf("Error occurred during streaming: %v", err)
            } else if event == nil {
                return
            }
        }
    }
}
```



```

    log.Println("Stream is Closed")
    return
} else {
    log.Fatalf("Unknown event type: %T", e)
}
}
}
}

```

Mulai sesi Live Tail.

```

cfg, err := config.LoadDefaultConfig(context.TODO())
if err != nil {
    panic("configuration error, " + err.Error())
}
client := cloudwatchlogs.NewFromConfig(cfg)

request := &cloudwatchlogs.StartLiveTailInput{
    LogGroupIdentifiers: logGroupIdentifiers,
    LogStreamNames:      logStreamNames,
    LogEventFilterPattern: logEventFilterPattern,
}

response, err := client.StartLiveTail(context.TODO(), request)
// Handle pre-stream Exceptions
if err != nil {
    log.Fatalf("Failed to start streaming: %v", err)
}

// Start a Goroutine to handle events over stream
stream := response.GetStream()
go handleEventStreamAsync(stream)

```

Hentikan sesi Live Tail setelah periode waktu berlalu.

```

// Close the stream (which ends the session) after a timeout
time.Sleep(10 * time.Second)
stream.Close()
log.Println("Event stream closed")

```

- Untuk detail API, lihat [StartLiveTail](#) di Referensi AWS SDK untuk Go API.

Java

SDK untuk Java 2.x

Sertakan file-file yang diperlukan.

```
import io.reactivex.FlowableSubscriber;
import io.reactivex.annotations.NonNull;
import org.reactivestreams.Subscription;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.services.cloudwatchlogs.CloudWatchLogsAsyncClient;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionLogEvent;
import software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionStart;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.LiveTailSessionUpdate;
import software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailRequest;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailResponseHandler;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.CloudWatchLogsException;
import
    software.amazon.awssdk.services.cloudwatchlogs.model.StartLiveTailResponseStream;

import java.util.Date;
import java.util.List;
import java.util.concurrent.atomic.AtomicReference;
```

Tangani acara dari sesi Live Tail.

```
private static StartLiveTailResponseHandler
getStartLiveTailResponseStreamHandler(
    AtomicReference<Subscription> subscriptionAtomicReference) {
    return StartLiveTailResponseHandler.builder()
        .onResponse(r -> System.out.println("Received initial response"))
        .onError(throwable -> {
            CloudWatchLogsException e = (CloudWatchLogsException)
throwable.getCause();
            System.err.println(e.awsErrorDetails().errorMessage());
```

```

        System.exit(1);
    })
    .subscriber(() -> new FlowableSubscriber<>() {
        @Override
        public void onSubscribe(@NonNull Subscription s) {
            subscriptionAtomicReference.set(s);
            s.request(Long.MAX_VALUE);
        }

        @Override
        public void onNext(StartLiveTailResponseStream event) {
            if (event instanceof LiveTailSessionStart) {
                LiveTailSessionStart sessionStart =
(LiveTailSessionStart) event;
                System.out.println(sessionStart);
            } else if (event instanceof LiveTailSessionUpdate) {
                LiveTailSessionUpdate sessionUpdate =
(LiveTailSessionUpdate) event;
                List<LiveTailSessionLogEvent> logEvents =
sessionUpdate.sessionResults();
                logEvents.forEach(e -> {
                    long timestamp = e.timestamp();
                    Date date = new Date(timestamp);
                    System.out.println "[" + date + "] " + e.message());
                });
            } else {
                throw CloudWatchLogsException.builder().message("Unknown
event type").build();
            }
        }

        @Override
        public void onError(Throwable throwable) {
            System.out.println(throwable.getMessage());
            System.exit(1);
        }

        @Override
        public void onComplete() {
            System.out.println("Completed Streaming Session");
        }
    })
    .build();
}

```

Mulai sesi Live Tail.

```
CloudWatchLogsAsyncClient cloudWatchLogsAsyncClient =
    CloudWatchLogsAsyncClient.builder()
        .credentialsProvider(ProfileCredentialsProvider.create())
        .build();

StartLiveTailRequest request =
    StartLiveTailRequest.builder()
        .logGroupIdentifiers(logGroupIdentifiers)
        .logStreamNames(logStreamNames)
        .logEventFilterPattern(logEventFilterPattern)
        .build();

/* Create a reference to store the subscription */
final AtomicReference<Subscription> subscriptionAtomicReference = new
AtomicReference<>(null);

cloudWatchLogsAsyncClient.startLiveTail(request,
getStartLiveTailResponseStreamHandler(subscriptionAtomicReference));
```

Hentikan sesi Live Tail setelah periode waktu berlalu.

```
/* Set a timeout for the session and cancel the subscription. This will:
 * 1). Close the stream
 * 2). Stop the Live Tail session
 */
try {
    Thread.sleep(10000);
} catch (InterruptedException e) {
    throw new RuntimeException(e);
}
if (subscriptionAtomicReference.get() != null) {
    subscriptionAtomicReference.get().cancel();
    System.out.println("Subscription to stream closed");
}
```

- Untuk detail API, lihat [StartLiveTail](#) di Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

Sertakan file-file yang diperlukan.

```
import { CloudWatchLogsClient, StartLiveTailCommand } from "@aws-sdk/client-cloudwatch-logs";
```

Tangani acara dari sesi Live Tail.

```
async function handleResponseAsync(response) {
  try {
    for await (const event of response.responseStream) {
      if (event.sessionStart !== undefined) {
        console.log(event.sessionStart);
      } else if (event.sessionUpdate !== undefined) {
        for (const logEvent of event.sessionUpdate.sessionResults) {
          const timestamp = logEvent.timestamp;
          const date = new Date(timestamp);
          console.log "[" + date + "]" + logEvent.message);
        }
      } else {
        console.error("Unknown event type");
      }
    }
  } catch (err) {
    // On-stream exceptions are captured here
    console.error(err)
  }
}
```

Mulai sesi Live Tail.

```
const client = new CloudWatchLogsClient();

const command = new StartLiveTailCommand({
  logGroupIdentifiers: logGroupIdentifiers,
  logStreamNames: logStreamNames,
  logEventFilterPattern: filterPattern
});
```

```
try{
    const response = await client.send(command);
    handleResponseAsync(response);
} catch (err){
    // Pre-stream exceptions are captured here
    console.log(err);
}
```

Hentikan sesi Live Tail setelah periode waktu berlalu.

```
/* Set a timeout to close the client. This will stop the Live Tail session.
*/
setTimeout(function() {
    console.log("Client timeout");
    client.destroy();
}, 10000);
```

- Untuk detail API, lihat [StartLiveTail](#) di Referensi AWS SDK untuk JavaScript API.

Kotlin

SDK untuk Kotlin

Sertakan file-file yang diperlukan.

```
import aws.sdk.kotlin.services.cloudwatchlogs.CloudWatchLogsClient
import aws.sdk.kotlin.services.cloudwatchlogs.model.StartLiveTailRequest
import aws.sdk.kotlin.services.cloudwatchlogs.model.StartLiveTailResponseStream
import kotlinx.coroutines.flow.takeWhile
```

Mulai sesi Live Tail.

```
val client = CloudWatchLogsClient.fromEnvironment()

val request = StartLiveTailRequest {
    logGroupIdentifiers = logGroupIdentifiersVal
    logStreamNames = logStreamNamesVal
    logEventFilterPattern = logEventFilterPatternVal
}
```

```

val startTime = System.currentTimeMillis()

try {
    client.startLiveTail(request) { response ->
        val stream = response.responseStream
        if (stream != null) {
            /* Set a timeout to unsubscribe from the flow. This will:
             * 1). Close the stream
             * 2). Stop the Live Tail session
             */
            stream.takeWhile { System.currentTimeMillis() - startTime <
10000 }.collect { value ->
                if (value is StartLiveTailResponseStream.SessionStart) {
                    println(value.asSessionStart())
                } else if (value is
StartLiveTailResponseStream.SessionUpdate) {
                    for (e in value.asSessionUpdate().sessionResults!!) {
                        println(e)
                    }
                } else {
                    throw IllegalArgumentException("Unknown event type")
                }
            }
        } else {
            throw IllegalArgumentException("No response stream")
        }
    }
} catch (e: Exception) {
    println("Exception occurred during StartLiveTail: $e")
    System.exit(1)
}

```

- Untuk detail API, lihat [StartLiveTail](#) di AWS SDK untuk referensi API Kotlin.

Python

SDK untuk Python (Boto3)

Sertakan file-file yang diperlukan.

```
import boto3
```

```
import time
from datetime import datetime
```

Mulai sesi Live Tail.

```
# Initialize the client
client = boto3.client('logs')

start_time = time.time()

try:
    response = client.start_live_tail(
        logGroupIdentifiers=log_group_identifiers,
        logStreamNames=log_streams,
        logEventFilterPattern=filter_pattern
    )
    event_stream = response['responseStream']
    # Handle the events streamed back in the response
    for event in event_stream:
        # Set a timeout to close the stream.
        # This will end the Live Tail session.
        if (time.time() - start_time >= 10):
            event_stream.close()
            break
        # Handle when session is started
        if 'sessionStart' in event:
            session_start_event = event['sessionStart']
            print(session_start_event)
        # Handle when log event is given in a session update
        elif 'sessionUpdate' in event:
            log_events = event['sessionUpdate']['sessionResults']
            for log_event in log_events:
                print('[{date}]'
                    .format(date=datetime.fromtimestamp(log_event['timestamp']/1000), log=log_event['message']))
            else:
                # On-stream exceptions are captured here
                raise RuntimeError(str(event))
except Exception as e:
    print(e)
```

- Untuk detail API, lihat [StartLiveTail](#) di AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **StartQuery** dengan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan `StartQuery`.

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:

- [Jalankan kueri besar](#)

.NET

SDK untuk .NET (v4)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/// <summary>
/// Starts a CloudWatch Logs Insights query.
/// </summary>
/// <param name="logGroupName">The name of the log group to query.</param>
/// <param name="queryString">The CloudWatch Logs Insights query string.</
param>
/// <param name="startTime">The start time for the query (seconds since
epoch).</param>
/// <param name="endTime">The end time for the query (seconds since epoch).</
param>
/// <param name="limit">The maximum number of results to return.</param>
/// <returns>The query ID if successful, null otherwise.</returns>
public async Task<string?> StartQueryAsync(
    string logGroupName,
    string queryString,
    long startTime,
    long endTime,
    int limit = 10000)
```

```
{
    try
    {
        var request = new StartQueryRequest
        {
            LogGroupName = logGroupName,
            QueryString = queryString,
            StartTime = startTime,
            EndTime = endTime,
            Limit = limit
        };

        var response = await _amazonCloudWatchLogs.StartQueryAsync(request);
        return response.QueryId;
    }
    catch (InvalidParameterException ex)
    {
        _logger.LogError($"Invalid parameter for query: {ex.Message}");
        return null;
    }
    catch (ResourceNotFoundException ex)
    {
        _logger.LogError($"Log group not found: {ex.Message}");
        return null;
    }
    catch (Exception ex)
    {
        _logger.LogError($"An error occurred while starting query: {ex.Message}");
        return null;
    }
}
```

- Untuk detail API, lihat [StartQuery](#) di Referensi AWS SDK untuk .NET API.

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
/**
 * Wrapper for the StartQueryCommand. Uses a static query string
 * for consistency.
 * @param {[Date, Date]} dateRange
 * @param {number} maxLogs
 * @returns {Promise<{ queryId: string }>}
 */
async _startQuery([startDate, endDate], maxLogs = 10000) {
  try {
    return await this.client.send(
      new StartQueryCommand({
        logGroupNames: this.logGroupNames,
        queryString: "fields @timestamp, @message | sort @timestamp asc",
        startTime: startDate.valueOf(),
        endTime: endDate.valueOf(),
        limit: maxLogs,
      }),
    );
  } catch (err) {
    /** @type {string} */
    const message = err.message;
    if (message.startsWith("Query's end date and time")) {
      // This error indicates that the query's start or end date occur
      // before the log group was created.
      throw new DateOutOfBoundsError(message);
    }

    throw err;
  }
}
```

- Untuk detail API, lihat [StartQuery](#) di Referensi AWS SDK untuk JavaScript API.

Python

SDK untuk Python (Boto3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
def perform_query(self, date_range):
    """
    Performs the actual CloudWatch log query.

    :param date_range: A tuple representing the start and end datetime for
the query.
    :type date_range: tuple
    :return: A list containing the query results.
    :rtype: list
    """
    client = boto3.client("logs")
    try:
        try:
            start_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
            )
            end_time = round(

self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
            )
            response = client.start_query(
                logGroupName=self.log_group,
                startTime=start_time,
                endTime=end_time,
                queryString=self.query_string,
                limit=self.limit,
            )
            query_id = response["queryId"]
        except client.exceptions.ResourceNotFoundException as e:
```

```

        raise DateOutOfBoundsError(f"Resource not found: {e}")
    while True:
        time.sleep(1)
        results = client.get_query_results(queryId=query_id)
        if results["status"] in [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ]:
            return results.get("results", [])
    except DateOutOfBoundsError:
        return []

def _initiate_query(self, client, date_range, max_logs):
    """
    Initiates the CloudWatch logs query.

    :param date_range: A tuple representing the start and end datetime for
    the query.
    :type date_range: tuple
    :param max_logs: The maximum number of logs to retrieve.
    :type max_logs: int
    :return: The query ID as a string.
    :rtype: str
    """
    try:
        start_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
        )
        end_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
        )
        response = client.start_query(
            logGroupName=self.log_group,
            startTime=start_time,
            endTime=end_time,
            queryString=self.query_string,
            limit=max_logs,
        )
        return response["queryId"]

```

```
except client.exceptions.ResourceNotFoundException as e:
    raise DateOutOfBoundsError(f"Resource not found: {e}")
```

- Untuk detail API, lihat [StartQuery](#) di AWS SDK for Python (Boto3) Referensi API.

SAP ABAP

SDK for SAP ABAP

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
TRY.
    " iv_log_group_name = '/aws/lambda/my-function'
    " iv_query_string = 'fields @timestamp, @message | sort @timestamp desc |
limit 20'
    " iv_start_time and iv_end_time must be in Unix epoch milliseconds (ms
since Jan 1, 1970 00:00:00 UTC)
    oo_result = lo_cwl->startquery(
        iv_loggroupname = iv_log_group_name
        iv_starttime     = iv_start_time
        iv_endtime       = iv_end_time
        iv_querystring   = iv_query_string
        iv_limit         = iv_limit ).

    " Display the query ID for tracking
    DATA(lv_query_id) = oo_result->get_queryid( ).
    MESSAGE |Query started successfully with ID: { lv_query_id }| TYPE 'I'.
CATCH /aws1/cx_cwlinvalidparameterex.
    MESSAGE 'Invalid parameter.' TYPE 'E'.
CATCH /aws1/cx_cwllimitexceededex.
    MESSAGE 'Limit exceeded.' TYPE 'E'.
CATCH /aws1/cx_cwlmalformedqueryex.
    MESSAGE 'Malformed query.' TYPE 'E'.
CATCH /aws1/cx_cwlresourcenotfoundex.
    MESSAGE 'Resource not found.' TYPE 'E'.
CATCH /aws1/cx_cwlserviceunavailex.
```

```
MESSAGE 'Service unavailable.' TYPE 'E'.  
ENDTRY.
```

- Untuk detail API, lihat [StartQuery](#) di AWS SDK untuk referensi SAP ABAP API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Skenario untuk CloudWatch Log menggunakan AWS SDKs

Contoh kode berikut menunjukkan kepada Anda bagaimana menerapkan skenario umum di CloudWatch Log dengan AWS SDKs. Skenario ini menunjukkan kepada Anda cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam CloudWatch Log atau digabungkan dengan yang lain Layanan AWS. Setiap skenario menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode.

Skenario menargetkan tingkat pengalaman menengah untuk membantu Anda memahami tindakan layanan dalam konteks.

Contoh

- [Konfigurasi Amazon ECS Service Connect](#)
- [Membuat fungsi Lambda pertama Anda](#)
- [Gunakan CloudWatch Log untuk menjalankan kueri besar](#)
- [Menggunakan peristiwa terjadwal untuk menginvokasi fungsi Lambda](#)

Konfigurasi Amazon ECS Service Connect

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat infrastruktur VPC
- Mengatur pencatatan
- Buat cluster ECS
- Konfigurasi peran IAM
- Buat layanan dengan Service Connect

- Verifikasi penyebaran
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori [tutorial pengembang Sample](#).

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
# service
# Uses the default VPC to avoid VPC limits

set -e # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()
```



```
# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
    "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}
```

```

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
"$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
"$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then

```

```

        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"

    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
"Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=( $SUBNETS )

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \

```

```

        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
    --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
    "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
    --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
    "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \

```

```

    --tags key=Environment,value=tutorial key=project,value=doc-smith
key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \

```

```

        --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
>/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi

```

```

}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."

    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam:${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam:${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

```

```

TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":'
"^[^"]*" | cut -d'"' -f4)
track_resource "TASK_DEF:service-connect-nginx"
log "Created task definition: $TASK_DEF_ARN"

# Clean up temporary file
rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ]
    },
},

```



```

        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        },
        "tags": [
            {
                "key": "Environment",
                "value": "tutorial"
            },
            {
                "key": "project",
                "value": "doc-smith"
            },
            {
                "key": "tutorial",
                "value": "amazon-ecs-service-connect"
            }
        ]
    }
}
EOF

```

```

SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
check_for_errors "$SERVICE_OUTPUT" "create-service"

track_resource "SERVICE:service-connect-nginx-service"
log "Created ECS service: service-connect-nginx-service"

wait_for_resource "service" "service-connect-nginx-service"

# Clean up temporary file
rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status

```

```

SERVICE_STATUS=$(aws ecs describe-services \
    --cluster "$CLUSTER_NAME" \
    --services "service-connect-nginx-service" \
    --query 'services[0].status' --output text)
log "Service status: $SERVICE_STATUS"

# Check running tasks
RUNNING_COUNT=$(aws ecs describe-services \
    --cluster "$CLUSTER_NAME" \
    --services "service-connect-nginx-service" \
    --query 'services[0].runningCount' --output text)
log "Running tasks: $RUNNING_COUNT"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks \
    --cluster "$CLUSTER_NAME" \
    --service-name "service-connect-nginx-service" \
    --query 'taskArns[0]' --output text)

if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
    log "Task ARN: $TASK_ARN"

    # Try to get task IP address
    TASK_IP=$(aws ecs describe-tasks \
        --cluster "$CLUSTER_NAME" \
        --tasks "$TASK_ARN" \
        --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`.value' \
        --output text 2>/dev/null || echo "")

    if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
        log "Task IP address: $TASK_IP"
    else
        log "Could not retrieve task IP address"
    fi
fi

# Check Service Connect namespace
NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
    --filters Name=TYPE,Values=HTTP \
    --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then

```

```

        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi

    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "=====
    echo "CREATED RESOURCES"
    echo "=====
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "=====
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
"$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
to scale down service $resource_id"

```

```

        aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
"$resource_id" 2>/dev/null || true
        aws ecs delete-service --cluster "$CLUSTER_NAME" --service
"$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
service $resource_id"
        ;;
    "TASK_DEF")
        TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
"$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
        for arn in $TASK_DEF_ARNS; do
            aws ecs deregister-task-definition --task-definition "$arn"
>/dev/null 2>&1 || true
        done
        ;;
    "ROLE")
        aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
        aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
"error" && log "Warning: Failed to delete role $resource_id"
        ;;
    "IAM_ROLE")
        aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
        aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
"error" && log "Warning: Failed to delete role $resource_id"
        ;;
    "CLUSTER")
        aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
"error" && log "Warning: Failed to delete cluster $resource_id"
        ;;
    "SG")
        for attempt in 1 2 3 4 5; do
            if aws ec2 delete-security-group --group-id "$resource_id"
2>/dev/null; then
                break
            fi
            log "Security group $resource_id still has dependencies,
retrying in 30s ($attempt/5)..."
            sleep 30
        done
        ;;
    "LOG_GROUP")

```

```

        aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
        grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
        ;;
    "NAMESPACE")
        # First, delete any services in the namespace
        NAMESPACE_SERVICES=$(aws servicediscovery list-services \
            --filters Name=NAMESPACE_ID,Values="$resource_id" \
            --query 'Services[].Id' --output text 2>/dev/null || echo "")

        if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
"None" ]]; then
            for service_id in $NAMESPACE_SERVICES; do
                aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                sleep 2
            done
            fi

            # Then delete the namespace
            aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
            ;;
        esac

        sleep 2 # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

```

```
# Setup infrastructure using default VPC
setup_default_vpc_infrastructure

# Create CloudWatch log groups
create_log_groups

# Create ECS cluster
create_ecs_cluster

# Create IAM roles
create_iam_roles

# Create task definition
create_task_definition

# Create ECS service
create_ecs_service

# Verify deployment
verify_deployment

log "Tutorial completed successfully!"

# Display created resources
display_resources

# Ask user if they want to clean up
echo ""
echo "=====
echo "CLEANUP CONFIRMATION"
echo "=====
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    cleanup_resources
    log "All resources have been cleaned up"
else
    log "Resources left intact. You can clean them up later by running the
cleanup function."
    echo ""
    echo "To clean up resources later, you can use the AWS CLI commands or
the AWS Management Console."
```

```
    echo "Remember to delete resources in the correct order to avoid
dependency issues."
fi
}

# Make script executable and run
chmod +x "$0"
main "$@"
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .

- [AttachRolePolicy](#)
- [AuthorizeSecurityGroupIngress](#)
- [CreateCluster](#)
- [CreateLogGroup](#)
- [CreateRole](#)
- [CreateSecurityGroup](#)
- [CreateService](#)
- [DeleteCluster](#)
- [DeleteLogGroup](#)
- [DeleteNamespace](#)
- [DeleteRole](#)
- [DeleteSecurityGroup](#)
- [DeleteService](#)
- [DeregisterTaskDefinition](#)
- [DescribeClusters](#)
- [DescribeServices](#)
- [DescribeSubnets](#)
- [DescribeTasks](#)
- [DescribeVpcs](#)
- [DetachRolePolicy](#)
- [GetCallerIdentity](#)

- [ListNamespaces](#)
- [ListServices](#)
- [ListTaskDefinitions](#)
- [ListTasks](#)
- [RegisterTaskDefinition](#)
- [UpdateService](#)
- [Tunggu](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat fungsi Lambda pertama Anda

Contoh kode berikut ini menunjukkan cara untuk melakukan:

- Buat peran IAM untuk Lambda
- Buat kode fungsi
- Buat fungsi Lambda
- Uji fungsi Lambda Anda
- Pembersihan sumber daya

Bash

AWS CLI dengan skrip Bash

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankan di repositori [tutorial pengembang Sample](#).

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
# views CloudWatch logs, and cleans up all resources.
```



```

#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE -o pipefail

#####
# Setup
#####

UNIQUE_ID=$(head -c 8 /dev/urandom | od -An -tx1 | tr -d ' ')
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
readonly TEMP_DIR
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1

declare -a CREATED_RESOURCES

#####
# Helper functions
#####

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
        local NAME="${RESOURCE#*:}"

```

```

    case "$TYPE" in
        log-group)
            echo "Deleting CloudWatch log group: ${NAME}"
            aws logs delete-log-group \
                --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete log group ${NAME}."
            ;;
        lambda-function)
            echo "Deleting Lambda function: ${NAME}"
            aws lambda delete-function \
                --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
delete Lambda function ${NAME}."
            echo "  Waiting for function deletion to complete..."
            local DELETE_WAIT=0
            while aws lambda get-function --function-name "$NAME" > /dev/null
2>&1; do
                sleep 2
                DELETE_WAIT=$((DELETE_WAIT + 2))
                if [ "$DELETE_WAIT" -ge 60 ]; then
                    echo "  WARNING: Timed out waiting for function
deletion."
                    break
                fi
            done
            ;;
        iam-role-policy)
            local ROLE_PART="${NAME%|*}"
            local POLICY_PART="${NAME#*|}"
            echo "Detaching policy from role: ${ROLE_PART}"
            aws iam detach-role-policy \
                --role-name "$ROLE_PART" \
                --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
not detach policy from role ${ROLE_PART}."
            ;;
        iam-role)
            echo "Deleting IAM role: ${NAME}"
            aws iam delete-role \
                --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
IAM role ${NAME}."
            ;;
    esac
done

```

```

    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "======"
    echo "ERROR: Script failed at $1"
    echo "======"
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
    local COMMAND="$2"
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
seconds."
            return 1
        fi
    done
}

```

```

        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

validate_input() {
    local input="$1"
    local pattern="$2"
    if ! [[ "$input" =~ $pattern ]]; then
        echo "ERROR: Invalid input: $input"
        return 1
    fi
    return 0
}

#####
# Region pre-check
#####

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
"$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi

#####
# Runtime selection
#####

echo ""
echo "=====
echo "AWS Lambda - Create Your First Function"
echo "=====
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Using default: Python 3.13"
RUNTIME_CHOICE="1"

case "$RUNTIME_CHOICE" in

```

```

1)
    RUNTIME="python3.13"
    HANDLER="lambda_function.lambda_handler"
    CODE_FILE="lambda_function.py"
    cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'

import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    if not isinstance(event, dict) or 'length' not in event or 'width' not in event:
        raise ValueError('Event must contain length and width')
    try:
        length = float(event['length'])
        width = float(event['width'])
        if length < 0 or width < 0:
            raise ValueError('Length and width must be non-negative')
        area = calculate_area(length, width)
        print(f'The area is {area}')
        logger.info(f'CloudWatch logs group: {context.log_group_name}')
        return json.dumps({'area': area})
    except (TypeError, ValueError) as e:
        logger.error(f'Error processing input: {str(e)}')
        raise
def calculate_area(length, width):
    return length * width
PYTHON_EOF
    echo "Selected runtime: Python 3.13"
    ;;

2)
    RUNTIME="nodejs22.x"
    HANDLER="index.handler"
    CODE_FILE="index.mjs"
    cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'

export const handler = async (event, context) => {
    if (!event || typeof event.length !== 'number' || typeof event.width !== 'number') {
        throw new Error('Event must contain numeric length and width');
    }
    if (event.length < 0 || event.width < 0) {
        throw new Error('Length and width must be non-negative');
    }
    const area = event.length * event.width;

```

```

    console.log(`The area is ${area}`);
    console.log('CloudWatch log group: ', context.logGroupName);
    return JSON.stringify({area});
};
NODEJS_EOF
    echo "Selected runtime: Node.js 22.x"
    ;;
*)
    echo "ERROR: Invalid choice. Please enter 1 or 2."
    exit 1
    ;;
esac

#####
# Step 1: Create IAM execution role
#####

echo ""
echo "=====
echo "Step 1: Create IAM execution role"
echo "=====
echo ""

TRUST_POLICY='{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)

if ! validate_input "$ROLE_OUTPUT" "^arn:aws:iam::[0-9]+:role/"; then

```

```

    echo "ERROR: Failed to create IAM role"
    exit 1
fi

echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

aws iam tag-role \
    --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=lambda-gettingstarted

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

#####
# Step 2: Create Lambda function
#####

echo ""
echo "=====
echo "Step 2: Create Lambda function"
echo "=====
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR" || exit 1
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1 || {
    echo "ERROR: Failed to create deployment package"
    exit 1
}

```

```

cd "$ORIGINAL_DIR" || exit 1

if [ ! -f "${TEMP_DIR}/function.zip" ]; then
    echo "ERROR: Deployment package creation failed"
    exit 1
fi

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)

if [ -z "$CREATE_OUTPUT" ]; then
    echo "ERROR: Failed to create Lambda function"
    exit 1
fi

echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
    query State --output text" \
    "Active"

#####
# Step 3: Invoke the function
#####

echo ""
echo "=====
echo "Step 3: Invoke the function"
echo "=====

```



```

echo ""

TEST_EVENT='{ "length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

if ! validate_input "$TEST_EVENT" '"length": [0-9]+, "width": [0-9]+'; then
    echo "ERROR: Invalid test event format"
    exit 1
fi

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

if [ ! -f "${TEMP_DIR}/response.json" ]; then
    echo "ERROR: No response file generated"
    exit 1
fi

RESPONSE=$(cat "${TEMP_DIR}/response.json")
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

#####
# Step 4: View CloudWatch logs
#####

echo ""
echo "=====
echo "Step 4: View CloudWatch Logs"
echo "=====
echo ""

```

```

echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

aws logs tag-log-group \
    --log-group-name "$LOG_GROUP_NAME" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted

#####

```

```
# Summary and cleanup
#####

echo ""
echo "=====
echo "SUMMARY"
echo "=====
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:    ${FUNCTION_NAME}"
echo "  CloudWatch logs:    ${LOG_GROUP_NAME}"
echo ""
echo "=====
echo "CLEANUP"
echo "=====
echo ""
echo "Cleaning up all created resources..."
cleanup_resources

echo ""
echo "Done."
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .
 - [AttachRolePolicy](#)
 - [CreateFunction](#)
 - [CreateRole](#)
 - [DeleteFunction](#)
 - [DeleteLogGroup](#)
 - [DeleteRole](#)
 - [DescribeLogStreams](#)
 - [DetachRolePolicy](#)
 - [GetFunction](#)
 - [GetFunctionConfiguration](#)
 - [GetLogEvents](#)
 - [Memohon](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan CloudWatch Log untuk menjalankan kueri besar

Contoh kode berikut menunjukkan cara menggunakan CloudWatch Log untuk menanyakan lebih dari 10.000 catatan.

.NET

SDK untuk .NET (v4)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Ini adalah alur kerja utama yang menunjukkan skenario kueri besar.

```
using System.Diagnostics;
using System.Text.RegularExpressions;
using Amazon.CloudFormation;
using Amazon.CloudFormation.Model;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
using CloudWatchLogsActions;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;

namespace CloudWatchLogsScenario;

public class LargeQueryWorkflow
{
    /*
        Before running this .NET code example, set up your development environment,
        including your credentials.
        This .NET code example performs the following tasks for the CloudWatch Logs
        Large Query workflow:
    */
}
```

1. Prepare the Application:
 - Prompt the user to deploy CloudFormation stack and generate sample logs.
 - Deploy the CloudFormation template for resource creation.
 - Generate 50,000 sample log entries using CloudWatch Logs API.
 - Wait 5 minutes for logs to be fully ingested.
2. Execute Large Query:
 - Perform recursive queries to retrieve all logs using binary search.
 - Display progress for each query executed.
 - Show total execution time and logs found.
3. Clean up:
 - Prompt the user to delete the CloudFormation stack and all resources.
 - Destroy the CloudFormation stack and wait until removed.

*/

```

public static ILogger<LargeQueryWorkflow> _logger = null!;
public static CloudWatchLogsWrapper _wrapper = null!;
public static IAmazonCloudFormation _amazonCloudFormation = null!;

private static string _logGroupName = "/workflows/cloudwatch-logs/large-
query";
private static string _logStreamName = "stream1";
private static long _queryStartDate;
private static long _queryEndDate;

public static bool _interactive = true;
public static string _stackName = "CloudWatchLargeQueryStack";
private static string _stackResourcePath = "../..../..../..../scenarios/
features/cloudwatch_logs_large_query/resources/stack.yaml";

public static async Task Main(string[] args)
{
    using var host = Host.CreateDefaultBuilder(args)
        .ConfigureLogging(logging =>
            logging.AddFilter("System", LogLevel.Debug)
                .AddFilter("Microsoft", LogLevel.Information))
        .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonCloudWatchLogs>()
                .AddAWSService<IAmazonCloudFormation>()
                .AddTransient<CloudWatchLogsWrapper>()
            )
        .Build();

```

```

        if (_interactive)
        {
            _logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
                .CreateLogger<LargeQueryWorkflow>();

            _wrapper = host.Services.GetRequiredService<CloudWatchLogsWrapper>();
            _amazonCloudFormation =
host.Services.GetRequiredService<IAmazonCloudFormation>();
        }

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Welcome to the CloudWatch Logs Large Query
Scenario.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("This scenario demonstrates how to perform large-scale
queries on");
        Console.WriteLine("CloudWatch Logs using recursive binary search to
retrieve more than");
        Console.WriteLine("the 10,000 result limit.");
        Console.WriteLine();

        try
        {
            Console.WriteLine(new string('-', 80));
            var prepareSuccess = await PrepareApplication();
            Console.WriteLine(new string('-', 80));

            if (prepareSuccess)
            {
                Console.WriteLine(new string('-', 80));
                await ExecuteLargeQuery();
                Console.WriteLine(new string('-', 80));
            }

            Console.WriteLine(new string('-', 80));
            await Cleanup();
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "There was a problem with the scenario,
initiating cleanup...");
            _interactive = false;
            await Cleanup();
        }
    }
}

```

```
    }

    Console.WriteLine("CloudWatch Logs Large Query scenario completed.");
}

/// <summary>
/// Runs the scenario workflow. Used for testing.
/// </summary>
public static async Task RunScenario()
{
    Console.WriteLine(new string('-', 80));
    Console.WriteLine("Welcome to the CloudWatch Logs Large Query
Scenario.");
    Console.WriteLine(new string('-', 80));
    Console.WriteLine("This scenario demonstrates how to perform large-scale
queries on");
    Console.WriteLine("CloudWatch Logs using recursive binary search to
retrieve more than");
    Console.WriteLine("the 10,000 result limit.");
    Console.WriteLine();

    try
    {
        Console.WriteLine(new string('-', 80));
        var prepareSuccess = await PrepareApplication();
        Console.WriteLine(new string('-', 80));

        if (prepareSuccess)
        {
            Console.WriteLine(new string('-', 80));
            await ExecuteLargeQuery();
            Console.WriteLine(new string('-', 80));
        }

        Console.WriteLine(new string('-', 80));
        await Cleanup();
        Console.WriteLine(new string('-', 80));
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, "There was a problem with the scenario,
initiating cleanup...");
        _interactive = false;
        await Cleanup();
    }
}
```

```

    }

    Console.WriteLine("CloudWatch Logs Large Query scenario completed.");
}

/// <summary>
/// Prepares the application by creating the necessary resources.
/// </summary>
/// <returns>True if the application was prepared successfully.</returns>
public static async Task<bool> PrepareApplication()
{
    Console.WriteLine("Preparing the application...");
    Console.WriteLine();

    try
    {
        var deployStack = !_interactive || GetYesNoResponse(
            "Would you like to deploy the CloudFormation stack and generate
sample logs? (y/n) ");

        if (deployStack)
        {
            if (!_interactive)
            {
                Console.Write(
                    $"Enter a path for the CloudFormation stack
resource .yaml file (or press Enter for default '{_stackResourcePath}'): ");
                string? inputPath = Console.ReadLine();
                if (!string.IsNullOrEmpty(inputPath))
                {
                    _stackResourcePath = inputPath;
                }
            }

            _stackName = PromptUserForStackName();

            var deploySuccess = await DeployCloudFormationStack(_stackName);

            if (deploySuccess)
            {
                Console.WriteLine();
                Console.WriteLine("Generating 50,000 sample log entries...");
                var generateSuccess = await GenerateSampleLogs();
            }
        }
    }
}

```



```

        if (generateSuccess)
        {
            Console.WriteLine();
            Console.WriteLine("Sample logs created. Waiting 5 minutes
for logs to be fully ingested...");
            await WaitWithCountdown(300);

            Console.WriteLine("Application preparation complete.");
            return true;
        }
    }
}
else
{
    _logGroupName = PromptUserForInput("Enter the log group name ",
_logGroupName);
    _logStreamName = PromptUserForInput("Enter the log stream name ",
_logStreamName);

    var startDateMs = PromptUserForLong("Enter the query start date
(milliseconds since epoch): ");
    var endDateMs = PromptUserForLong("Enter the query end date
(milliseconds since epoch): ");

    _queryStartDate = startDateMs / 1000;
    _queryEndDate = endDateMs / 1000;

    Console.WriteLine("Application preparation complete.");
    return true;
}
}
catch (Exception ex)
{
    _logger.LogError(ex, "An error occurred while preparing the
application.");
}

Console.WriteLine("Application preparation failed.");
return false;
}

/// <summary>
/// Deploys the CloudFormation stack with the necessary resources.
/// </summary>

```

```
/// <param name="stackName">The name of the CloudFormation stack.</param>
/// <returns>True if the stack was deployed successfully.</returns>
private static async Task<bool> DeployCloudFormationStack(string stackName)
{
    Console.WriteLine($"\\nDeploying CloudFormation stack: {stackName}");

    try
    {
        var request = new CreateStackRequest
        {
            StackName = stackName,
            TemplateBody = await File.ReadAllTextAsync(_stackResourcePath)
        };

        var response = await _amazonCloudFormation.CreateStackAsync(request);

        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"CloudFormation stack creation started:
{stackName}");

            bool stackCreated = await
WaitForStackCompletion(response.StackId);

            if (stackCreated)
            {
                Console.WriteLine("CloudFormation stack created
successfully.");
                return true;
            }
            else
            {
                _logger.LogError($"CloudFormation stack creation failed:
{stackName}");
                return false;
            }
        }
        else
        {
            _logger.LogError($"Failed to create CloudFormation stack:
{stackName}");
            return false;
        }
    }
}
```

```

        catch (AlreadyExistsException)
        {
            _logger.LogWarning($"CloudFormation stack '{stackName}' already
exists. Please provide a unique name.");
            var newStackName = PromptUserForStackName();
            return await DeployCloudFormationStack(newStackName);
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while deploying the
CloudFormation stack: {stackName}");
            return false;
        }
    }
}

/// <summary>
/// Waits for the CloudFormation stack to be in the CREATE_COMPLETE state.
/// </summary>
/// <param name="stackId">The ID of the CloudFormation stack.</param>
/// <returns>True if the stack was created successfully.</returns>
private static async Task<bool> WaitForStackCompletion(string stackId)
{
    int retryCount = 0;
    const int maxRetries = 30;
    const int retryDelay = 10000;

    while (retryCount < maxRetries)
    {
        var describeStacksRequest = new DescribeStacksRequest
        {
            StackName = stackId
        };

        var describeStacksResponse = await
        _amazonCloudFormation.DescribeStacksAsync(describeStacksRequest);

        if (describeStacksResponse.Stacks.Count > 0)
        {
            if (describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.CREATE_COMPLETE)
            {
                return true;
            }
        }
    }
}

```

```

        if (describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.CREATE_FAILED ||
            describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.ROLLBACK_COMPLETE)
        {
            return false;
        }
    }

    Console.WriteLine("Waiting for CloudFormation stack creation to
complete...");
    await Task.Delay(retryDelay);
    retryCount++;
}

_logger.LogError("Timed out waiting for CloudFormation stack creation to
complete.");
return false;
}

/// <summary>
/// Generates sample logs directly using CloudWatch Logs API.
/// Creates 50,000 log entries spanning 5 minutes.
/// </summary>
/// <returns>True if logs were generated successfully.</returns>
private static async Task<bool> GenerateSampleLogs()
{
    const int totalEntries = 50000;
    const int entriesPerBatch = 10000;
    const int fiveMinutesMs = 5 * 60 * 1000;

    try
    {
        // Calculate timestamps
        var startTimeMs = DateTimeOffset.UtcNow.ToUnixTimeMilliseconds();
        var timestampIncrement = fiveMinutesMs / totalEntries;

        Console.WriteLine($"Generating {totalEntries} log entries...");

        var entryCount = 0;
        var currentTimestamp = startTimeMs;
        var numBatches = totalEntries / entriesPerBatch;

        // Generate and upload logs in batches

```

```

        for (int batchNum = 0; batchNum < numBatches; batchNum++)
        {
            var logEvents = new List<InputLogEvent>();

            for (int i = 0; i < entriesPerBatch; i++)
            {
                logEvents.Add(new InputLogEvent
                {
                    Timestamp =
DateTimeOffset.FromUnixTimeMilliseconds(currentTimestamp).UtcDateTime,
                    Message = $"Entry {entryCount}"
                });

                entryCount++;
                currentTimestamp += timestampIncrement;
            }

            // Upload batch
            var success = await _wrapper.PutLogEventsAsync(_logGroupName,
                _logStreamName, logEvents);
            if (!success)
            {
                _logger.LogError($"Failed to upload batch {batchNum + 1}/
{numBatches}");
                return false;
            }

            Console.WriteLine($"Uploaded batch {batchNum + 1}/{numBatches}");
        }

        // Set query date range (convert milliseconds to seconds for query
API)
        _queryStartDate = startTimeMs / 1000;
        _queryEndDate = (currentTimestamp - timestampIncrement) / 1000;

        Console.WriteLine($"Query start date:
{DateTimeOffset.FromUnixTimeSeconds(_queryStartDate):yyyy-MM-
ddTHH:mm:ss.fffZ}");
        Console.WriteLine($"Query end date:
{DateTimeOffset.FromUnixTimeSeconds(_queryEndDate):yyyy-MM-ddTHH:mm:ss.fffZ}");
        Console.WriteLine($"Successfully uploaded {totalEntries} log
entries");

        return true;

```

```
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, "An error occurred while generating sample
logs.");
        return false;
    }
}

/// <summary>
/// Executes the large query workflow.
/// </summary>
public static async Task ExecuteLargeQuery()
{
    Console.WriteLine("Starting recursive query to retrieve all logs...");
    Console.WriteLine();

    var queryLimit = PromptUserForInteger("Enter the query limit (max 10000)
", 10000);
    if (queryLimit > 10000) queryLimit = 10000;

    var queryString = "fields @timestamp, @message | sort @timestamp asc";

    var stopwatch = Stopwatch.StartNew();
    var allResults = await PerformLargeQuery(_logGroupName, queryString,
_queryStartDate, _queryEndDate, queryLimit);
    stopwatch.Stop();

    Console.WriteLine();
    Console.WriteLine($"Queries finished in
{stopwatch.Elapsed.TotalSeconds:F3} seconds.");
    Console.WriteLine($"Total logs found: {allResults.Count}");

    // Check for duplicates
    Console.WriteLine();
    Console.WriteLine("Checking for duplicate logs...");
    var duplicates = FindDuplicateLogs(allResults);
    if (duplicates.Count > 0)
    {
        Console.WriteLine($"WARNING: Found {duplicates.Count} duplicate log
entries!");
        Console.WriteLine("Duplicate entries (showing first 10):");
        foreach (var dup in duplicates.Take(10))
        {
```

```

        Console.WriteLine($" [{dup.Timestamp}] {dup.Message} (appears
{dup.Count} times)");
    }

    var uniqueCount = allResults.Count - duplicates.Sum(d => d.Count -
1);
    Console.WriteLine($"Unique logs: {uniqueCount}");
}
else
{
    Console.WriteLine("No duplicates found. All logs are unique.");
}
Console.WriteLine();

var viewSample = !_interactive || GetYesNoResponse("Would you like to see
a sample of the logs? (y/n) ");
if (viewSample)
{
    Console.WriteLine();
    Console.WriteLine($"Sample logs (first 10 of {allResults.Count}):");
    for (int i = 0; i < Math.Min(10, allResults.Count); i++)
    {
        var timestamp = allResults[i].Find(f => f.Field ==
"@timestamp")?.Value ?? "N/A";
        var message = allResults[i].Find(f => f.Field ==
"@message")?.Value ?? "N/A";
        Console.WriteLine($"[{timestamp}] {message}");
    }
}
}

/// <summary>
/// Performs a large query using recursive binary search.
/// </summary>
private static async Task<List<List<ResultField>>> PerformLargeQuery(
    string logGroupName,
    string queryString,
    long startTime,
    long endTime,
    int limit)
{
    var queryId = await _wrapper.StartQueryAsync(logGroupName, queryString,
startTime, endTime, limit);
    if (queryId == null)

```

```

    {
        return new List<List<ResultField>>>();
    }

    var results = await PollQueryResults(queryId);
    if (results == null || results.Count == 0)
    {
        return new List<List<ResultField>>>();
    }

    var startDate =
        DateTimeOffset.FromUnixTimeSeconds(startTime).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
    var endDate = DateTimeOffset.FromUnixTimeSeconds(endTime).ToString("yyyy-
MM-ddTHH:mm:ss.fffZ");
    Console.WriteLine($"Query date range: {startDate} ({startTime}s) to
{endDate} ({endTime}s). Found {results.Count} logs.");

    if (results.Count < limit)
    {
        Console.WriteLine($" -> Returning {results.Count} logs (less than
limit of {limit})");
        return results;
    }

    Console.WriteLine($" -> Hit limit of {limit}. Need to split and
recurse.");

    // Get the timestamp of the last log (sorted to find the actual last one)
    var lastLogTimestamp = GetLastLogTimestamp(results);
    if (lastLogTimestamp == null)
    {
        Console.WriteLine($" -> No timestamp found in results. Returning
{results.Count} logs.");
        return results;
    }

    Console.WriteLine($" -> Last log timestamp: {lastLogTimestamp}");

    // Parse the timestamp and add 1 millisecond to avoid querying the same
log again
    var lastLogDate = DateTimeOffset.Parse(lastLogTimestamp + " +0000");
    Console.WriteLine($" -> Last log as DateTimeOffset: {lastLogDate:yyyy-
MM-ddTHH:mm:ss.fffZ} ({lastLogDate.ToUnixTimeSeconds()}s)");

```



```

        var offsetLastLogDate = lastLogDate.AddMilliseconds(1);
        Console.WriteLine($" -> Offset timestamp (last
+ 1ms): {offsetLastLogDate:yyyy-MM-ddTHH:mm:ss.fffZ}
({offsetLastLogDate.ToUnixTimeSeconds()}s)");

        // Convert to seconds, but round UP to the next second to avoid
        overlapping with logs in the same second
        // This ensures we don't re-query logs that share the same second as the
        last log
        var offsetLastLogTime = offsetLastLogDate.ToUnixTimeSeconds();
        if (offsetLastLogDate.Millisecond > 0)
        {
            offsetLastLogTime++; // Move to the next full second
            Console.WriteLine($" -> Adjusted to next full second:
{offsetLastLogTime}s
({DateTimeOffset.FromUnixTimeSeconds(offsetLastLogTime):yyyy-MM-
ddTHH:mm:ss.fffZ})");
        }

        Console.WriteLine($" -> Comparing:
offsetLastLogTime={offsetLastLogTime}s vs endTime={endTime}s");
        Console.WriteLine($" -> End time as date:
{DateTimeOffset.FromUnixTimeSeconds(endTime):yyyy-MM-ddTHH:mm:ss.fffZ}");

        // Check if there's any time range left to query
        if (offsetLastLogTime >= endTime)
        {
            Console.WriteLine($" -> No time range left to query. Offset time
({offsetLastLogTime}s) >= end time ({endTime}s)");
            return results;
        }

        // Split the remaining date range in half
        var (range1Start, range1End, range2Start, range2End) =
SplitDateRange(offsetLastLogTime, endTime);

        var range1StartDate =
DateTimeOffset.FromUnixTimeSeconds(range1Start).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");
        var range1EndDate =
DateTimeOffset.FromUnixTimeSeconds(range1End).ToString("yyyy-MM-
ddTHH:mm:ss.fffZ");

```

```

        var range2StartDate =
            DateTimeOffset.FromUnixTimeSeconds(range2Start).ToString("yyyy-MM-
            ddTHH:mm:ss.fffZ");
        var range2EndDate =
            DateTimeOffset.FromUnixTimeSeconds(range2End).ToString("yyyy-MM-
            ddTHH:mm:ss.fffZ");

        Console.WriteLine($" -> Splitting remaining range:");
        Console.WriteLine($"      Range 1: {range1StartDate} ({range1Start}s) to
        {range1EndDate} ({range1End}s)");
        Console.WriteLine($"      Range 2: {range2StartDate} ({range2Start}s) to
        {range2EndDate} ({range2End}s)");

        // Query both halves recursively
        Console.WriteLine($" -> Querying range 1...");
        var results1 = await PerformLargeQuery(logGroupName, queryString,
        range1Start, range1End, limit);
        Console.WriteLine($" -> Range 1 returned {results1.Count} logs");

        Console.WriteLine($" -> Querying range 2...");
        var results2 = await PerformLargeQuery(logGroupName, queryString,
        range2Start, range2End, limit);
        Console.WriteLine($" -> Range 2 returned {results2.Count} logs");

        // Combine all results
        var allResults = new List<List<ResultField>>(results);
        allResults.AddRange(results1);
        allResults.AddRange(results2);

        Console.WriteLine($" -> Combined total: {allResults.Count} logs
        ({results.Count} + {results1.Count} + {results2.Count})");

        return allResults;
    }

    /// <summary>
    /// Gets the timestamp string of the most recent log from a list of logs.
    /// Sorts timestamps to find the actual last one.
    /// </summary>
    private static string? GetLastLogTimestamp(List<List<ResultField>> logs)
    {
        var timestamps = logs
            .Select(log => log.Find(f => f.Field == "@timestamp")?.Value)
            .Where(t => !string.IsNullOrEmpty(t))

```

```

        .OrderBy(t => t)
        .ToList();

    if (timestamps.Count == 0)
    {
        return null;
    }

    return timestamps[timestamps.Count - 1];
}

/// <summary>
/// Splits a date range in half.
/// Range 2 starts at midpoint + 1 second to avoid overlap.
/// </summary>
private static (long range1Start, long range1End, long range2Start, long
range2End) SplitDateRange(long startTime, long endTime)
{
    var midpoint = startTime + (endTime - startTime) / 2;
    // Range 2 starts at midpoint + 1 to avoid querying the same second twice
    return (startTime, midpoint, midpoint + 1, endTime);
}

/// <summary>
/// Polls for query results until complete.
/// </summary>
private static async Task<List<List<ResultField>>>> PollQueryResults(string
queryId)
{
    int retryCount = 0;
    const int maxRetries = 60;
    const int retryDelay = 1000;

    while (retryCount < maxRetries)
    {
        var response = await _wrapper.GetQueryResultsAsync(queryId);
        if (response == null)
        {
            return null;
        }

        if (response.Status == QueryStatus.Complete)
        {
            return response.Results;
        }
    }
}

```

```

    }

    if (response.Status == QueryStatus.Failed ||
        response.Status == QueryStatus.Cancelled ||
        response.Status == QueryStatus.Timeout ||
        response.Status == QueryStatus.Unknown)
    {
        _logger.LogError($"Query failed with status: {response.Status}");
        return null;
    }

    await Task.Delay(retryDelay);
    retryCount++;
}

_logger.LogError("Timed out waiting for query results.");
return null;
}

/// <summary>
/// Cleans up the resources created during the scenario.
/// </summary>
public static async Task<bool> Cleanup()
{
    var cleanup = !_interactive || GetYesNoResponse(
        "Do you want to delete the CloudFormation stack and all resources?
(y/n) ");

    if (cleanup)
    {
        try
        {
            var stackDeleteSuccess = await
DeleteCloudFormationStack(_stackName, false);
            return stackDeleteSuccess;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "An error occurred while cleaning up the
resources.");
            return false;
        }
    }
}

```

```
        Console.WriteLine($"Resources will remain. Stack name: {_stackName}, Log
group: {_logGroupName}");
        _logger.LogInformation("CloudWatch Logs Large Query scenario is
complete.");
        return true;
    }

    /// <summary>
    /// Deletes the CloudFormation stack and waits for confirmation.
    /// </summary>
    private static async Task<bool> DeleteCloudFormationStack(string stackName,
bool forceDelete)
    {
        var request = new DeleteStackRequest
        {
            StackName = stackName,
        };

        if (forceDelete)
        {
            request.DeletionMode = DeletionMode.FORCE_DELETE_STACK;
        }

        await _amazonCloudFormation.DeleteStackAsync(request);
        Console.WriteLine($"CloudFormation stack '{stackName}' is being deleted.
This may take a few minutes.");

        bool stackDeleted = await WaitForStackDeletion(stackName, forceDelete);

        if (stackDeleted)
        {
            Console.WriteLine($"CloudFormation stack '{stackName}' has been
deleted.");
            return true;
        }
        else
        {
            _logger.LogError($"Failed to delete CloudFormation stack
'{stackName}'.");
            return false;
        }
    }

    /// <summary>
```

```
/// Waits for the stack to be deleted.
/// </summary>
private static async Task<bool> WaitForStackDeletion(string stackName, bool
forceDelete)
{
    int retryCount = 0;
    const int maxRetries = 30;
    const int retryDelay = 10000;

    while (retryCount < maxRetries)
    {
        var describeStacksRequest = new DescribeStacksRequest
        {
            StackName = stackName
        };

        try
        {
            var describeStacksResponse = await
_amazonCloudFormation.DescribeStacksAsync(describeStacksRequest);

            if (describeStacksResponse.Stacks.Count == 0 ||
                describeStacksResponse.Stacks[0].StackStatus ==
StackStatus.DELETE_COMPLETE)
            {
                return true;
            }

            if (!forceDelete && describeStacksResponse.Stacks[0].StackStatus
== StackStatus.DELETE_FAILED)
            {
                return await DeleteCloudFormationStack(stackName, true);
            }
        }
        catch (AmazonCloudFormationException ex) when (ex.ErrorCode ==
"ValidationError")
        {
            return true;
        }

        Console.WriteLine($"Waiting for CloudFormation stack '{stackName}' to
be deleted...");
        await Task.Delay(retryDelay);
        retryCount++;
    }
}
```

```

    }

    _logger.LogError($"Timed out waiting for CloudFormation stack
'{stackName}' to be deleted.");
    return false;
}

/// <summary>
/// Waits with a countdown display.
/// </summary>
private static async Task WaitWithCountdown(int seconds)
{
    for (int i = seconds; i > 0; i--)
    {
        Console.Write($"\\rWaiting: {i} seconds remaining... ");
        await Task.Delay(1000);
    }
    Console.WriteLine("\\rWait complete.                ");
}

/// <summary>
/// Helper method to get a yes or no response from the user.
/// </summary>
private static bool GetYesNoResponse(string question)
{
    Console.WriteLine(question);
    var ynResponse = Console.ReadLine();
    var response = ynResponse != null && ynResponse.Equals("y",
StringComparison.InvariantCultureIgnoreCase);
    return response;
}

/// <summary>
/// Prompts the user for a stack name.
/// </summary>
private static string PromptUserForStackName()
{
    if (_interactive)
    {
        Console.Write($"Enter a name for the CloudFormation stack (press
Enter for default '{_stackName}'): ");
        string? input = Console.ReadLine();
        if (!string.IsNullOrEmpty(input))
        {

```

```

        var regex = "[a-zA-Z][-a-zA-Z0-9]*";
        if (!Regex.IsMatch(input, regex))
        {
            Console.WriteLine($"Invalid stack name. Using default:
{_stackName}");
            return _stackName;
        }
        return input;
    }
}
return _stackName;
}

/// <summary>
/// Prompts the user for input with a default value.
/// </summary>
private static string PromptUserForInput(string prompt, string defaultValue)
{
    if (_interactive)
    {
        Console.Write($"{prompt}(press Enter for default '{defaultValue}'):
");
        string? input = Console.ReadLine();
        return string.IsNullOrEmpty(input) ? defaultValue : input;
    }
    return defaultValue;
}

/// <summary>
/// Prompts the user for an integer value.
/// </summary>
private static int PromptUserForInteger(string prompt, int defaultValue)
{
    if (_interactive)
    {
        Console.Write($"{prompt}(press Enter for default '{defaultValue}'):
");
        string? input = Console.ReadLine();
        if (string.IsNullOrEmpty(input) || !int.TryParse(input, out var
result))
        {
            return defaultValue;
        }
        return result;
    }
}

```



```
    }
    return defaultValue;
}

/// <summary>
/// Prompts the user for a long value.
/// </summary>
private static long PromptUserForLong(string prompt)
{
    if (_interactive)
    {
        Console.Write(prompt);
        string? input = Console.ReadLine();
        if (long.TryParse(input, out var result))
        {
            return result;
        }
    }
    return 0;
}

/// <summary>
/// Finds duplicate log entries based on timestamp and message.
/// </summary>
private static List<(string Timestamp, string Message, int Count)>
FindDuplicateLogs(List<List<ResultField>> logs)
{
    var logSignatures = new Dictionary<string, int>();

    foreach (var log in logs)
    {
        var timestamp = log.Find(f => f.Field == "@timestamp")?.Value ?? "";
        var message = log.Find(f => f.Field == "@message")?.Value ?? "";
        var signature = $"{timestamp}|{message}";

        if (logSignatures.ContainsKey(signature))
        {
            logSignatures[signature]++;
        }
        else
        {
            logSignatures[signature] = 1;
        }
    }
}
```

```

        return logSignatures
            .Where(kvp => kvp.Value > 1)
            .Select(kvp =>
            {
                var parts = kvp.Key.Split('|');
                return (Timestamp: parts[0], Message: parts[1], Count:
kvp.Value);
            })
            .OrderByDescending(x => x.Count)
            .ToList();
    }
}

```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk .NET .
- [GetQueryResults](#)
- [StartQuery](#)

JavaScript

SDK untuk JavaScript (v3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

Ini adalah titik masuknya.

```

// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { CloudWatchLogsClient } from "@aws-sdk/client-cloudwatch-logs";
import { CloudWatchQuery } from "./cloud-watch-query.js";

console.log("Starting a recursive query...");

if (!process.env.QUERY_START_DATE || !process.env.QUERY_END_DATE) {
    throw new Error(
        "QUERY_START_DATE and QUERY_END_DATE environment variables are required.",

```

```

    );
  }

  const cloudWatchQuery = new CloudWatchQuery(new CloudWatchLogsClient({}), {
    logGroupNames: ["/workflows/cloudwatch-logs/large-query"],
    dateRange: [
      new Date(Number.parseInt(process.env.QUERY_START_DATE)),
      new Date(Number.parseInt(process.env.QUERY_END_DATE)),
    ],
  });

  await cloudWatchQuery.run();

  console.log(
    `Queries finished in ${cloudWatchQuery.secondsElapsed} seconds.\nTotal logs found: ${cloudWatchQuery.results.length}`,
  );

```

Ini adalah kelas yang membagi kueri menjadi beberapa langkah jika perlu.

```

// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import {
  StartQueryCommand,
  GetQueryResultsCommand,
} from "@aws-sdk/client-cloudwatch-logs";
import { splitDateRange } from "@aws-doc-sdk-examples/lib/utils/util-date.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

class DateOutOfBoundsError extends Error {}

export class CloudWatchQuery {
  /**
   * Run a query for all CloudWatch Logs within a certain date range.
   * CloudWatch logs return a max of 10,000 results. This class
   * performs a binary search across all of the logs in the provided
   * date range if a query returns the maximum number of results.
   *
   * @param {import('@aws-sdk/client-cloudwatch-logs').CloudWatchLogsClient}
  client
   * @param {{ logGroupNames: string[], dateRange: [Date, Date], queryConfig:
  { limit: number } }} config

```

```

    */
    constructor(client, { logGroupNames, dateRange, queryConfig }) {
        this.client = client;
        /**
         * All log groups are queried.
         */
        this.logGroupNames = logGroupNames;

        /**
         * The inclusive date range that is queried.
         */
        this.dateRange = dateRange;

        /**
         * CloudWatch Logs never returns more than 10,000 logs.
         */
        this.limit = queryConfig?.limit ?? 10000;

        /**
         * @type {import("@aws-sdk/client-cloudwatch-logs").ResultField[][]}
         */
        this.results = [];
    }

    /**
     * Run the query.
     */
    async run() {
        this.secondsElapsed = 0;
        const start = new Date();
        this.results = await this._largeQuery(this.dateRange);
        const end = new Date();
        this.secondsElapsed = (end - start) / 1000;
        return this.results;
    }

    /**
     * Recursively query for logs.
     * @param {[Date, Date]} dateRange
     * @returns {Promise<import("@aws-sdk/client-cloudwatch-logs").ResultField[
[]>}
     */
    async _largeQuery(dateRange) {
        const logs = await this._query(dateRange, this.limit);
    }

```

```

    console.log(
      `Query date range: ${dateRange
        .map((d) => d.toISOString())
        .join(" to ")}. Found ${logs.length} logs.`
    );

    if (logs.length < this.limit) {
      return logs;
    }

    const lastLogDate = this._getLastLogDate(logs);
    const offsetLastLogDate = new Date(lastLogDate);
    offsetLastLogDate.setMilliseconds(lastLogDate.getMilliseconds() + 1);
    const subDateRange = [offsetLastLogDate, dateRange[1]];
    const [r1, r2] = splitDateRange(subDateRange);
    const results = await Promise.all([
      this._largeQuery(r1),
      this._largeQuery(r2),
    ]);
    return [logs, ...results].flat();
  }

  /**
   * Find the most recent log in a list of logs.
   * @param {import("@aws-sdk/client-cloudwatch-logs").ResultField[][]} logs
   */
  _getLastLogDate(logs) {
    const timestamps = logs
      .map(
        (log) =>
          log.find((fieldMeta) => fieldMeta.field === "@timestamp")?.value,
      )
      .filter((t) => !!t)
      .map((t) => `${t}Z`)
      .sort();

    if (!timestamps.length) {
      throw new Error("No timestamp found in logs.");
    }

    return new Date(timestamps[timestamps.length - 1]);
  }

```

```

/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
  return this.client.send(new GetQueryResultsCommand({ queryId }));
}

/**
 * Starts a query and waits for it to complete.
 * @param {[Date, Date]} dateRange
 * @param {number} maxLogs
 */
async _query(dateRange, maxLogs) {
  try {
    const { queryId } = await this._startQuery(dateRange, maxLogs);
    const { results } = await this._waitUntilQueryDone(queryId);
    return results ?? [];
  } catch (err) {
    /**
     * This error is thrown when StartQuery returns an error indicating
     * that the query's start or end date occur before the log group was
     * created.
     */
    if (err instanceof DateOutOfBoundsError) {
      return [];
    }
    throw err;
  }
}

/**
 * Wrapper for the StartQueryCommand. Uses a static query string
 * for consistency.
 * @param {[Date, Date]} dateRange
 * @param {number} maxLogs
 * @returns {Promise<{ queryId: string }>}
 */
async _startQuery([startDate, endDate], maxLogs = 10000) {
  try {
    return await this.client.send(
      new StartQueryCommand({
        logGroupNames: this.logGroupNames,
        queryString: "fields @timestamp, @message | sort @timestamp asc",

```

```

        startTime: startDate.valueOf(),
        endTime: endDate.valueOf(),
        limit: maxLogs,
    )),
    );
} catch (err) {
    /** @type {string} */
    const message = err.message;
    if (message.startsWith("Query's end date and time")) {
        // This error indicates that the query's start or end date occur
        // before the log group was created.
        throw new DateOutOfBoundsError(message);
    }

    throw err;
}

/**
 * Call GetQueryResultsCommand until the query is done.
 * @param {string} queryId
 */
_waitUntilQueryDone(queryId) {
    const getResults = async () => {
        const results = await this._getQueryResults(queryId);
        const queryDone = [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ].includes(results.status);

        return { queryDone, results };
    };

    return retry(
        { intervalInMs: 1000, maxRetries: 60, quiet: true },
        async () => {
            const { queryDone, results } = await getResults();
            if (!queryDone) {
                throw new Error("Query not done.");
            }
        }
    );
}

```

```
        return results;
    },
);
}
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk JavaScript .
 - [GetQueryResults](#)
 - [StartQuery](#)

Python

SDK untuk Python (Boto3)

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di [Repositori Contoh Kode AWS](#).

File ini memanggil modul contoh untuk mengelola CloudWatch kueri melebihi 10.000 hasil.

```
import logging
import os
import sys

import boto3
from botocore.config import Config

from cloudwatch_query import CloudWatchQuery
from date_utilities import DateUtilities

# Configure logging at the module level.
logging.basicConfig(
    level=logging.INFO,
    format="%(asctime)s - %(levelname)s - %(filename)s:%(lineno)d - %(message)s",
)

DEFAULT_QUERY_LOG_GROUP = "/workflows/cloudwatch-logs/large-query"
```



```

class CloudWatchLogsQueryRunner:
    def __init__(self):
        """
        Initializes the CloudWatchLogsQueryRunner class by setting up date
        utilities
        and creating a CloudWatch Logs client with retry configuration.
        """
        self.date_utilities = DateUtilities()
        self.cloudwatch_logs_client = self.create_cloudwatch_logs_client()

    def create_cloudwatch_logs_client(self):
        """
        Creates and returns a CloudWatch Logs client with a specified retry
        configuration.

        :return: A CloudWatch Logs client instance.
        :rtype: boto3.client
        """
        try:
            return boto3.client("logs", config=Config(retries={"max_attempts":
10}))
        except Exception as e:
            logging.error(f"Failed to create CloudWatch Logs client: {e}")
            sys.exit(1)

    def fetch_environment_variables(self):
        """
        Fetches and validates required environment variables for query start and
        end dates.
        Fetches the environment variable for log group, returning the default
        value if it
        does not exist.

        :return: Tuple of query start date and end date as integers and the log
        group.
        :rtype: tuple
        :raises SystemExit: If required environment variables are missing or
        invalid.
        """
        try:
            query_start_date = int(os.environ["QUERY_START_DATE"])
            query_end_date = int(os.environ["QUERY_END_DATE"])
        except KeyError:

```

```

        logging.error(
            "Both QUERY_START_DATE and QUERY_END_DATE environment variables
are required."
        )
        sys.exit(1)
    except ValueError as e:
        logging.error(f"Error parsing date environment variables: {e}")
        sys.exit(1)

    try:
        log_group = os.environ["QUERY_LOG_GROUP"]
    except KeyError:
        logging.warning("No QUERY_LOG_GROUP environment variable, using
default value")
        log_group = DEFAULT_QUERY_LOG_GROUP

    return query_start_date, query_end_date, log_group

def convert_dates_to_iso8601(self, start_date, end_date):
    """
    Converts UNIX timestamp dates to ISO 8601 format using DateUtilities.

    :param start_date: The start date in UNIX timestamp.
    :type start_date: int
    :param end_date: The end date in UNIX timestamp.
    :type end_date: int
    :return: Start and end dates in ISO 8601 format.
    :rtype: tuple
    """
    start_date_iso8601 =
self.date_utilities.convert_unix_timestamp_to_iso8601(
    start_date
)
    end_date_iso8601 = self.date_utilities.convert_unix_timestamp_to_iso8601(
    end_date
)
    return start_date_iso8601, end_date_iso8601

def execute_query(
    self,
    start_date_iso8601,
    end_date_iso8601,
    log_group="/workflows/cloudwatch-logs/large-query",
    query="fields @timestamp, @message | sort @timestamp asc"

```

```

):
    """
    Creates a CloudWatchQuery instance and executes the query with provided
    date range.

    :param start_date_iso8601: The start date in ISO 8601 format.
    :type start_date_iso8601: str
    :param end_date_iso8601: The end date in ISO 8601 format.
    :type end_date_iso8601: str
    :param log_group: Log group to search: "/workflows/cloudwatch-logs/large-
query"
    :type log_group: str
    :param query: Query string to pass to the CloudWatchQuery instance
    :type query: str
    """
    cloudwatch_query = CloudWatchQuery(
        log_group=log_group,
        query_string=query
    )
    cloudwatch_query.query_logs((start_date_iso8601, end_date_iso8601))
    logging.info("Query executed successfully.")
    logging.info(
        f"Queries completed in {cloudwatch_query.query_duration} seconds.
Total logs found: {len(cloudwatch_query.query_results)}"
    )

def main():
    """
    Main function to start a recursive CloudWatch logs query.
    Fetches required environment variables, converts dates, and executes the
    query.
    """
    logging.info("Starting a recursive CloudWatch logs query...")
    runner = CloudWatchLogsQueryRunner()
    query_start_date, query_end_date, log_group =
runner.fetch_environment_variables()
    start_date_iso8601 = DateUtilities.convert_unix_timestamp_to_iso8601(
        query_start_date
    )
    end_date_iso8601 =
DateUtilities.convert_unix_timestamp_to_iso8601(query_end_date)
    runner.execute_query(start_date_iso8601, end_date_iso8601,
log_group=log_group)

```

```
if __name__ == "__main__":
    main()
```

Modul ini memproses CloudWatch kueri melebihi 10.000 hasil.

```
import logging
import time
from datetime import datetime
import threading
import boto3

from date_utilities import DateUtilities

DEFAULT_QUERY = "fields @timestamp, @message | sort @timestamp asc"
DEFAULT_LOG_GROUP = "/workflows/cloudwatch-logs/large-query"

class DateOutOfBoundsError(Exception):
    """Exception raised when the date range for a query is out of bounds."""

    pass

class CloudWatchQuery:
    """
    A class to query AWS CloudWatch logs within a specified date range.

    :vartype date_range: tuple
    :ivar limit: Maximum number of log entries to return.
    :vartype limit: int
    :log_group str: Name of the log group to query
    :query_string str: query
    """

    def __init__(self, log_group: str = DEFAULT_LOG_GROUP, query_string:
str=DEFAULT_QUERY) -> None:
        self.lock = threading.Lock()
        self.log_group = log_group
        self.query_string = query_string
        self.query_results = []
        self.query_duration = None
```

```

self.datetime_format = "%Y-%m-%d %H:%M:%S.%f"
self.date_utilities = DateUtilities()
self.limit = 10000

def query_logs(self, date_range):
    """
    Executes a CloudWatch logs query for a specified date range and
    calculates the execution time of the query.

    :return: A batch of logs retrieved from the CloudWatch logs query.
    :rtype: list
    """
    start_time = datetime.now()

    start_date, end_date = self.date_utilities.normalize_date_range_format(
        date_range, from_format="unix_timestamp", to_format="datetime"
    )

    logging.info(
        f"Original query:"
        f"\n      START:      {start_date}"
        f"\n      END:        {end_date}"
        f"\n      LOG GROUP: {self.log_group}"
    )
    self.recursive_query((start_date, end_date))
    end_time = datetime.now()
    self.query_duration = (end_time - start_time).total_seconds()

def recursive_query(self, date_range):
    """
    Processes logs within a given date range, fetching batches of logs
    recursively if necessary.

    :param date_range: The date range to fetch logs for, specified as a tuple
    (start_timestamp, end_timestamp).
    :type date_range: tuple
    :return: None if the recursive fetching is continued or stops when the
    final batch of logs is processed.
    Although it doesn't explicitly return the query results, this
    method accumulates all fetched logs
    in the `self.query_results` attribute.
    :rtype: None
    """
    batch_of_logs = self.perform_query(date_range)

```

```

# Add the batch to the accumulated logs
with self.lock:
    self.query_results.extend(batch_of_logs)
if len(batch_of_logs) == self.limit:
    logging.info(f"Fetched {self.limit}, checking for more...")
    most_recent_log = self.find_most_recent_log(batch_of_logs)
    most_recent_log_timestamp = next(
        item["value"]
        for item in most_recent_log
        if item["field"] == "@timestamp"
    )
    new_range = (most_recent_log_timestamp, date_range[1])
    midpoint = self.date_utilities.find_middle_time(new_range)

    first_half_thread = threading.Thread(
        target=self.recursive_query,
        args=((most_recent_log_timestamp, midpoint),),
    )
    second_half_thread = threading.Thread(
        target=self.recursive_query, args=((midpoint, date_range[1]),),
    )

    first_half_thread.start()
    second_half_thread.start()

    first_half_thread.join()
    second_half_thread.join()

def find_most_recent_log(self, logs):
    """
    Search a list of log items and return most recent log entry.
    :param logs: A list of logs to analyze.
    :return: log
    :type :return List containing log item details
    """
    most_recent_log = None
    most_recent_date = "1970-01-01 00:00:00.000"

    for log in logs:
        for item in log:
            if item["field"] == "@timestamp":
                logging.debug(f"Compared: {item['value']} to
{most_recent_date}")
                if (

```

```

        self.date_utilities.compare_dates(
            item["value"], most_recent_date
        )
        == item["value"]
    ):
        logging.debug(f"New most recent: {item['value']}")
        most_recent_date = item["value"]
        most_recent_log = log
    logging.info(f"Most recent log date of batch: {most_recent_date}")
    return most_recent_log

def perform_query(self, date_range):
    """
    Performs the actual CloudWatch log query.

    :param date_range: A tuple representing the start and end datetime for
    the query.
    :type date_range: tuple
    :return: A list containing the query results.
    :rtype: list
    """
    client = boto3.client("logs")
    try:
        try:
            start_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
            )
            end_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
            )
            response = client.start_query(
                logGroupName=self.log_group,
                startTime=start_time,
                endTime=end_time,
                queryString=self.query_string,
                limit=self.limit,
            )
            query_id = response["queryId"]
        except client.exceptions.ResourceNotFoundException as e:
            raise DateOutOfBoundsError(f"Resource not found: {e}")
        while True:
            time.sleep(1)

```

```

        results = client.get_query_results(queryId=query_id)
        if results["status"] in [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ]:
            return results.get("results", [])
    except DateOutOfBoundsError:
        return []

def _initiate_query(self, client, date_range, max_logs):
    """
    Initiates the CloudWatch logs query.

    :param date_range: A tuple representing the start and end datetime for
    the query.
    :type date_range: tuple
    :param max_logs: The maximum number of logs to retrieve.
    :type max_logs: int
    :return: The query ID as a string.
    :rtype: str
    """
    try:
        start_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[0])
        )
        end_time = round(
self.date_utilities.convert_iso8601_to_unix_timestamp(date_range[1])
        )
        response = client.start_query(
            logGroupName=self.log_group,
            startTime=start_time,
            endTime=end_time,
            queryString=self.query_string,
            limit=max_logs,
        )
        return response["queryId"]
    except client.exceptions.ResourceNotFoundException as e:
        raise DateOutOfBoundsError(f"Resource not found: {e}")

```



```
def _wait_for_query_results(self, client, query_id):
    """
    Waits for the query to complete and retrieves the results.

    :param query_id: The ID of the initiated query.
    :type query_id: str
    :return: A list containing the results of the query.
    :rtype: list
    """
    while True:
        time.sleep(1)
        results = client.get_query_results(queryId=query_id)
        if results["status"] in [
            "Complete",
            "Failed",
            "Cancelled",
            "Timeout",
            "Unknown",
        ]:
            return results.get("results", [])
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK untuk Python (Boto3).
 - [GetQueryResults](#)
 - [StartQuery](#)

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

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 untuk Java 2.x

Menunjukkan cara membuat acara EventBridge terjadwal Amazon yang memanggil AWS Lambda fungsi. Konfigurasi 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

- CloudWatch Log
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara membuat acara EventBridge terjadwal Amazon yang memanggil AWS Lambda fungsi. Konfigurasi EventBridge untuk menggunakan ekspresi cron untuk menjadwalkan saat fungsi Lambda dipanggil. Dalam contoh ini, Anda membuat fungsi Lambda dengan 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 untuk JavaScript v3](#).

Layanan yang digunakan dalam contoh ini

- CloudWatch Log
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Python

SDK untuk 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
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihat [Menggunakan CloudWatch Log dengan AWS SDK](#). Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Enkripsi tabel pencarian di CloudWatch Log menggunakan AWS Key Management Service

Data tabel pencarian selalu dienkripsi di Log. CloudWatch Secara default, CloudWatch Log menggunakan enkripsi sisi server dengan Advanced Encryption Standard Galois/Counter Mode (AES-GCM) 256-bit untuk mengenkripsi data tabel pencarian saat istirahat. Sebagai alternatif, Anda dapat menggunakan AWS Key Management Service enkripsi ini. Jika Anda melakukannya, enkripsi dilakukan dengan menggunakan AWS KMS kunci. Penggunaan enkripsi AWS KMS diaktifkan pada tingkat tabel pencarian, dengan mengaitkan kunci KMS dengan tabel pencarian, baik saat Anda membuat tabel pencarian atau saat Anda memperbaruinya.

Important

CloudWatch Log hanya mendukung kunci KMS simetris. Jangan gunakan kunci asimetris untuk mengenkripsi data di tabel pencarian Anda. Untuk informasi selengkapnya, lihat [Menggunakan Kunci Simetris dan Asimetris](#).

Setelah Anda mengaitkan kunci KMS dengan tabel pencarian, semua data yang disimpan dalam tabel pencarian dienkripsi menggunakan kunci ini. CloudWatch Log mendekripsi data ini setiap kali diminta. CloudWatch Log harus memiliki izin untuk kunci KMS setiap kali data terenkripsi diminta.

Jika Anda kemudian memisahkan kunci KMS dari tabel pencarian, CloudWatch Log mengenkripsi data menggunakan metode enkripsi default Log. CloudWatch Namun, jika kunci dinonaktifkan atau dihapus sebelum Anda memutuskannya, maka CloudWatch Log tidak dapat membaca data yang dienkripsi dengan kunci itu.

Untuk informasi umum tentang cara CloudWatch Log digunakan AWS KMS untuk mengenkripsi data log, lihat [Enkripsi data log di CloudWatch Log menggunakan AWS Key Management Service](#).

Bagaimana CloudWatch Log menggunakan AWS KMS untuk tabel pencarian

CloudWatch Log menggunakan enkripsi AWS KMS amplop untuk melindungi data tabel pencarian. Saat Anda mengaitkan kunci KMS dengan tabel pencarian, CloudWatch Log mengirimkan `GenerateDataKey` permintaan ke. AWS KMS AWS KMS menghasilkan kunci enkripsi data unik

(DEK) dan mengembalikan salinan teks biasa dan salinan DEK yang dienkripsi. CloudWatch Log menggunakan DEK plaintext untuk mengenkripsi data tabel pencarian, dan kemudian menyimpan DEK terenkripsi bersama data terenkripsi. DEK plaintext tidak disimpan dan dibuang dari memori setelah digunakan.

Ketika CloudWatch Log perlu membaca data tabel pencarian, ia mengirimkan Decrypt permintaan ke AWS KMS DEK terenkripsi. AWS KMS mendekripsi DEK dan mengembalikan DEK teks biasa ke CloudWatch Log, yang kemudian menggunakannya untuk mendekripsi data tabel pencarian.

CloudWatch Log menggunakan konteks enkripsi berikut saat membuat permintaan ke AWS KMS:

```
{
  "aws:logs:arn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-name"
}
```

Anda dapat menggunakan konteks enkripsi ini dalam kebijakan IAM dan kebijakan AWS KMS utama untuk mengontrol akses ke kunci KMS. Untuk informasi selengkapnya, lihat [AWS KMS kunci dan konteks enkripsi](#).

Izin yang diperlukan

Untuk menggunakan AWS KMS enkripsi dengan tabel pencarian, prinsipal IAM harus memiliki AWS KMS izin berikut pada kunci KMS:

- kms:Decrypt
- kms:GenerateDataKey

kms:DecryptIzin diperlukan saat memanggil GetLookupTable tabel pencarian yang dienkripsi dengan kunci KMS, sehingga CloudWatch Log dapat mendekripsi data atas nama Anda.

kms:DecryptIzin juga diperlukan pada kunci (kunci KMS yang digunakan untuk mengenkripsi tabel pencarian) saat memanggil StartQuery dengan kueri yang menggunakan lookup perintah pada tabel pencarian terenkripsi. kms:GenerateDataKeyIzin diperlukan saat memanggil CreateLookupTable atau UpdateLookupTable dengan kunci KMS, sehingga CloudWatch Log dapat menghasilkan kunci enkripsi data untuk mengenkripsi data tabel pencarian.

Selain itu, layanan CloudWatch Log harus memiliki izin untuk menggunakan kunci KMS. Anda memberikan izin ini dengan menambahkan pernyataan kebijakan ke kebijakan kunci KMS, seperti yang dijelaskan di bagian berikut.

Langkah 1: Buat AWS KMS kunci

Untuk membuat kunci KMS simetris, gunakan perintah [create-key](#) berikut:

```
aws kms create-key
```

Output berisi ID kunci dan Amazon Resource Name (ARN) dari kunci. Berikut ini adalah output contoh:

```
{
  "KeyMetadata": {
    "Origin": "AWS_KMS",
    "KeyId": "1234abcd-12ab-34cd-56ef-1234567890ab",
    "Description": "",
    "KeyManager": "CUSTOMER",
    "Enabled": true,
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "CreationDate": 1478910250.94,
    "Arn": "arn:aws:kms:us-west-2:123456789012:key/6f815f63-e628-448c-8251-
e40cb0d29f59",
    "AWSAccountId": "123456789012",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ]
  }
}
```

Rekam kunci ARN. Anda membutuhkannya dalam langkah-langkah berikut.

Langkah 2: Tetapkan izin pada tombol KMS

Secara default, semua AWS KMS kunci bersifat pribadi. Hanya pemilik sumber daya yang dapat menggunakannya untuk mengenkripsi dan mendekripsi data. Anda harus memberikan izin utama layanan CloudWatch Log untuk menggunakan kunci, dan juga memberikan izin peran panggilan untuk menggunakan kunci.

Pertama, simpan kebijakan default untuk kunci KMS Anda seperti `policy.json` menggunakan [get-key-policy](#) perintah berikut:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./policy.json
```

Buka `policy.json` file dalam editor teks dan tambahkan pernyataan berikut untuk memberikan izin utama layanan CloudWatch Log untuk menggunakan kunci. Contoh ini menggunakan Condition bagian yang cocok dengan konteks enkripsi untuk membatasi kunci ke tabel pencarian tertentu.

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "logs.region.amazonaws.com"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-name",
      "aws:SourceAccount": "account-id",
      "aws:SourceArn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-name"
    }
  }
}
```

Selanjutnya, tambahkan izin ke peran yang akan memanggil CloudWatch Log `CreateLookupTable` atau `UpdateLookupTable` APIs. CloudWatch Log digunakan `kms:ViaService` untuk melakukan panggilan ke AWS KMS atas nama pelanggan. Untuk informasi lebih lanjut, lihat [kms: ViaService](#).

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account-id:role/role-name"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey"
  ],
}
```

```
"Resource": "*",
"Condition": {
  "StringEquals": {
    "kms:ViaService": [
      "logs.region.amazonaws.com"
    ]
  }
}
```

Terakhir, tambahkan kebijakan yang diperbarui menggunakan [put-key-policy](#) perintah berikut:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Langkah 3: Kaitkan kunci KMS dengan tabel pencarian

Anda dapat mengaitkan kunci KMS dengan tabel pencarian saat Anda membuatnya menggunakan CreateLookupTable API, atau memperbarui tabel pencarian yang ada menggunakan API. UpdateLookupTable APIs Keduanya adalah bagian dari AWSLogsConfigService.

Untuk mengaitkan kunci KMS dengan tabel pencarian saat Anda membuatnya

Gunakan CreateLookupTable API dan tentukan kmsKeyArn parameter dengan ARN kunci KMS Anda:

```
aws logs create-lookup-table \
  --lookup-table-name my-lookup-table \
  --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```

Untuk mengaitkan kunci KMS dengan tabel pencarian yang ada

Gunakan UpdateLookupTable API dan tentukan kmsKeyArn parameter dengan ARN kunci KMS Anda:

```
aws logs update-lookup-table \
  --lookup-table-name my-lookup-table \
  --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```


Pertimbangan-pertimbangan

- Setelah Anda mengaitkan atau memisahkan kunci KMS dari tabel pencarian, diperlukan waktu hingga lima menit agar operasi diterapkan.
- Jika Anda mencabut akses CloudWatch Log ke kunci terkait atau menghapus kunci KMS terkait, data tabel pencarian terenkripsi Anda di CloudWatch Log tidak dapat diambil lagi.
- Untuk melakukan langkah-langkah dalam topik ini, Anda harus memiliki izin berikut: `kms:CreateKey`, `kms:GetKeyPolicy`, `kms:PutKeyPolicy`, dan izin CloudWatch Log yang sesuai untuk memanggil `CreateLookupTable` atau `UpdateLookupTable`.

Keamanan di Amazon CloudWatch Log

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model tanggung jawab bersama](#) menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [Program AWS Kepatuhan](#) . Untuk mempelajari tentang program kepatuhan yang berlaku WorkSpaces, lihat [AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan](#) .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup kepekaan data Anda, persyaratan perusahaan, serta peraturan perundangan yang berlaku

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon CloudWatch Logs. Ini menunjukkan kepada Anda cara mengonfigurasi CloudWatch Log Amazon untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya CloudWatch Log Anda.

Konten

- [Perlindungan data di Amazon CloudWatch Logs](#)
- [Manajemen identitas dan akses untuk Amazon CloudWatch Logs](#)
- [Validasi kepatuhan untuk Amazon Logs CloudWatch](#)
- [Ketahanan di Log Amazon CloudWatch](#)
- [Keamanan infrastruktur di Amazon CloudWatch Logs](#)
- [Menggunakan CloudWatch Log dengan titik akhir VPC antarmuka](#)

Perlindungan data di Amazon CloudWatch Logs

Note

Selain informasi berikut tentang perlindungan data umum di AWS, CloudWatch Log juga memungkinkan Anda untuk melindungi data sensitif dalam peristiwa log dengan menutupinya. Untuk informasi selengkapnya, lihat [Membantu melindungi data log sensitif dengan masking](#).

[Model tanggung jawab AWS bersama model tanggung](#) berlaku untuk perlindungan data di Amazon CloudWatch Logs. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam [Pertanyaan Umum Privasi Data](#). Lihat informasi tentang perlindungan data di Eropa di pos blog [Model Tanggung Jawab Bersama dan GDPR AWS](#) di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensial dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat [Bekerja dengan CloudTrail jejak](#) di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.

- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di [Standar Pemrosesan Informasi Federal \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan CloudWatch Log atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Enkripsi saat diam

CloudWatch Log melindungi data saat istirahat menggunakan enkripsi. Semua grup log dienkripsi. Secara default, layanan CloudWatch Log mengelola enkripsi sisi server dan menggunakan enkripsi sisi server dengan 256-bit Advanced Encryption Standard Galois/Counter Mode (AES-GCM) untuk mengenkripsi data log saat istirahat.

Jika Anda ingin mengelola kunci yang digunakan untuk mengenkripsi dan mendekripsi log Anda, gunakan kunci AWS KMS. Untuk informasi selengkapnya, lihat [Enkripsi data log di CloudWatch Log menggunakan AWS Key Management Service](#).

Enkripsi saat bergerak

CloudWatch Log menggunakan end-to-end enkripsi data dalam perjalanan. Layanan CloudWatch Log mengelola kunci enkripsi sisi server.

Manajemen identitas dan akses untuk Amazon CloudWatch Logs

Akses ke Amazon CloudWatch Log memerlukan kredensial yang AWS dapat digunakan untuk mengautentikasi permintaan Anda. Kredensi tersebut harus memiliki izin untuk mengakses AWS sumber daya, seperti untuk mengambil data CloudWatch Log tentang sumber daya cloud Anda. Bagian berikut memberikan rincian tentang bagaimana Anda dapat menggunakan [AWS Identity and Access Management \(IAM\)](#) dan CloudWatch Log untuk membantu mengamankan sumber daya Anda dengan mengontrol siapa yang dapat mengaksesnya:

- [Autentikasi](#)
- [Kontrol akses](#)

Autentikasi

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Kontrol akses

Anda dapat memiliki kredensi yang valid untuk mengautentikasi permintaan Anda, tetapi kecuali Anda memiliki izin, Anda tidak dapat membuat atau mengakses sumber daya Log. CloudWatch Misalnya, Anda harus memiliki izin untuk membuat pengaliran log, membuat grup log, dan sebagainya.

Bagian berikut menjelaskan cara mengelola izin untuk CloudWatch Log. Anda sebaiknya membaca gambaran umum terlebih dahulu.

- [Ikhtisar mengelola izin akses ke sumber daya CloudWatch Log Anda](#)
- [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Log CloudWatch](#)
- [CloudWatch Referensi izin log](#)

Ikhtisar mengelola izin akses ke sumber daya CloudWatch Log Anda

Untuk memberikan akses dan menambahkan izin bagi pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

Buat rangkaian izin. Ikuti instruksi di [Buat rangkaian izin](#) dalam Panduan Pengguna AWS IAM Identity Center .

- Pengguna yang dikelola di IAM melalui penyedia identitas:

Buat peran untuk federasi identitas. Ikuti instruksi dalam [Buat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) dalam Panduan Pengguna IAM.

- Pengguna IAM:

- Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam [Buat peran untuk pengguna IAM](#) dalam Panduan Pengguna IAM.
- (Tidak disarankan) Lampirkan kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam [Menambahkan izin ke pengguna \(konsol\)](#) dalam Panduan Pengguna IAM.

Topik

- [CloudWatch Log sumber daya dan operasi](#)
- [Memahami kepemilikan sumber daya](#)
- [Mengelola akses ke sumber daya](#)
- [Menentukan elemen kebijakan: Tindakan, efek, dan penanggung jawab](#)
- [Menetapkan ketentuan dalam kebijakan](#)

CloudWatch Log sumber daya dan operasi

Di CloudWatch Log, sumber daya utama adalah grup log, aliran log, dan tujuan. CloudWatch Log tidak mendukung subsumber daya (sumber daya lain untuk digunakan dengan sumber daya utama).

Sumber daya dan subsumber daya ini memiliki Nama Sumber Daya Amazon (ARNs) unik yang terkait dengannya seperti yang ditunjukkan pada tabel berikut.

Tipe sumber daya	Format ARN

Tipe sumber daya	Format ARN
Grup log	Kedua hal berikut ini digunakan. Yang kedua, dengan : * di akhir, adalah apa yang dikembalikan oleh perintah <code>describe-log-groups</code> CLI dan API. <code>DescribeLogGroups</code> <code>arn:aws:logs: ::log-group: <i>region account-id log_group_name</i></code> <code>arn:aws:logs: ::log-group:: * <i>region account-id log_group_name</i></code> Gunakan versi pertama, tanpa trailing : *, dalam situasi berikut: • Di bidang <code>logGroupIdentifier</code> input di banyak CloudWatch Logs APIs. • Di <code>resourceArn</code> lapangan dalam penandaan APIs • Dalam IAM kebijakan, saat menentukan izin untuk TagResource, UntagResource, dan ListTagsForResource Gunakan versi kedua, dengan tambahan : *, untuk merujuk ke ARN saat menentukan izin dalam kebijakan IAM untuk semua tindakan API lainnya.
Pengaliran log	<code>arn:aws:logs: ::log-group ::log-stream: <i>region account-id log_group_name log-stream-name</i></code>

Tipe sumber daya	Format ARN
Tujuan	arn:aws:logs: :tujuan: <i>region account-id destination_name</i>

Untuk informasi selengkapnya ARNs, lihat [ARNs](#) di Panduan Pengguna IAM. Untuk informasi tentang CloudWatch Log ARNs, lihat [Amazon Resource Names \(ARNs\)](#) di Referensi Umum Amazon Web Services. Untuk contoh kebijakan yang mencakup CloudWatch Log, lihat [Menggunakan kebijakan berbasis identitas \(kebijakan IAM\) untuk Log CloudWatch](#).

CloudWatch Log menyediakan serangkaian operasi untuk bekerja dengan sumber daya CloudWatch Log. Untuk daftar operasi yang tersedia, lihat [CloudWatch Referensi izin log](#).

Memahami kepemilikan sumber daya

AWS Akun memiliki sumber daya yang dibuat di akun, terlepas dari siapa yang membuat sumber daya. Secara khusus, pemilik sumber daya adalah AWS akun [entitas utama](#) (yaitu, akun root, pengguna, atau peran IAM) yang mengautentikasi permintaan pembuatan sumber daya. Contoh berikut menggambarkan cara kerjanya:

- Jika Anda menggunakan kredensi akun root AWS akun Anda untuk membuat grup log, AWS akun Anda adalah pemilik sumber daya CloudWatch Log.
- Jika Anda membuat pengguna di AWS akun Anda dan memberikan izin untuk membuat sumber daya CloudWatch Log kepada pengguna tersebut, pengguna dapat membuat sumber daya CloudWatch Log. Namun, AWS akun Anda, yang menjadi milik pengguna, memiliki sumber daya CloudWatch Log.
- Jika Anda membuat peran IAM di AWS akun Anda dengan izin untuk membuat sumber daya CloudWatch Log, siapa pun yang dapat mengambil peran tersebut dapat membuat sumber daya CloudWatch Log. AWS Akun Anda, yang menjadi milik peran tersebut, memiliki sumber daya CloudWatch Log.

Mengelola akses ke sumber daya

Kebijakan izin menjelaskan siapa yang memiliki akses ke suatu objek. Bagian berikut menjelaskan opsi yang tersedia untuk membuat kebijakan izin.

 Note

Bagian ini membahas penggunaan IAM dalam konteks Log. CloudWatch Bagian ini tidak memberikan informasi yang mendetail tentang layanan IAM. Untuk dokumentasi lengkap IAM, lihat [Apa yang Dimaksud dengan IAM?](#) dalam Panduan Pengguna IAM. Untuk informasi tentang sintaksis dan penjelasan kebijakan IAM, lihat [Referensi Kebijakan IAM](#) dalam Panduan Pengguna IAM.

Kebijakan yang melekat pada identitas IAM disebut sebagai kebijakan berbasis identitas (kebijakan IAM) dan kebijakan yang melekat pada sumber daya disebut sebagai kebijakan berbasis sumber daya. CloudWatch Log mendukung kebijakan berbasis identitas, dan kebijakan berbasis sumber daya untuk tujuan, yang digunakan untuk mengaktifkan langganan lintas akun. Untuk informasi selengkapnya, lihat [Langganan lintas akun Lintas wilayah](#).

Topik

- [Izin grup log dan Wawasan Kontributor](#)
- [Kebijakan berbasis sumber daya](#)

Izin grup log dan Wawasan Kontributor

Contributor Insights adalah fitur CloudWatch yang memungkinkan Anda menganalisis data dari grup log dan membuat deret waktu yang menampilkan data kontributor. Anda dapat melihat metrik tentang kontributor N teratas, total kontributor unik, dan penggunaannya. Untuk informasi selengkapnya, lihat [Menggunakan Wawasan Kontributor untuk Menganalisis Data Berkardinalitas Tinggi](#).

Saat Anda memberikan izin `cloudwatch:PutInsightRule` dan `cloudwatch:GetInsightRuleReport` izin kepada pengguna, pengguna tersebut dapat membuat aturan yang mengevaluasi grup log apa pun di CloudWatch Log dan kemudian melihat hasilnya. Hasil dapat memuat data kontributor untuk grup log tersebut. Pastikan untuk memberikan izin ini hanya kepada pengguna yang harus dapat melihat data ini.

Kebijakan berbasis sumber daya

CloudWatch Log mendukung kebijakan berbasis sumber daya untuk tujuan, yang dapat Anda gunakan untuk mengaktifkan langganan lintas akun. Untuk informasi selengkapnya, lihat [Langkah 1: Buat tujuan](#). Tujuan dapat dibuat menggunakan [PutDestination](#) API, dan Anda dapat menambahkan kebijakan sumber daya ke tujuan menggunakan [PutDestinationPolicy](#) API. Contoh berikut

mengizinkan akun lain dengan ID akun AWS 111122223333 untuk melanggankan grup lognya ke tujuan `arn:aws:logs:us-east-1:123456789012:destination:testDestination`.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": "logs:PutSubscriptionFilter",
      "Resource": "arn:aws:logs:us-east-1:123456789012:destination:testDestination"
    }
  ]
}
```

Menentukan elemen kebijakan: Tindakan, efek, dan penanggung jawab

Untuk setiap sumber daya CloudWatch Log, layanan mendefinisikan satu set operasi API. Untuk memberikan izin untuk operasi API ini, CloudWatch Log mendefinisikan serangkaian tindakan yang dapat Anda tentukan dalam kebijakan. Beberapa operasi API dapat memerlukan izin untuk lebih dari satu tindakan untuk melakukan operasi API. Untuk informasi selengkapnya tentang sumber daya dan operasi API, lihat [CloudWatch Log sumber daya dan operasi](#) dan [CloudWatch Referensi izin log](#).

Berikut ini adalah elemen-elemen kebijakan dasar:

- Sumber daya – Anda menggunakan Amazon Resource Name (ARN) untuk mengidentifikasi sumber daya yang diberlakukan oleh kebijakan tersebut. Untuk informasi selengkapnya, lihat [CloudWatch Log sumber daya dan operasi](#).
- Tindakan – Anda menggunakan kata kunci tindakan untuk mengidentifikasi operasi sumber daya yang ingin Anda izinkan atau tolak. Misalnya, izin `logs:DescribeLogGroups` memungkinkan pengguna untuk melakukan `DescribeLogGroups` operasi.
- Pengaruh – Anda menetapkan pengaruh, baik memperbolehkan atau menolak, ketika pengguna meminta tindakan tertentu. Jika Anda tidak secara eksplisit memberikan akses ke (mengizinkan)

sumber daya, akses akan ditolak secara implisit. 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.

- **Principal** – Dalam kebijakan berbasis identitas (Kebijakan IAM), pengguna yang kebijakannya terlampir adalah principal yang implisit. Untuk kebijakan berbasis sumber daya, Anda menentukan pengguna, akun, layanan, atau entitas lain yang ingin Anda terima izin (hanya berlaku untuk kebijakan berbasis sumber daya). CloudWatch Log mendukung kebijakan berbasis sumber daya untuk tujuan.

Untuk mempelajari selengkapnya tentang sintaksis dan deskripsi kebijakan IAM, lihat [Referensi Kebijakan IAM AWS](#) dalam Panduan Pengguna IAM.

Untuk tabel yang menampilkan semua tindakan API CloudWatch Log dan sumber daya yang diterapkan, lihat [CloudWatch Referensi izin log](#).

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 menentukan kondisi dalam bahasa kebijakan, lihat [Kondisi](#) dalam Panduan Pengguna IAM.

Untuk menyatakan kondisi, Anda menggunakan kunci kondisi standar. Untuk daftar kunci konteks yang didukung oleh setiap AWS layanan dan daftar kunci kebijakan AWS-wide, lihat Kunci [tindakan, sumber daya, dan kondisi untuk AWS layanan dan kunci konteks kondisi AWS global](#).

Note

Anda dapat menggunakan tag untuk mengontrol akses ke sumber CloudWatch Log, termasuk grup log dan tujuan. Akses ke aliran log dikontrol pada tingkat grup log, karena hubungan hierarkis antara grup log dan aliran log. Untuk informasi selengkapnya tentang penggunaan tanda untuk mengendalikan akses, lihat [Mengendalikan akses ke sumber daya Amazon Web Services menggunakan tanda](#).

Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk Log CloudWatch

Topik ini memberikan contoh kebijakan berbasis identitas di mana administrator akun dapat melampirkan kebijakan izin ke identitas IAM (yaitu, pengguna, grup, dan peran).

Important

Kami menyarankan Anda terlebih dahulu meninjau topik pengantar yang menjelaskan konsep dasar dan opsi yang tersedia bagi Anda untuk mengelola akses ke sumber daya CloudWatch Log Anda. Untuk informasi selengkapnya, lihat [Ikhtisar mengelola izin akses ke sumber daya CloudWatch Log Anda](#).

Topik ini mencakup hal-hal berikut:

- [Izin diperlukan untuk menggunakan konsol CloudWatch](#)
- [AWS kebijakan terkelola \(standar\) untuk CloudWatch Log](#)
- [Contoh kebijakan yang dikelola pelanggan](#)

Berikut ini adalah contoh kebijakan izin:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "arn:aws:logs:*:*:*"
      ]
    }
  ]
}
```

```
}  
]  
}
```

Kebijakan ini memiliki satu pernyataan yang memberikan izin untuk membuat grup log dan pengaliran log, untuk mengunggah log acara ke pengaliran log, dan daftar detail tentang pengaliran log.

Karakter wildcard (*) di akhir nilai Resource berarti bahwa pernyataan memungkinkan izin untuk tindakan `logs:CreateLogGroup`, `logs:CreateLogStream`, `logs:PutLogEvents`, dan `logs:DescribeLogStreams` di setiap grup log. Untuk membatasi izin ini ke grup log tertentu, ganti karakter wildcard (*) di ARN sumber daya dengan ARN grup log tertentu. Untuk informasi selengkapnya tentang bagian-bagian dalam pernyataan kebijakan IAM, lihat [Referensi Elemen Kebijakan IAM](#) dalam Panduan Pengguna IAM. Untuk daftar yang menampilkan semua tindakan CloudWatch Log, lihat [CloudWatch Referensi izin log](#).

Izin diperlukan untuk menggunakan konsol CloudWatch

Agar pengguna dapat bekerja dengan CloudWatch Log di CloudWatch konsol, pengguna tersebut harus memiliki seperangkat izin minimum yang memungkinkan pengguna mendeskripsikan AWS sumber daya lain di AWS akun mereka. Untuk menggunakan CloudWatch Log di CloudWatch konsol, Anda harus memiliki izin dari layanan berikut:

- CloudWatch
- CloudWatch Log
- OpenSearch Layanan
- IAM
- Kinesis
- Lambda
- Amazon S3

Jika Anda membuat kebijakan IAM yang lebih ketat dari izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana dimaksudkan untuk pengguna dengan kebijakan IAM tersebut. Untuk memastikan bahwa pengguna tersebut masih dapat menggunakan CloudWatch konsol, lampirkan juga kebijakan `CloudWatchReadOnlyAccess` terkelola ke pengguna, seperti yang dijelaskan dalam [AWS kebijakan terkelola \(standar\) untuk CloudWatch Log](#).

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau API CloudWatch Log.

Set lengkap izin yang diperlukan untuk bekerja dengan CloudWatch konsol untuk pengguna yang tidak menggunakan konsol untuk mengelola langganan log adalah:

- jam tangan awan: GetMetricData
- jam tangan awan: ListMetrics
- log: CancelExportTask
- log: CreateExportTask
- log: CreateLogGroup
- log: CreateLogStream
- log: DeleteLogGroup
- log: DeleteLogStream
- log: DeleteMetricFilter
- log: DeleteQueryDefinition
- log: DeleteRetentionPolicy
- log: DeleteSubscriptionFilter
- log: DescribeExportTasks
- log: DescribeLogGroups
- log: DescribeLogStreams
- log: DescribeMetricFilters
- log: DescribeQueryDefinitions
- log: DescribeQueries
- log: DescribeSubscriptionFilters
- log: FilterLogEvents
- log: GetLogEvents
- log: GetLogGroupFields
- log: GetLogRecord
- log: GetQueryResults

- log: PutMetricFilter
- log: PutQueryDefinition
- log: PutRetentionPolicy
- log: StartQuery
- log: StopQuery
- log: PutSubscriptionFilter
- log: TestMetricFilter

Untuk pengguna yang juga akan menggunakan konsol untuk mengelola langganan log, izin berikut juga diperlukan:

- es: DescribeElasticsearchDomain
- es: ListDomainNames
- saya: AttachRolePolicy
- saya: CreateRole
- saya: GetPolicy
- saya: GetPolicyVersion
- saya: GetRole
- saya: ListAttachedRolePolicies
- saya: ListRoles
- kinesis: DescribeStreams
- kinesis: ListStreams
- lambda: AddPermission
- lambda: CreateFunction
- lambda: GetFunctionConfiguration
- lambda: ListAliases
- lambda: ListFunctions
- lambda: ListVersionsByFunction
- lambda: RemovePermission
- s3: ListBuckets

AWS kebijakan terkelola (standar) untuk CloudWatch Log

AWS mengatasi banyak kasus penggunaan umum dengan menyediakan kebijakan IAM mandiri yang dibuat dan dikelola oleh. AWS Kebijakan terkelola memberikan izin yang diperlukan untuk kasus penggunaan umum sehingga Anda tidak perlu menyelidiki izin apa yang diperlukan. Untuk informasi selengkapnya, lihat [Kebijakan Terkelola AWS](#) dalam Panduan Pengguna IAM.

Kebijakan AWS terkelola berikut, yang dapat Anda lampirkan ke pengguna dan peran di akun Anda, khusus untuk CloudWatch Log:

- `CloudWatchLogsFullAccess`— Memberikan akses penuh ke CloudWatch Log.
- `CloudWatchLogsReadOnlyAccess`— Memberikan akses hanya-baca ke Log. CloudWatch

`CloudWatchLogsFullAccess`

`CloudWatchLogsFullAccess` Kebijakan ini memberikan akses penuh ke CloudWatch Log. Kebijakan ini menyertakan `cloudwatch:GenerateQueryResultsSummary` izin `cloudwatch:GenerateQuery` dan, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri [Wawasan CloudWatch Log](#) dari prompt bahasa alami. Untuk melihat isi lengkap kebijakan, lihat [CloudWatchLogsFullAccess](#) di Panduan Referensi Kebijakan AWS Terkelola.

`CloudWatchLogsReadOnlyAccess`

`CloudWatchLogsReadOnlyAccess` Kebijakan ini memberikan akses hanya-baca ke Log. CloudWatch Ini menyertakan `cloudwatch:GenerateQuery` dan `cloudwatch:GenerateQueryResultsSummary` izin, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri [Wawasan CloudWatch Log](#) dari prompt bahasa alami. Untuk melihat isi lengkap kebijakan, lihat [CloudWatchLogsReadOnlyAccess](#) di Panduan Referensi Kebijakan AWS Terkelola.

`CloudWatchOpenSearchDashboardsFullAccess`

`CloudWatchOpenSearchDashboardsFullAccess` Kebijakan ini memberikan akses untuk membuat, mengelola, dan menghapus integrasi dengan OpenSearch Service, dan untuk membuat hapus dan mengelola dasbor log vended dalam integrasi tersebut. Untuk informasi selengkapnya, lihat [Analisis dengan Amazon OpenSearch Service](#).

Untuk melihat isi lengkap kebijakan, lihat [CloudWatchOpenSearchDashboardsFullAccess](#) di Panduan Referensi Kebijakan AWS Terkelola.

CloudWatchOpenSearchDashboardAccess

CloudWatchOpenSearchDashboardAccessKebijakan ini memberikan akses untuk melihat dasbor log vended yang dibuat dengan analitik. Amazon OpenSearch Service Untuk informasi selengkapnya, lihat [Analisis dengan Amazon OpenSearch Service](#).

Important

Selain memberikan kebijakan ini, agar peran atau pengguna dapat melihat dasbor log penjual otomatis, Anda juga harus menentukannya saat membuat integrasi dengan Layanan. OpenSearch Untuk informasi selengkapnya, lihat [Langkah 1: Buat integrasi dengan OpenSearch Layanan](#).

Untuk melihat isi lengkap kebijakan, lihat [CloudWatchOpenSearchDashboardAccess](#) di Panduan Referensi Kebijakan AWS Terkelola.

CloudWatchLogsCrossAccountSharingConfiguration

CloudWatchLogsCrossAccountSharingConfigurationKebijakan ini memberikan akses untuk membuat, mengelola, dan melihat tautan Pengelola Akses Observabilitas untuk berbagi sumber CloudWatch Log antar akun. Untuk informasi lebih lanjut, lihat [CloudWatch observabilitas lintas akun](#).

Untuk melihat isi lengkap kebijakan, lihat [CloudWatchLogsCrossAccountSharingConfiguration](#) di Panduan Referensi Kebijakan AWS Terkelola.

CloudWatchLogsAPIKeyAkses

Kebijakan CloudWatchLogsAPIKeyAccess memungkinkan otentikasi kunci CloudWatch Logs API dan konsumsi log terenkripsi. Kebijakan ini memberikan izin untuk mengautentikasi menggunakan token pembawa dan menulis peristiwa log ke Log, dengan AWS KMS izin tambahan untuk mendekripsi dan membuat kunci data saat CloudWatch log dienkripsi.

Kebijakan ini memberikan izin berikut:

- **logs**— Memungkinkan prinsipal untuk mengautentikasi melalui token pembawa kunci API dan menulis peristiwa log ke aliran Log. CloudWatch
- **kms**— Memungkinkan prinsipal membaca metadata kunci, menghasilkan AWS KMS kunci data untuk enkripsi, dan mendekripsi data. Izin ini mendukung CloudWatch Log terenkripsi dengan

memungkinkan layanan mengenkripsi data log menggunakan kunci yang dikelola pelanggan. AWS KMS Akses dibatasi untuk operasi yang dipanggil melalui layanan CloudWatch Log.

Untuk melihat detail selengkapnya tentang kebijakan, termasuk versi terbaru dari dokumen kebijakan JSON, lihat [CloudWatchLogsAPIKeyAkses](#) di Panduan Referensi Kebijakan AWS Terkelola.

CloudWatch Log pembaruan ke kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk CloudWatch Log sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen CloudWatch Log.

Ubah	Deskripsi	Date
CloudWatchLogsAPIKeyAkses — Kebijakan baru.	CloudWatch Log menambahkan CloudWatchLogsAPIKeyAccess kebijakan terkelola baru. Kebijakan ini memungkinkan otentikasi kunci API CloudWatch Log dan konsumsi log terenkripsi, memberikan izin untuk mengautentikasi menggunakan token pembawa dan menulis peristiwa log ke Log. CloudWatch	Februari 17, 2026
CloudWatchLogsFullAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsFullAccess. Izin untuk tindakan administrasi observabilitas ditambahkan untuk memungkinkan	Desember 02, 2025

Ubah	Deskripsi	Date
	akses hanya-baca ke pipeline telemetri dan integrasi tabel S3.	
CloudWatchLogsReadOnlyAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsReadOnlyAccess. Izin untuk tindakan administrasi observabilitas ditambahkan untuk memungkinkan akses hanya-baca ke pipeline telemetri dan integrasi tabel S3.	Desember 02, 2025
CloudWatchLogsFullAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsFullAccess. Izin untuk cloudwatch:GenerateQueryResultsSummary ditambahkan untuk memungkinkan pembuatan ringkasan bahasa alami dari hasil kueri.	Mei 20, 2025

Ubah	Deskripsi	Date
CloudWatchLogsReadOnlyAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsReadOnlyAccess. Izin untuk <code>cloudwatch:GenerateQueryResultsSummary</code> ditambahkan untuk memungkinkan pembuatan ringkasan bahasa alami dari hasil kueri.	Mei 20, 2025
CloudWatchLogsFullAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsFullAccess. Izin untuk Amazon OpenSearch Service dan IAM ditambahkan, untuk mengaktifkan integrasi CloudWatch Log dengan OpenSearch Layanan untuk beberapa fitur.	Desember 1, 2024

Ubah	Deskripsi	Date
CloudWatchOpenSearchDashboardsFullAccess — Kebijakan IAM baru.	CloudWatch Log menambahkan kebijakan IAM baru, CloudWatchOpenSearchDashboardsFullAccess.- Kebijakan ini memberikan akses untuk membuat, mengelola, dan menghapus integrasi dengan OpenSearch Layanan, dan untuk membuat, mengelola, dan menghapus dasbor log vendes dalam integrasi tersebut. Untuk informasi selengkapnya, lihat Analisis dengan Amazon OpenSearch Service .	Desember 1, 2024
CloudWatchOpenSearchDashboardAccess — Kebijakan IAM baru.	CloudWatch Log menambahkan kebijakan IAM baru, CloudWatchOpenSearchDashboardAccess.- Kebijakan ini memberikan akses untuk melihat dasbor log vendes yang didukung oleh. Amazon OpenSearch Service Untuk informasi selengkapnya, lihat Analisis dengan Amazon OpenSearch Service .	Desember 1, 2024

Ubah	Deskripsi	Date
CloudWatchLogsFullAccess — Perbarui ke kebijakan yang ada.	CloudWatch Log menambahkan izin ke CloudWatchLogsFullAccess. <code>cloudwatch:GenerateQuery</code> Izin ditambahkan, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri Wawasan CloudWatch Log dari prompt bahasa alami.	27 November 2023
CloudWatchLogsReadOnlyAccess — Perbarui ke kebijakan yang ada.	CloudWatch menambahkan izin untuk CloudWatchLogsReadOnlyAccess. <code>cloudwatch:GenerateQuery</code> Izin ditambahkan, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri Wawasan CloudWatch Log dari prompt bahasa alami.	27 November 2023

Ubah	Deskripsi	Date
CloudWatchLogsReadOnlyAccess – Pembaruan ke kebijakan yang ada	CloudWatch Log menambahkan izin ke CloudWatchLogsReadOnlyAccess. Izin <code>logs:StartLiveTail</code> dan <code>logs:StopLiveTail</code> izin ditambahkan sehingga pengguna dengan kebijakan ini dapat menggunakan konsol untuk memulai dan menghentikan sesi ekor langsung CloudWatch Log. Untuk informasi selengkapnya, silakan lihat Menggunakan live tail untuk melihat log mendekati waktu nyata.	6 Juni 2023
CloudWatchLogsCrossAccountSharingConfiguration – Kebijakan baru	CloudWatch Log menambahkan kebijakan baru untuk memungkinkan Anda mengelola tautan pengamatan CloudWatch lintas akun yang berbagi grup CloudWatch log Log. Untuk informasi lebih lanjut, lihat CloudWatch observabilitas lintas akun	27 November 2022

Ubah	Deskripsi	Date
CloudWatchLogsReadOnlyAccess – Pembaruan ke kebijakan yang ada	CloudWatch Log menambahkan izin ke CloudWatchLogsReadOnlyAccess. Izin <code>oam:ListSinks</code> dan <code>oam:ListAttachedLinks</code> izin ditambahkan sehingga pengguna dengan kebijakan ini dapat menggunakan konsol untuk melihat data yang dibagikan dari akun sumber dalam pengamatan CloudWatch lintas akun.	27 November 2022

Contoh kebijakan yang dikelola pelanggan

Anda dapat membuat kebijakan IAM kustom Anda sendiri untuk mengizinkan izin untuk tindakan dan sumber CloudWatch daya Log. Anda dapat menyematkan kebijakan khusus ini untuk pengguna atau grup yang memerlukan izin tersebut.

Di bagian ini, Anda dapat menemukan contoh kebijakan pengguna yang memberikan izin untuk berbagai tindakan CloudWatch Log. Kebijakan ini berfungsi saat Anda menggunakan CloudWatch Logs API, AWS SDKs, atau file AWS CLI.

Contoh

- [Contoh 1: Izinkan akses penuh ke CloudWatch Log](#)
- [Contoh 2: Izinkan akses hanya-baca ke Log CloudWatch](#)
- [Contoh 3: Izinkan akses ke satu grup log](#)

Contoh 1: Izinkan akses penuh ke CloudWatch Log

Kebijakan berikut memungkinkan pengguna mengakses semua tindakan CloudWatch Log.

JSON

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

Contoh 2: Izinkan akses hanya-baca ke Log CloudWatch

AWS menyediakan `CloudWatchLogsReadOnlyAccess` kebijakan yang memungkinkan akses hanya-baca ke data CloudWatch Log. Kebijakan ini mencakup izin berikut.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:Describe*",
        "logs:Get*",
        "logs:List*",
        "logs:StartQuery",
        "logs:StopQuery",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "logs:StartLiveTail",
        "logs:StopLiveTail",
        "cloudwatch:GenerateQuery"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

```
    ]
  }
```

Contoh 3: Izinkan akses ke satu grup log

Kebijakan berikut mengizinkan pengguna untuk membaca dan menulis log acara dalam satu grup log tertentu.

Important

: *Di akhir nama grup log di Resource baris diperlukan untuk menunjukkan bahwa kebijakan berlaku untuk semua aliran log di grup log ini. Jika Anda menghilangkan : *, kebijakan tidak akan diberlakukan.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:PutLogEvents",
        "logs:GetLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:us-west-2:123456789012:log-
group:SampleLogGroupName:*"
    }
  ]
}
```

Menggunakan penandaan dan kebijakan IAM untuk mengendalikan di tingkat grup log

Anda dapat memberi pengguna akses ke grup log tertentu serta mencegah mereka mengakses grup log lainnya. Untuk melakukannya, beri tanda grup log Anda dan gunakan kebijakan IAM yang merujuk ke tanda tersebut. Untuk menerapkan tag ke grup log, Anda harus memiliki

logs:TagLogGroup izin logs:TagResource atau izin. Ini berlaku baik jika Anda menetapkan tag ke grup log saat Anda membuatnya. atau menetapkannya nanti.

Untuk informasi selengkapnya tentang penandaan grup log, lihat [Tandai grup log di Amazon CloudWatch Logs](#).

Ketika Anda menandai grup log, Anda kemudian dapat memberikan kebijakan IAM kepada pengguna untuk mengizinkan akses hanya ke grup log dengan tanda tertentu. Sebagai contoh, pernyataan kebijakan berikut ini memberikan akses ke hanya grup log dengan nilai Green untuk kunci tanda Team.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:*"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:ResourceTag/Team": "Green"
        }
      }
    }
  ]
}
```

Operasi StopLiveTailAPI StopQuerydan tidak berinteraksi dengan AWS sumber daya dalam pengertian tradisional. Mereka tidak mengembalikan data apa pun, memasukkan data apa pun, atau memodifikasi sumber daya dengan cara apa pun. Sebaliknya, mereka hanya beroperasi pada sesi ekor langsung tertentu atau kueri Wawasan CloudWatch Log tertentu, yang tidak dikategorikan sebagai sumber daya. Akibatnya, ketika Anda menentukan Resource bidang dalam kebijakan IAM untuk operasi ini, Anda harus menetapkan nilai Resource bidang sebagai*, seperti pada contoh berikut.

JSON

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

Untuk informasi selengkapnya tentang menggunakan pernyataan kebijakan IAM, lihat [Mengendalikan Akses Menggunakan Kebijakan](#) dalam Panduan Pengguna IAM.

CloudWatch Referensi izin log

Ketika Anda mengatur [Kontrol akses](#) dan menulis kebijakan izin yang dapat Anda lampirkan ke identitas IAM (kebijakan berbasis identitas), Anda dapat menggunakan tabel berikut sebagai referensi. Tabel mencantumkan setiap operasi API CloudWatch Log dan tindakan terkait yang dapat Anda berikan izin untuk melakukan tindakan. Anda menentukan tindakan di bidang Action kebijakan. Untuk Resource bidang, Anda dapat menentukan ARN grup log atau aliran log, atau menentukan * untuk mewakili semua sumber CloudWatch Log.

Anda dapat menggunakan kunci kondisi AWS-wide dalam kebijakan CloudWatch Log Anda untuk menyatakan kondisi. Untuk daftar lengkap kunci AWS-wide, lihat [Kunci Konteks Kondisi AWS Global dan IAM di Panduan Pengguna IAM](#).

Note

Untuk menentukan tindakan, gunakan awalan logs: diikuti dengan nama operasi API. Misalnya: logs:CreateLogGroup, logs:CreateLogStream, atau logs:* (untuk semua tindakan CloudWatch Log).

CloudWatch Log operasi API dan izin yang diperlukan untuk tindakan

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
CancelExportTask	<code>logs:CancelExportTask</code> Diperlukan untuk membatalkan tugas ekspor yang tertunda atau berjalan.
CreateExportTask	<code>logs:CreateExportTask</code> Diperlukan untuk mengekspor data dari grup log ke bucket Amazon S3.
CreateLogGroup	<code>logs:CreateLogGroup</code> Diperlukan untuk membuat grup log baru.
CreateLogStream	<code>logs:CreateLogStream</code> Diperlukan untuk membuat aliran log baru dalam grup log.
DeleteDestination	<code>logs:DeleteDestination</code> Diperlukan untuk menghapus tujuan log dan menonaktifkan penyaring berlangganan apa pun.
DeleteLogGroup	<code>logs:DeleteLogGroup</code> Diperlukan untuk menghapus grup log dan semua peristiwa log yang diarsipkan yang terkait.
DeleteLogStream	<code>logs:DeleteLogStream</code>

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
	Diperlukan untuk menghapus aliran log dan peristiwa log yang diarsipkan yang terkait.
DeleteMetricFilter	<code>logs:DeleteMetricFilter</code> Diperlukan untuk menghapus penyaring metrik yang terkait dengan grup log.
DeleteQueryDefinition	<code>logs:DeleteQueryDefinition</code> Diperlukan untuk menghapus definisi kueri yang disimpan di Wawasan CloudWatch Log.
DeleteResourcePolicy	<code>logs:DeleteResourcePolicy</code> Diperlukan untuk menghapus kebijakan sumber daya CloudWatch Log.
DeleteRetentionPolicy	<code>logs:DeleteRetentionPolicy</code> Diperlukan untuk menghapus kebijakan penyimpanan grup log.
DeleteSubscriptionFilter	<code>logs:DeleteSubscriptionFilter</code> Diperlukan untuk menghapus penyaring berlangganan yang terkait dengan grup log.
DescribeDestinations	<code>logs:DescribeDestinations</code> Diperlukan untuk melihat semua destinasi yang terkait dengan akun.

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
DescribeExportTasks	<code>logs:DescribeExportTasks</code> Diperlukan untuk melihat semua tugas ekspor yang terkait dengan akun.
DescribeLogGroups	<code>logs:DescribeLogGroups</code> Diperlukan untuk melihat semua grup log yang terkait dengan akun.
DescribeLogStreams	<code>logs:DescribeLogStreams</code> Diperlukan untuk melihat semua aliran log yang terkait dengan grup log.
DescribeMetricFilters	<code>logs:DescribeMetricFilters</code> Diperlukan untuk melihat semua metrik yang terkait dengan grup log.
DescribeQueryDefinitions	<code>logs:DescribeQueryDefinitions</code> Diperlukan untuk melihat daftar definisi kueri yang disimpan di Wawasan CloudWatch Log.
DescribeQueries	<code>logs:DescribeQueries</code> Diperlukan untuk melihat daftar kueri Wawasan CloudWatch Log yang dijadwalkan, dijalankan, atau baru-baru ini dikeluarkan.

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
DescribeResourcePolicies	<code>logs:DescribeResourcePolicies</code> Diperlukan untuk melihat daftar kebijakan sumber daya CloudWatch Log.
DescribeSubscriptionFilters	<code>logs:DescribeSubscriptionFilters</code> Diperlukan untuk melihat semua penyaring berlangganan yang terkait dengan grup log.
FilterLogEvents	<code>logs:FilterLogEvents</code> Diperlukan untuk mengurutkan peristiwa log berdasarkan pola penyaringan grup log.
GetLogEvents	<code>logs:GetLogEvents</code> Diperlukan untuk mengambil kejadian log dari aliran log.
GetLogGroupFields	<code>logs:GetLogGroupFields</code> Diperlukan untuk mengambil daftar kolom yang disertakan dalam peristiwa log di grup log.
GetLogRecord	<code>logs:GetLogRecord</code> Diperlukan untuk mengambil rincian dari satu peristiwa log.

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
GetLogObject	logs: GetLogRecord Diperlukan untuk mengambil konten sebagian besar peristiwa log yang telah dicerna melalui API. PutOpenTelemetryLogs
GetQueryResults	logs: GetQueryResults Diperlukan untuk mengambil hasil kueri Wawasan CloudWatch Log.
ListEntitiesForLogGroup (CloudWatch izin khusus konsol)	logs: ListEntitiesForLogGroup Diperlukan untuk menemukan entitas yang terkait dengan grup log. Diperlukan untuk menjelajahi log terkait di dalam CloudWatch konsol.
ListLogGroupsForEntity (CloudWatch izin khusus konsol)	logs: ListLogGroupsForEntity Diperlukan untuk menemukan grup log yang terkait dengan entitas. Diperlukan untuk menjelajahi log terkait di dalam CloudWatch konsol.
ListTagsLogGroup	logs: ListTagsLogGroup Diperlukan untuk membuat daftar tanda yang terkait dengan grup log.
ListLogGroups	logs: DescribeLogGroups Diperlukan untuk melihat semua grup log yang terkait dengan akun.

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
PutDestination	<code>logs:PutDestination</code> Diperlukan untuk membuat atau memperbarui aliran log tujuan (seperti aliran Kinesis).
PutDestinationPolicy	<code>logs:PutDestinationPolicy</code> Diperlukan untuk membuat atau memperbarui kebijakan akses yang terkait dengan tujuan log yang sudah ada.
PutLogEvents	<code>logs:PutLogEvents</code> Diperlukan untuk mengunggah kumpulan peristiwa log ke aliran log.
PutMetricFilter	<code>logs:PutMetricFilter</code> Diperlukan untuk membuat atau memperbarui penyaring metrik dan mengaitkannya dengan grup log.
PutQueryDefinition	<code>logs:PutQueryDefinition</code> Diperlukan untuk menyimpan kueri di Wawasan CloudWatch Log, termasuk kueri tersimpan dengan parameter.
PutResourcePolicy	<code>logs:PutResourcePolicy</code> Diperlukan untuk membuat kebijakan sumber daya CloudWatch Log.

CloudWatch Log operasi API	Izin yang diperlukan (tindakan API)
PutRetentionPolicy	<code>logs:PutRetentionPolicy</code> Diperlukan untuk mengatur jumlah hari untuk menyimpan peristiwa (penyimpanan) log dalam grup log.
PutSubscriptionFilter	<code>logs:PutSubscriptionFilter</code> Diperlukan untuk membuat atau memperbarui penyaring berlangganan dan mengaitkannya dengan grup log.
StartQuery	<code>logs:StartQuery</code> Diperlukan untuk memulai kueri Wawasan CloudWatch Log. Untuk menjalankan kueri yang disimpan dengan parameter, Anda juga perlu <code>logs:DescribeQueryDefinitions</code>.
StopQuery	<code>logs:StopQuery</code> Diperlukan untuk menghentikan kueri Wawasan CloudWatch Log yang sedang berlangsung.
TagLogGroup	<code>logs:TagLogGroup</code> Perlu menambahkan atau memperbarui tag grup log.
TestMetricFilter	<code>logs:TestMetricFilter</code> Diperlukan untuk menguji pola penyaringan terhadap sampel pesan peristiwa log.

Menggunakan peran terkait layanan untuk Log CloudWatch

Amazon CloudWatch Logs menggunakan AWS Identity and Access Management peran [terkait layanan](#) (IAM). Peran terkait layanan adalah jenis unik peran IAM yang ditautkan langsung ke Log. CloudWatch Peran terkait layanan telah ditentukan sebelumnya oleh CloudWatch Log dan menyertakan semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran terkait layanan membuat pengaturan CloudWatch Log lebih efisien karena Anda tidak diharuskan menambahkan izin yang diperlukan secara manual. CloudWatch Log mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya CloudWatch Log yang dapat mengambil peran tersebut. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin. Kebijakan izin itu tidak dapat dilampirkan ke entitas IAM lainnya.

Untuk informasi tentang layanan lain yang mendukung peran yang terhubung dengan layanan, lihat [AWS Layanan yang Bekerja dengan IAM](#). Cari layanan yang memiliki Yes di kolom Service-Linked Role. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk Log CloudWatch

CloudWatch Log menggunakan nama peran terkait layanan. `AWSServiceRoleForLogDelivery` CloudWatch Log menggunakan peran terkait layanan ini untuk menulis log langsung ke Firehose. Untuk informasi selengkapnya, lihat [Aktifkan logging dari AWS layanan](#).

Peran tertaut layanan `AWSServiceRoleForLogDelivery` memercayai layanan berikut untuk mengambil peran tersebut:

- `logs.amazonaws.com`

Kebijakan izin peran memungkinkan CloudWatch Log untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `firehose:PutRecord` dan `firehose:PutRecordBatch` pada semua aliran Firehose yang memiliki tag dengan `LogDeliveryEnabled` kunci dengan nilai `True`. Tag ini secara otomatis dilampirkan ke aliran Firehose saat Anda membuat langganan untuk mengirimkan log ke Firehose.

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM untuk membuat, mengedit, atau menghapus peran yang terhubung dengan layanan. Entitas ini dapat berupa pengguna, grup, atau peran. Untuk informasi lebih lanjut, lihat [Izin Peran yang Terhubung dengan Layanan](#) di Panduan Pengguna IAM.

Membuat peran terkait layanan untuk Log CloudWatch

Anda tidak perlu membuat peran yang terhubung dengan layanan secara manual. Saat Anda menyiapkan log untuk dikirim langsung ke aliran Firehose di Konsol Manajemen AWS, the, atau AWS API AWS CLI, CloudWatch Log akan membuat peran terkait layanan untuk Anda.

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 mengatur kembali log untuk dikirim langsung ke aliran Firehose, CloudWatch Log akan membuat peran terkait layanan untuk Anda lagi.

Mengedit peran terkait layanan untuk Log CloudWatch

CloudWatch Log tidak memungkinkan Anda untuk mengedit `AWSServiceRoleForLogDelivery`, atau peran terkait layanan lainnya, setelah Anda membuatnya. Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat menyunting penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit Peran Tertaut Layanan](#) dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk Log CloudWatch

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, sebaiknya hapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

Note

Jika layanan CloudWatch Log menggunakan peran saat Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus sumber daya CloudWatch Log yang digunakan oleh `AWSServiceRoleForLogDelivery` peran terkait layanan

- Berhenti mengirim log langsung ke aliran Firehose.

Untuk menghapus peran tertaut layanan secara manual menggunakan IAM

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForLogDelivery` terkait layanan. Untuk informasi selengkapnya, lihat [Menghapus Peran yang Terhubung dengan Layanan](#)

Wilayah yang Didukung untuk CloudWatch peran terkait layanan Log

CloudWatch Log mendukung penggunaan peran terkait layanan di semua AWS Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat [CloudWatch Logs Regions and Endpoints](#).

CloudWatch Memutakhirkan log ke peran terkait AWS layanan

Lihat detail tentang pembaruan peran terkait AWS layanan untuk CloudWatch Log sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen CloudWatch Log.

Ubah	Deskripsi	Date
AWSServiceRoleForLogDelivery kebijakan peran terkait layanan — Pembaruan ke kebijakan yang ada	CloudWatch Log mengubah izin dalam kebijakan IAM yang terkait dengan peran terkait <code>AWSServiceRoleForLogDelivery</code> layanan. Perubahan berikut dibuat: Kunci <code>firehose:ResourceTag/LogDeliveryEnabled</code>: <code>"true"</code> kondisi diubah menjadi <code>aws:Resou</code>	15 Juli 2021

Ubah	Deskripsi	Date
	<pre> rceTag/LogDelivery Enabled": "true" . </pre>	
CloudWatch Log mulai melacak perubahan	CloudWatch Log mulai melacak perubahan untuk kebijakan AWS terkelolanya.	10 Juni 2021

Validasi kepatuhan untuk Amazon Logs CloudWatch

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [Program AWS Kepatuhan Program AWS](#) .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#) .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. Untuk informasi selengkapnya tentang tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS, lihat [Dokumentasi AWS Keamanan](#).

Ketahanan di Log Amazon CloudWatch

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. Wilayah memberikan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik, yang terkoneksi melalui jaringan latensi rendah, throughput tinggi, dan sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat [Infrastruktur AWS Global](#).

Keamanan infrastruktur di Amazon CloudWatch Logs

Sebagai layanan terkelola, Amazon CloudWatch Logs dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat [Keamanan AWS Cloud](#). Untuk mendesain AWS lingkungan Anda menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur dalam Kerangka Kerja](#) yang AWS Diarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses CloudWatch Log melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Pengangkutan (TLS). Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Sandi cocok dengan sistem kerahasiaan maju sempurna (perfect forward secrecy, PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Menggunakan CloudWatch Log dengan titik akhir VPC antarmuka

Jika Anda menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk meng-host AWS sumber daya Anda, Anda dapat membuat koneksi pribadi antara VPC dan Log Anda. CloudWatch Anda dapat menggunakan koneksi ini untuk mengirim CloudWatch log ke Log tanpa mengirimnya melalui internet. CloudWatch Log mendukung titik akhir IPv4 VPC di semua Wilayah, dan mendukung IPv6 titik akhir di semua Wilayah.

Amazon VPC adalah AWS layanan yang dapat Anda gunakan untuk meluncurkan AWS sumber daya di jaringan virtual yang Anda tentukan. Dengan VPC, Anda memiliki kendali terhadap pengaturan jaringan, seperti rentang alamat IP, subnet, tabel rute, dan pintu masuk jaringan. Untuk menghubungkan VPC Anda ke CloudWatch Log, Anda menentukan titik akhir VPC antarmuka untuk Log. CloudWatch Jenis titik akhir ini memungkinkan Anda untuk menghubungkan VPC Anda ke layanan AWS . Endpoint menyediakan konektivitas yang andal dan dapat diskalakan ke CloudWatch Log 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, sebuah AWS teknologi yang memungkinkan komunikasi pribadi antara AWS layanan menggunakan antarmuka jaringan elastis dengan alamat IP pribadi. Untuk informasi selengkapnya, lihat [Baru — AWS PrivateLink untuk AWS Layanan](#).

Langkah-langkah berikut ditujukan untuk para pengguna Amazon VPC. Untuk informasi selengkapnya, silakan lihat [Getting Started](#) di Panduan Pengguna Amazon VPC.

Ketersediaan

CloudWatch Log saat ini mendukung titik akhir VPC di semua AWS Wilayah, termasuk Wilayah. AWS GovCloud (US)

Membuat titik akhir VPC untuk Log CloudWatch

Untuk mulai menggunakan CloudWatch Log dengan VPC Anda, buat antarmuka VPC endpoint untuk Log. CloudWatch Layanan yang harus dipilih adalah `com.amazonaws.Region.log`. Untuk terhubung dengan titik akhir FIPS, layanan yang harus dipilih adalah `com.amazonaws.Region.logs-fips` Anda tidak perlu mengubah pengaturan apa pun untuk CloudWatch Log. Untuk informasi selengkapnya, silakan lihat [Membuat sebuah Titik Akhir Antarmuka](#) dalam Panduan Pengguna Amazon VPC.

Beberapa CloudWatch Log APIs, seperti `StartLiveTail` dan `GetLogObject`, dihosting di bawah titik akhir dan titik akhir VPC yang berbeda: `stream-logs.Region.amazonaws.com` Untuk membuat titik akhir VPC antarmuka untuk ini APIs, layanan yang harus dipilih adalah `com.amazonaws.Region.stream-log`. Untuk terhubung dengan titik akhir FIPS, layanan yang harus dipilih adalah `com.amazonaws.Region.stream-logs-fips`

Menguji koneksi antara VPC dan Log CloudWatch

Setelah Anda membuat titik akhir, Anda dapat menguji koneksi.

Untuk menguji koneksi antara VPC dan titik akhir Log CloudWatch

1. Connect ke instans Amazon EC2 yang berada di VPC Anda. Untuk informasi tentang menghubungkan, lihat [Hubungkan ke Instans Linux Anda](#) atau [Connect ke Instans Windows Anda](#) dalam dokumentasi Amazon EC2.
2. Dari contoh, gunakan AWS CLI untuk membuat entri log di salah satu grup log yang ada.

Pertama, buat file JSON dengan log acara. Stempel waktu harus ditetapkan sebagai angka dalam milidetik setelah 1 Jan 1970 00:00:00 UTC.

```
[  
  {
```

```
    "timestamp": 1533854071310,  
    "message": "VPC Connection Test"  
  }  
]
```

Kemudian, gunakan perintah `put-log-events` untuk membuat entri log:

```
aws logs put-log-events --log-group-name LogGroupName --log-stream-  
name LogStreamName --log-events file://JSONFileName
```

Jika respons terhadap perintah termasuk `nextSequenceToken`, perintah telah berhasil dan VPC endpoint Anda bekerja.

Mengontrol akses ke titik akhir VPC CloudWatch Log

Kebijakan titik akhir VPC adalah kebijakan sumber daya IAM yang Anda lampirkan ke titik akhir ketika membuat atau mengubah titik akhir. Jika Anda tidak melampirkan kebijakan ketika membuat titik akhir, kami melampirkan kebijakan default untuk Anda sehingga memungkinkan akses penuh ke layanan. Kebijakan endpoint tidak mengesampingkan atau mengganti kebijakan IAM atau kebijakan khusus layanan. Ini adalah kebijakan terpisah untuk mengendalikan akses dari titik akhir ke layanan tertentu.

Kebijakan titik akhir harus ditulis dalam format JSON.

Untuk informasi selengkapnya, silakan lihat [Mengendalikan Akses ke Layanan dengan titik akhir VPC](#) dalam Panduan Pengguna Amazon VPC.

Berikut ini adalah contoh kebijakan endpoint untuk CloudWatch Log. Kebijakan ini memungkinkan pengguna yang terhubung ke CloudWatch Log melalui VPC untuk membuat aliran log dan mengirim CloudWatch log ke Log, serta mencegah mereka melakukan tindakan Log lainnya CloudWatch .

```
{  
  "Statement": [  
    {  
      "Sid": "PutOnly",  
      "Principal": "*",  
      "Action": [  
        "logs:CreateLogStream",  
        "logs:PutLogEvents"  
      ],  
    },  
  ],  
}
```

```
    "Effect": "Allow",  
    "Resource": "*"    
  }  
]  
}
```

Untuk mengubah kebijakan titik akhir VPC untuk Log CloudWatch

1. Buka konsol Amazon VPC di. <https://console.aws.amazon.com/vpc/>
2. Di panel navigasi, pilih Titik akhir.
3. Jika Anda belum membuat endpoint untuk CloudWatch Log, pilih Create Endpoint. Kemudian pilih com.amazonaws. **Region**.logs dan pilih Create endpoint.
4. Pilih com.amazonaws. **Region**.logs endpoint, dan pilih tab Policy di bagian bawah layar.
5. Pilih Edit Kebijakan dan buat perubahan pada kebijakan.

Support untuk kunci konteks VPC

CloudWatch Log mendukung `aws:SourceVpc` dan kunci `aws:SourceVpce` konteks yang dapat membatasi akses ke titik akhir VPC tertentu VPCs atau spesifik. Kunci ini bekerja hanya ketika pengguna menggunakan VPC endpoint. Untuk informasi selengkapnya, lihat [Kunci yang Tersedia untuk Beberapa Layanan](#) di Panduan Pengguna IAM.

Logging CloudWatch Logs API dan operasi konsol di AWS CloudTrail

Amazon CloudWatch Logs terintegrasi dengan [AWS CloudTrail](#), layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau Layanan AWS. CloudTrail menangkap panggilan API untuk CloudWatch Log sebagai peristiwa. Panggilan yang diambil mencakup panggilan dari konsol CloudWatch Log dan panggilan kode ke operasi CloudWatch Logs API. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk CloudWatch Log, alamat IP dari mana permintaan dibuat, kapan dibuat, dan detail tambahan.

Setiap entri peristiwa atau log berisi informasi tentang entitas 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 tersebut dibuat dengan kredensial keamanan sementara untuk satu 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 Konsol Manajemen AWS 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](#).

CloudWatch Log mendukung pencatatan tindakan berikut sebagai peristiwa dalam file CloudTrail log:

- [AssociateKmsKey](#)
- [CancelExportTask](#)
- [CreateDelivery](#)
- [CreateExportTask](#)
- [CreateLogAnomalyDetector](#)
- [CreateLogGroup](#)
- [CreateLogStream](#)
- [DeleteAccountPolicy](#)

- [DeleteDataProtectionPolicy](#)
- [DeleteDelivery](#)
- [DeleteDeliveryDestination](#)
- [DeleteDeliveryDestinationPolicy](#)
- [DeleteDeliverySource](#)
- [DeleteDestination](#)
- [DeleteIndexPolicy](#)
- [DeleteIntegration](#)
- [DeleteLogAnomalyDetector](#)
- [DeleteLogGroup](#)
- [DeleteLogStream](#)
- [DeleteMetricFilter](#)
- [DeleteQueryDefinition](#)
- [DeleteResourcePolicy](#)
- [DeleteRetentionPolicy](#)
- [DeleteSubscriptionFilter](#)
- [DeleteTransformer](#)
- [DescribeAccountPolicies](#)
- [DescribeConfigurationTemplates](#)
- [DescribeDeliveries](#)
- [DescribeDeliveryDestinations](#)
- [DescribeDeliverySources](#)
- [DescribeDestinations](#)
- [DescribeExportTasks](#)
- [DescribeFieldIndexes](#)
- [DescribeIndexPolicies](#)
- [DescribeLogGroups](#)
- [DescribeLogStreams](#)
- [DescribeMetricFilters](#)

- [DescribeQueries](#)
- [DescribeQueryDefinitions](#)
- [DescribeResourcePolicies](#)
- [DescribeSubscriptionFilters](#)
- [DisassociateKmsKey](#)
- [FilterLogEvents](#)
- [GetDataProtectionPolicy](#)
- [GetDelivery](#)
- [GetDeliveryDestination](#)
- [GetDeliveryDestinationPolicy](#)
- [GetDeliverySource](#)
- [GetIntegration](#)
- [GetLogAnomalyDetector](#)
- [GetLogEvents](#)
- [GetLogGroupFields](#)
- [GetLogRecord](#)
- [GetQueryResults](#)
- [GetTransformer](#)
- [ListAnomalies](#)
- [ListIntegrations](#)
- [ListLogAnomalyDetectors](#)
- [ListLogGroups](#)
- [ListLogGroupsForQuery](#)
- [ListTagsForResource](#)
- [ListTagsLogGroup](#)
- [PutAccountPolicy](#)
- [PutDataProtectionPolicy](#)
- [PutDeliveryDestination](#)
- [PutDeliveryDestinationPolicy](#)

- [PutDeliverySource](#)
- [PutDestination](#)
- [PutDestinationPolicy](#)
- [PutIndexPolicy](#)
- [PutIntegration](#)
- [PutMetricFilter](#)
- [PutQueryDefinition](#)
- [PutResourcePolicy](#)
- [PutRetentionPolicy](#)
- [PutSubscriptionFilter](#)
- [PutTransformer](#)
- [StartLiveTail](#)
- [StartQuery](#)
- [StopQuery](#)
- [TagResource](#)
- [TestMetricFilter](#)
- [TestTransformer](#)
- [UntagResource](#)
- [UpdateAnomaly](#)
- [UpdateDeliveryConfiguration](#)
- [UpdateLogAnomalyDetector](#)

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan tersebut dibuat dengan kredensial root atau pengguna IAM.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi lain, lihat [Elemen userIdentity CloudTrail](#) .

Informasi pembuatan kueri di CloudTrail

CloudTrail logging untuk acara konsol generator Query juga didukung. Generator kueri saat ini didukung untuk Wawasan CloudWatch Log dan Wawasan CloudWatch Metrik. Dalam CloudTrail peristiwa ini, eventSource adalah `monitoring.amazonaws.com`.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan GenerateQuery tindakan di Wawasan CloudWatch Log.

```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:assumed-role/role_name",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::111222333444:role/Administrator",
        "accountId": "123456789012",
        "userName": "SAMPLE_NAME"
      },
      "attributes": {
        "creationDate": "2020-04-08T21:43:24Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2020-04-08T23:06:30Z",
  "eventSource": "monitoring.amazonaws.com",
  "eventName": "GenerateQuery",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "exampleUserAgent",
  "requestParameters": {
    "query_ask": "****",
    "query_type": "LogsInsights",
    "logs_insights": {
      "fields": "****",
      "log_group_names": ["yourloggroup"]
    }
  }
}
```

```
    },
    "include_description": true
  },
  "responseElements": null,
  "requestID": "2f56318c-cfbd-4b60-9d93-1234567890",
  "eventID": "52723fd9-4a54-478c-ac55-1234567890",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```

Memahami entri file log

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Entri file log berikut menunjukkan bahwa pengguna bernama CreateExportTask tindakan CloudWatch Log.

```
{
  "eventVersion": "1.03",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "EX_PRINCIPAL_ID",
    "arn": "arn:aws:iam::123456789012:user/someuser",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "someuser"
  },
  "eventTime": "2016-02-08T06:35:14Z",
  "eventSource": "logs.amazonaws.com",
  "eventName": "CreateExportTask",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "aws-sdk-ruby2/2.0.0.rc4 ruby/1.9.3 x86_64-linux Seahorse/0.1.0",
```

```
    "requestParameters": {
      "destination": "yourdestination",
      "logGroupName": "yourloggroup",
      "to": 123456789012,
      "from": 0,
      "taskName": "yourtask"
    },
    "responseElements": {
      "taskId": "15e5e534-9548-44ab-a221-64d9d2b27b9b"
    },
    "requestID": "1cd74c1c-ce2e-12e6-99a9-8dbb26bd06c9",
    "eventID": "fd072859-bd7c-4865-9e76-8e364e89307c",
    "eventType": "AwsApiCall",
    "apiVersion": "20140328",
    "recipientAccountId": "123456789012"
  }
```

Pemantauan dengan CloudWatch metrik

Anda dapat menggunakan tabel di bagian ini untuk meninjau metrik yang dikirimkan Amazon CloudWatch Logs ke Amazon CloudWatch setiap menit.

CloudWatch Metrik log

Namespace AWS/Logs mencakup metrik berikut.

Metrik	Deskripsi
CallCount	Jumlah operasi API tertentu yang dilakukan di akun Anda. CallCount adalah metrik penggunaan layanan CloudWatch Log. Untuk informasi selengkapnya, lihat CloudWatch Metrik penggunaan layanan log. Dimensi yang Valid: Kelas, Sumber Daya, Layanan, Jenis Statistik Valid: Sum Unit: Tidak ada
DeliveryErrors	Jumlah peristiwa log yang CloudWatch Log menerima kesalahan saat meneruskan data ke tujuan langganan. Jika layanan tujuan mengembalikan kesalahan yang dapat dicoba ulang seperti pengecualian pembatasan atau pengecualian layanan yang dapat dicoba ulang (misalnya HTTP 5xx), CloudWatch Log terus mencoba lagi pengiriman hingga 24 jam. CloudWatch Log tidak mencoba mengirim ulang jika kesalahan adalah kesalahan yang tidak dapat dicoba ulang, seperti atau. <code>AccessDeniedException</code> <code>ResourceNotFoundException</code> Dimensi yang Valid: LogGroupName DestinationType,, FilterName, PolicyLevel Statistik Valid: Sum Unit: Tidak ada

Metrik	Deskripsi
DeliveryThrottling	Jumlah peristiwa log yang CloudWatch Log dibatasi saat meneruskan data ke tujuan langganan. Jika layanan tujuan mengembalikan kesalahan yang dapat dicoba ulang seperti pengecualian pembatasan atau pengecualian layanan yang dapat dicoba ulang (misalnya HTTP 5xx), CloudWatch Log terus mencoba lagi pengiriman hingga 24 jam. CloudWatch Log tidak mencoba mengirim ulang jika kesalahan adalah kesalahan yang tidak dapat dicoba ulang, seperti <code>AccessDeniedException</code> atau <code>ResourceNotFoundException</code>. Dimensi yang Valid: LogGroupName DestinationType, FilterName, PolicyLevel Statistik Valid: Sum Unit: Tidak ada
EMFDisabledErrors	Jumlah peristiwa log berformat EMF yang diabaikan karena kebijakan akun aktif jenis METRIC_EXTRACTION_POLICY yang cocok dengan grup log. Untuk informasi selengkapnya tentang kebijakan ekstraksi metrik, lihat PutAccountPolicy. Dimensi yang Benar: LogGroupName Statistik Valid: Sum Unit: Tidak ada

Metrik	Deskripsi
EMFParsingErrors	Jumlah kesalahan penguraian yang ditemui saat memproses log format metrik yang disematkan. Kesalahan seperti itu terjadi ketika log diidentifikasi sebagai format metrik tertanam tetapi tidak mengikuti format yang benar. Untuk informasi selengkapnya tentang format metrik yang disematkan, lihat Spesifikasi: Format metrik tertanam. Dimensi yang Benar: LogGroupName Statistik Valid: Sum Unit: Tidak ada
EMFValidationErrors	Jumlah kesalahan validasi yang ditemui saat memproses log format metrik tertanam. Kesalahan ini terjadi ketika definisi metrik dalam log format metrik tertanam tidak mematuhi format dan MetricDatum spesifikasi metrik yang disematkan. Untuk informasi tentang format metrik yang CloudWatch disematkan, lihat Spesifikasi: Format metrik tertanam. Untuk informasi tentang tipe data MetricDatum, lihat MetricDatum di Referensi Amazon CloudWatch API.  Note Kesalahan validasi tertentu dapat menyebabkan beberapa metrik dalam log EMF tidak dipublikasikan. Misalnya, semua metrik yang disetel dengan namespace yang tidak valid akan dihapus. Dimensi yang Benar: LogGroupName Statistik Valid: Sum Unit: Tidak ada

Metrik	Deskripsi
ErrorCount	Jumlah operasi API yang dilakukan di akun Anda yang mengakibatkan kesalahan. ErrorCount adalah metrik penggunaan layanan CloudWatch Log. Untuk informasi selengkapnya, lihat CloudWatch Metrik penggunaan layanan log. Dimensi yang Valid: Kelas, Sumber Daya, Layanan, Jenis Statistik Valid: Sum Unit: Tidak ada
ForwardedBytes	Volume log acara dalam byte terkompresi yang diteruskan ke tujuan langganan. Dimensi yang Valid: LogGroupName, DestinationType, FilterName Statistik Valid: Sum Unit: Byte
ForwardedLogEvents	Jumlah log acara yang diteruskan ke tujuan langganan. Dimensi yang Valid: LogGroupName DestinationType,, FilterName, PolicyLevel Statistik Valid: Sum Unit: Tidak ada

Metrik	Deskripsi
IncomingBytes	Volume peristiwa log dalam byte tidak terkompresi yang diunggah ke Log. CloudWatch Log. Ketika digunakan dengan dimensi LogGroupName , ini adalah volume log acara dalam byte tak terkompresi yang diunggah ke grup log. Dimensi yang valid: LogGroupName Statistik Valid: Sum Unit: Byte
IncomingLogEvents	Jumlah peristiwa log yang diunggah ke CloudWatch Log. Ketika digunakan dengan dimensi LogGroupName , ini adalah jumlah log acara yang diunggah ke grup log. Dimensi yang valid: LogGroupName Statistik Valid: Sum Unit: Tidak ada
LogEventsWithFindings	Jumlah peristiwa log yang cocok dengan string data yang Anda audit menggunakan fitur perlindungan data CloudWatch Log. Untuk informasi selengkapnya, lihat Membantu melindungi data log sensitif dengan masking. Dimensi valid: Tidak Ada Statistik Valid: Sum Unit: Tidak ada

Metrik	Deskripsi
ThrottleCount	Jumlah operasi API yang dilakukan di akun Anda yang dibatasi karena kuota penggunaan. ThrottleCount adalah metrik penggunaan layanan CloudWatch Log. Untuk informasi selengkapnya, lihat CloudWatch Metrik penggunaan layanan log. Dimensi yang Valid: Kelas, Sumber Daya, Layanan, Jenis Statistik Valid: Sum Unit: Tidak ada

Dimensi untuk metrik CloudWatch Log

Dimensi yang dapat Anda gunakan dengan sebagian besar metrik CloudWatch Log tercantum dalam tabel berikut.

Dimensi	Deskripsi
LogGroupName	Nama grup CloudWatch log Log untuk menampilkan metrik.
DestinationType	Tujuan berlangganan untuk data CloudWatch Log, yang dapat berupa AWS Lambda, Amazon Kinesis Data Streams, atau Amazon Data Firehose.
FilterName	Nama filter langganan yang meneruskan data dari grup log ke tujuan. Nama filter langganan secara otomatis dikonversi CloudWatch menjadi ASCII dan karakter apa pun yang tidak didukung diganti dengan tanda tanya (?).

Dimensi metrik filter langganan

Dimensi untuk metrik yang terkait dengan filter langganan tingkat akun tercantum dalam tabel berikut.

Dimensi	Deskripsi
<code>PolicyLevel</code>	Tingkat di mana kebijakan berlaku. Saat ini, satu-satunya nilai yang valid untuk dimensi ini adalah <code>AccountPolicy</code>
<code>DestinationType</code>	Tujuan berlangganan untuk data CloudWatch Log, yang dapat berupa AWS Lambda, Amazon Kinesis Data Streams, atau Amazon Data Firehose.
<code>FilterName</code>	Nama filter langganan yang meneruskan data dari grup log ke tujuan. Nama filter langganan secara otomatis dikonversi CloudWatch menjadi ASCII dan karakter apa pun yang tidak didukung diganti dengan tanda tanya (?).

Metrik dan dimensi transformator log

CloudWatch Log menerbitkan metrik transformator log berikut ke CloudWatch dalam namespace. `AWS/Logs`

Metrik	Deskripsi
<code>TransformationErrors</code>	Jumlah kesalahan yang ditemui saat mengubah peristiwa log dengan transformator yang ditentukan. Satuan: Tidak ada Statistik yang valid: Jumlah
<code>TransformedBytes</code>	Volume output dari peristiwa log yang diubah, dalam byte yang tidak terkompresi. Unit: Bit Statistik yang valid: Jumlah
<code>TransformedLogEvents</code>	Jumlah peristiwa log yang diubah. Satuan: Tidak ada

Metrik	Deskripsi
	Statistik yang valid: Jumlah

Dimensi berikut digunakan oleh metrik transformator.

Dimensi	Deskripsi
LogGroupName	Dimensi ini hanya digunakan untuk log-group-level transformer.
PolicyLevel	Dimensi ini hanya digunakan untuk transformator tingkat akun. Saat ini satu-satunya nilai yang valid untuk dimensi ini adalah AccountPolicy

Metrik dan dimensi sentralisasi

Untuk pemantauan terpusat di beberapa akun dan wilayah, Anda dapat menggunakan Sentralisasi CloudWatch Log untuk mengkonsolidasikan data log dan metrik di lokasi pusat. Untuk informasi selengkapnya, lihat [Sentralisasi log lintas wilayah lintas akun](#).

CloudWatch Log menerbitkan metrik sentralisasi berikut ke CloudWatch dalam namespace. AWS/Logs Metrik ini membantu Anda memantau replikasi data log dari akun sumber ke akun tujuan saat menggunakan aturan sentralisasi CloudWatch Log.

Note

CloudWatch mulai melaporkan metrik sentralisasi segera setelah Anda membuat aturan sentralisasi. Metrik diterbitkan dengan upaya terbaik dan hanya melacak peristiwa log baru yang tiba setelah aturan sentralisasi dibuat.

Metrik	Deskripsi	Diterbitkan di
IncomingCopiedBytes	Volume data log dalam byte terkompresi yang direplikasi ke akun tujuan. Metrik ini hanya berlaku untuk peristiwa log baru yang tiba setelah aturan sentralisasi dibuat.	Akun tujuan

Metrik	Deskripsi	Diterbitkan di
	Dimensi yang Valid: SourceRegion, SourceAccount Statistik Valid: Sum Unit: Byte	
IncomingCopiedLogEvents	Jumlah peristiwa log yang direplikasi ke akun tujuan. Metrik ini hanya berlaku untuk peristiwa log baru yang tiba setelah aturan sentralisasi dibuat. Dimensi yang Valid: SourceRegion, SourceAccount Statistik Valid: Sum Unit: Tidak ada	Akun tujuan
OutgoingCopiedBytes	Volume data log dalam byte tidak terkompresi yang dikirim dari akun sumber ke akun tujuan. Metrik ini hanya berlaku untuk peristiwa log baru yang disalin ke tujuan setelah aturan sentralisasi dibuat. Dimensi yang valid: DestinationRegion Statistik Valid: Sum Unit: Byte	Akun sumber

Metrik	Deskripsi	Diterbitkan di
OutgoingCopiedLogEvents	Jumlah peristiwa log yang dikirim dari akun sumber ke akun tujuan. Metrik ini hanya berlaku untuk peristiwa log baru yang disalin ke tujuan setelah aturan sentralisasi dibuat. Dimensi yang valid: DestinationRegion Statistik Valid: Sum Unit: Tidak ada	Akun sumber
CentralizationError	Jumlah kesalahan yang ditemui saat mereplikasi data log. Kesalahan dapat terjadi karena berbagai alasan seperti masalah izin kunci KMS, batas kuota grup log, atau ketidakcocokan tingkat log. Untuk mengidentifikasi grup log tertentu atau aliran log yang gagal replikasi dan alasan kegagalannya, pantau dimensinya. ErrorType Dimensi yang valid: ErrorType Statistik Valid: Sum Unit: Tidak ada	Akun tujuan
CentralizationThrottled	Berapa kali pemrosesan sentralisasi dibatasi. Pelambatan menghasilkan pemrosesan sentralisasi yang lebih lambat tetapi tidak mencegah data log direplikasi. Dimensi yang valid: DestinationRegion Statistik Valid: Sum Unit: Tidak ada	Akun sumber

Dimensi berikut digunakan oleh metrik sentralisasi.

Dimensi	Deskripsi
SourceRegion	AWS Wilayah tempat data log sumber berasal.
SourceAccount	ID AWS akun tempat data log sumber berasal.
DestinationRegion	AWS Wilayah tempat data log direplikasi.
ErrorType	Jenis kesalahan yang ditemui selama sentralisasi. Nilai yang mungkin termasuk: • LogGroupQuotaExceeded : Akun tujuan telah mencapai kuota grup log. • InvalidKMS : Kunci KMS hilang, dihapus, tidak diaktifkan, atau asimetris, bukan simetris. • AccessDenied : Akses ditolak saat mencoba mereplikasi data log. Hal ini dapat terjadi karena izin yang tidak mencukupi, seperti izin kunci KMS yang salah, izin IAM yang hilang, atau kebijakan sumber daya yang membatasi. • LogTierMismatch : Tingkat log dari grup log sumber dan tujuan tidak cocok. • InvalidLogStream : Parameter tidak valid yang ditemui saat membuat atau menulis ke aliran log, seperti ketika nama aliran log melebihi panjang maksimum. • InvalidLogGroup : Parameter tidak valid ditemui saat membuat atau mengonfigurasi grup log di akun tujuan. • DestinationEncryptionMismatch : Grup log tujuan sudah ada dengan konfigurasi enkripsi KMS yang berbeda dari yang ditentukan oleh aturan sentralisasi.

CloudWatch Metrik penggunaan layanan log

CloudWatch Log mengirimkan metrik untuk CloudWatch melacak penggunaan operasi API CloudWatch Log. Metrik ini sesuai dengan kuota AWS layanan. Dengan melacak metrik-metrik

tersebut dapat membantu Anda mengelola kuota secara proaktif. Untuk informasi selengkapnya, lihat [Integrasi Service Quotas dan Metrik Penggunaan](#).

Misalnya, Anda dapat melacak `ThrottleCount` metrik atau mengatur alarm pada metrik itu. Jika nilai metrik ini naik, Anda harus mempertimbangkan untuk meminta peningkatan kuota untuk operasi API yang terhambat. Untuk informasi selengkapnya tentang kuota layanan CloudWatch Log, lihat [CloudWatch Kuota log](#).

CloudWatch Log menerbitkan metrik penggunaan kuota layanan setiap menit di ruang nama dan ruang nama. `AWS/Usage AWS/Logs`

Tabel berikut mencantumkan metrik penggunaan layanan yang diterbitkan oleh CloudWatch Log. Metrik ini tidak memiliki unit tertentu. Statistik yang paling berguna untuk metrik ini adalah `SUM`, yang mewakili jumlah operasi total untuk periode 1 menit.

Masing-masing metrik ini diterbitkan dengan nilai untuk semua `Service`, `ClassType`, dan `Resource` dimensi. Mereka juga diterbitkan dengan satu dimensi yang disebut `Account Metrics`. Gunakan `Account Metrics` dimensi untuk melihat jumlah metrik untuk semua operasi API di akun Anda. Gunakan dimensi lain dan tentukan nama operasi API untuk `Resource` dimensi tersebut guna menemukan metrik untuk API tertentu.

Metrik-metrik

Metrik	Deskripsi
<code>CallCount</code>	Jumlah operasi tertentu yang dilakukan di akun Anda. <code>CallCount</code> diterbitkan di <code>AWS/Logs</code> ruang nama <code>AWS/Usage</code> dan ruang nama.
<code>ErrorCount</code>	Jumlah operasi API yang dilakukan di akun Anda yang mengakibatkan kesalahan. <code>ErrorCount</code> diterbitkan hanya dalam <code>AWS/Logs</code>.
<code>ThrottleCount</code>	Jumlah operasi API yang dilakukan di akun Anda yang dibatasi karena kuota penggunaan. <code>ThrottleCount</code> diterbitkan hanya dalam <code>AWS/Logs</code>.

Dimensi

Dimensi	Deskripsi
Account metrics	Gunakan dimensi ini untuk mendapatkan jumlah metrik di semua CloudWatch Log APIs. Jika Anda ingin melihat metrik untuk satu API tertentu, gunakan dimensi lain yang tercantum dalam tabel ini dan tentukan nama API sebagai nilaiResource.
Service	Nama AWS layanan yang berisi sumber daya. Untuk metrik penggunaan CloudWatch Log, nilai untuk dimensi ini adalahLogs.
Class	Kelas sumber daya yang dilacak. CloudWatch Metrik penggunaan API log menggunakan dimensi ini dengan nilai. None
Type	Jenis sumber daya yang sedang ditelusuri. Saat ini, ketika dimensi Service adalah Logs, satu-satunya nilai yang benar untuk Type adalah API.
Resource	Nama operasi API. Nilai yang valid mencakup semua nama operasi API yang tercantum dalam Tindakan . Sebagai contoh, PutLogEvents .

CloudWatch Kuota log

Anda dapat menggunakan tabel di bagian ini untuk meninjau kuota layanan default, juga disebut sebagai batas, untuk AWS akun di Amazon CloudWatch Logs. Sebagian besar kuota layanan, tetapi tidak semua, terdaftar di bawah namespace Amazon CloudWatch Logs di konsol Service Quotas.

Note

Jika Anda ingin meminta kenaikan kuota untuk salah satu kuota ini, lihat [prosedur di bagian ini](#).

Nama	Default	Dapat diseskan	Deskripsi
Tugas ekspor aktif	Setiap Wilayah yang didukung: 1	Tidak	Jumlah tugas ekspor aktif (berjalan atau tertunda) per akun
AssociateKmsKey batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum associate-kms-key panggilan per detik per account/per wilayah
AssociateSourceToBatas TableIntegration throttle S3 dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum TableIntegration panggilan Associate SourceTo S3 per detik per wilayah account/per
AssociateSourceToBatas TableIntegration throttle S3 dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum TableIntegration panggilan Associate SourceTo S3 per detik per account/per wilayah dalam ledakan

Nama	Default	Dapat disesuaikan	Deskripsi
Ukuran batch	Setiap Wilayah yang didukung: 1 Megabyte	Tidak	Ukuran batch maksimum dalam MB put-log-events permintaan
CancelExportTask batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum cancel-export-task panggilan per detik per account/per wilayah
CancellImportTask batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 1	Tidak	Jumlah maksimum CancellImportTask panggilan per detik per account/per wilayah
CancellImportTask batas throttle dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 1	Tidak	Jumlah maksimum CancellImportTask panggilan per detik per account/per wilayah dalam ledakan
CreateExportTask batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum create-export-task panggilan per detik per account/per wilayah
CreateImportTask batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 1	Tidak	Jumlah maksimum CreateImportTask panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
CreateImportTask batas throttle dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 1	Tidak	Jumlah maksimum CreateImportTask panggilan per detik per account/per wilayah dalam ledakan
CreateLogGroup batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 10 per detik	Ya	Jumlah maksimum create-log-group panggilan per detik per account/per wilayah
CreateLogStream batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 50 per detik	Ya	Jumlah maksimum create-log-stream panggilan per detik per account/per wilayah
Pengarsipan data	Setiap Wilayah yang didukung: 5 Gigabytes	Tidak	Ukuran maksimum dalam GB pengarsipan data gratis
DeleteDataProtectionPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum delete-data-protection-policy panggilan per detik per account/per wilayah
DeleteDestination batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan hapus-tujuan per detik per wilayah account/per
DeleteLogGroup batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 10 per detik	Ya	Jumlah maksimum delete-log-group panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
DeleteLogStream batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 15 per detik	Ya	Jumlah maksimum delete-log-stream panggilan per detik per account/per wilayah
DeleteMetricFilter batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum delete-metric-filter panggilan per detik per account/per wilayah
DeleteRetentionPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum delete-retention-policy panggilan per detik per account/per wilayah
DeleteSubscriptionFilter batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum delete-subscription-filter panggilan per detik per account/per wilayah
DeleteTransformer batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan hapus-transformator per detik per wilayah account/per
DescribeDestinations batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan deskripsi-tujuan per detik per wilayah account/per
DescribeExportTasks batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum describe-export-tasks panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
DescribeImportTaskBatches batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum DescribeImportTask Batches panggilan per detik per account/per wilayah
DescribeImportTaskBatches batas throttle dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum DescribeImportTask Batches panggilan per detik per account/per wilayah dalam ledakan
DescribeImportTasks batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum DescribeImportTasks panggilan per detik per account/per wilayah
DescribeImportTasks batas throttle dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum DescribeImportTasks panggilan per detik per account/per wilayah dalam ledakan
DescribeLogGroups batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 10 per detik	Ya	Jumlah maksimum describe-log-groups panggilan per detik per account/per wilayah
DescribeLogStreams batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 25 per detik	Ya	Jumlah maksimum describe-log-streams panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
DescribeMetricFilters batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum describe-metric-filters panggilan per detik per account/per wilayah
DescribeSubscriptionFilters batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum describe-subscription-filters panggilan per detik per account/per wilayah
DisassociateSourceFromBatas TableIntegration throttle S3 dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum TableIntegration panggilan DisassociateSourceFrom S3 per detik per wilayah account/per
DisassociateSourceFromBatas TableIntegration throttle S3 dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum TableIntegration panggilan DisassociateSourceFrom S3 per detik per account/per wilayah dalam ledakan
Ukuran peristiwa	Setiap Wilayah yang didukung: 1.024 Kilobyte	Tidak	Ukuran peristiwa log maksimum dalam KB

Nama	Default	Dapat disesuaikan	Deskripsi
FilterLogEvents batas throttle dalam transaksi per detik	us-east-1:25 per detik ap-northeast-3:5 per detik ap-southeast-3:5 per detik ca-west-1:5 per detik eu-central-1:5 per detik il-central-1:5 per detik Masing-masing Daerah yang didukung lainnya: 10 per detik	Tidak	Jumlah maksimum filter-log-events panggilan per detik per account/per wilayah
GetDataProtectionPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum get-data-protection-policy panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
GetLogEvents batas throttle dalam transaksi per detik	us-west-2:10 per detik ap-northeast-3:10 per detik ap-southeast-3:10 per detik ca-west-1:10 per detik eu-central-1:10 per detik eu-west-1:10 per detik eu-west-3:30 per detik il-central-1:10 per detik Masing-masing Daerah yang didukung lainnya: 25 per detik	Tidak	Jumlah maksimum get-log-events panggilan per detik per account/per wilayah
GetQueryResults batas throttle dalam transaksi per detik	me-central-1:6 Masing-masing Wilayah yang didukung lainnya: 10	Tidak	Jumlah maksimum get-query-results panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
GetTransformer batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan get-transformator per detik per wilayah account/per
ListSourcesForBatas TableIntegration throttle S3 dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum TableIntegration panggilan ListSourcesFor S3 per detik per wilayah account/per
ListSourcesForBatas TableIntegration throttle S3 dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum TableIntegration panggilan ListSourcesFor S3 per detik per account/per wilayah dalam ledakan
ListTagsForResource batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum list-tags-for-resource panggilan per detik per account/per wilayah
ListTagsLogGroup batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum list-tags-log-group panggilan per detik per account/per wilayah
Batas sesi bersamaan Live Tail	Setiap Wilayah yang didukung: 15	Ya	Jumlah maksimum sesi Live Tail aktif bersamaan per akun

Nama	Default	Dapat disesuaikan	Deskripsi
Grup log	Setiap Wilayah yang didukung: 1.000.000	Ya	Jumlah maksimum grup log yang dapat dimiliki akun
Grup log dipindai per sesi Live Tail	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum grup log yang dapat dipindai per sesi Live Tail
Filter metrik per grup log	Setiap Wilayah yang didukung: 100	Tidak	Jumlah filter metrik per grup log
PutBearerTokenAuthentication batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum put-bearer-token-authentication panggilan per detik per account/per wilayah
PutBearerTokenAuthentication batas throttle dalam transaksi per detik dalam ledakan	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum put-bearer-token-authentication panggilan per detik per account/per wilayah dalam ledakan
PutDataProtectionPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum put-data-protection-policy panggilan per detik per account/per wilayah
PutDestination batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan put-destination per detik per wilayah account/per

Nama	Default	Dapat disesuaikan	Deskripsi
PutDestinationPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum put-destination-policy panggilan per detik per account/per wilayah
PutLogEvents batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5.000 per detik	Ya	Jumlah maksimum put-log-events panggilan per detik per account/per wilayah
PutMetricFilter batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum put-metric-filter panggilan per detik per account/per wilayah
PutRetentionPolicy batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum put-retention-policy panggilan per detik per account/per wilayah
PutSubscriptionFilter batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum put-subscription-filter panggilan per detik per account/per wilayah
PutTransformer batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan put-transformator per detik per wilayah account/per
Kebijakan sumber daya	Setiap Wilayah yang didukung: 10	Tidak	Jumlah maksimum kebijakan sumber daya per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
StartLiveTail batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5	Tidak	Jumlah maksimum start-live-tail panggilan per detik per account/per wilayah. Batas ini berlaku secara independen untuk konsol dan API
StartQuery batas throttle dalam transaksi per detik	me-central-1:6 Masing-masing Wilayah yang didukung lainnya: 10	Tidak	Jumlah maksimum panggilan start-query per detik per wilayah account/per
Filter langganan per grup log	Setiap Wilayah yang didukung: 2	Tidak	Jumlah filter langganan per grup log
TagLogGroup batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum tag-log-group panggilan per detik per account/per wilayah
TagResource batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan tag-resource per detik per wilayah account/per
TestMetricFilter batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum test-metric-filter panggilan per detik per account/per wilayah

Nama	Default	Dapat disesuaikan	Deskripsi
TestTransformer batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan tes-transformator per detik per wilayah account/per
UntagLogGroup batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum untag-log-group panggilan per detik per account/per wilayah
UntagResource batas throttle dalam transaksi per detik	Setiap Wilayah yang didukung: 5 per detik	Tidak	Jumlah maksimum panggilan untag-resource per detik per wilayah account/per
log detektor anomali	Setiap Wilayah yang didukung: 500	Ya	Jumlah maksimum detektor anomali log aktif

Mengelola kuota layanan CloudWatch Log Anda

CloudWatch Log telah terintegrasi dengan Service Quotas, sebuah AWS layanan yang memungkinkan Anda untuk melihat dan mengelola kuota Anda dari lokasi pusat. Untuk informasi selengkapnya, lihat [Apa itu Service Quotas?](#) di Panduan Pengguna Service Quotas.

Service Quotas memudahkan untuk mencari nilai kuota layanan CloudWatch Log Anda.

Konsol Manajemen AWS

Untuk melihat kuota layanan CloudWatch Log menggunakan konsol

1. Buka konsol Service Quotas di <https://console.aws.amazon.com/servicequotas/>.
2. Di panel navigasi, pilih Layanan AWS .
3. Dari daftar AWS layanan, cari dan pilih Amazon CloudWatch Logs.

Dalam daftar service quotas, Anda dapat melihat nama service quotas, nilai terapan (jika tersedia), kuota default AWS, dan apakah nilai kuota dapat disesuaikan.

4. Untuk melihat informasi tambahan tentang service quotas, seperti deskripsi, pilih nama kuota.
5. (Opsional) Untuk meminta peningkatan kuota, pilih kuota yang ingin Anda tingkatkan, pilih Request quota increase (Meminta kenaikan kuota), masukkan atau pilih informasi yang diperlukan, dan pilih Request (Permintaan).

Untuk bekerja lebih lanjut dengan kuota layanan menggunakan konsol lihat [Panduan Pengguna Service Quotas](#). Untuk meminta kenaikan kuota, lihat [Meminta kenaikan kuota](#) dalam Panduan Pengguna Service Quotas.

AWS CLI

Untuk melihat kuota layanan CloudWatch Log menggunakan AWS CLI

Jalankan perintah berikut untuk melihat kuota CloudWatch Log default.

```
aws service-quotas list-aws-default-service-quotas \
  --query 'Quotas[*]'.
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
  --service-code logs \
  --output table
```

Untuk bekerja lebih banyak dengan kuota layanan menggunakan AWS CLI, lihat Referensi Perintah [Service AWS CLI Quotas](#). Untuk meminta kenaikan kuota, lihat perintah [request-service-quota-increase](#) di [Referensi Perintah AWS CLI](#).

Riwayat dokumen

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan Pengguna CloudWatch Log, dimulai pada Juni 2018. Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

Perubahan	Deskripsi	Tanggal
Kueri tersimpan dengan parameter untuk Wawasan CloudWatch Log	Sekarang Anda dapat membuat templat kueri yang dapat digunakan kembali dengan parameter bernama di Wawasan CloudWatch Log. Tentukan satu templat dan berikan nilai yang berbeda pada waktu berjalan alih-alih mempertahankan beberapa kueri tersimpan yang hampir identik. Untuk informasi selengkapnya, lihat Menggunakan kueri tersimpan dengan parameter .	Maret 26, 2026
Kebijakan CloudWatch terkelola Log Baru CloudWatch Logs APIKey Akses	CloudWatch Logs telah merilis kebijakan terkelola baru CloudWatchLogsAPIKeyAccess yang memungkinkan otentikasi kunci CloudWatch Logs API dan konsumsi log terenkripsi. Untuk selengkapnya, lihat CloudWatch Memutakhirkan log ke kebijakan AWS terkelola .	Februari 17, 2026

[CloudWatchLogsReadOnlyAccess Kebijakan yang diperbarui](#)

Izin untuk tindakan administrasi observabilitas ditambahkan untuk memungkinkan akses hanya-baca [CloudWatchLogsReadOnlyAccess](#) ke pipeline telemetri dan integrasi tabel S3.

Desember 2, 2025

[CloudWatchLogsFullAccess Kebijakan yang diperbarui](#)

Izin untuk tindakan administrasi observabilitas ditambahkan untuk memungkinkan akses hanya-baca [CloudWatchLogsFullAccess](#) ke pipeline telemetri dan integrasi tabel S3.

Desember 2, 2025

[Fitur manajemen log baru di CloudWatch Log](#)

CloudWatch mengkonsolidasikan manajemen log ke dalam satu layanan dengan kemampuan tata kelola bawaan tanpa menyimpan dan memelihara banyak salinan data yang sama di berbagai alat dan penyimpanan data. Untuk informasi selengkapnya, lihat [Manajemen log](#).

Desember 2, 2025

[Fitur konsumsi dan normalisasi data baru di Log CloudWatch](#)

CloudWatch secara otomatis mengumpulkan log AWS vended di seluruh akun dan AWS Wilayah melalui integrasi dengan AWS Organizations dan konektor pra-bangun untuk sumber pihak ketiga. Untuk informasi selengkapnya, lihat [Mengubah log selama konsumsi](#).

Desember 2, 2025

[Analisis dan kueri baru berdasarkan fitur sumber data di Wawasan CloudWatch Log](#)

CloudWatch Wawasan memperkenalkan aspek untuk kueri. Aspek berguna untuk menganalisis log karena memungkinkan Anda memfilter dan mengelompokkan data secara interaktif tanpa menjalankan kueri. Aspek adalah bidang di log Anda (seperti ServiceName atau StatusCode) yang memungkinkan penyaringan, agregasi, dan analisis di seluruh grup log. Untuk informasi selengkapnya, lihat [Menggunakan aspek untuk mengelompokkan dan menjelajahi log](#).

Desember 2, 2025

[Integrasi Tabel S3 Baru](#)

[Integrasi Tabel S3 CloudWatch](#) memungkinkan Anda [mengakses data log yang tertelan CloudWatch menggunakan mesin analitik seperti Amazon Athena, Amazon Redshift, dan alat pihak ketiga yang mendukung koneksi ke toko yang kompatibel dengan Apache Iceberg](#). Untuk informasi selengkapnya, lihat [Akses log dengan Integrasi Tabel S3](#).

Desember 2, 2025

[CloudWatch Log menambahkan dukungan untuk sentralisasi log lintas wilayah lintas akun](#)

CloudWatch Log menambahkan dukungan untuk sentralisasi log Lintas akun Lintas wilayah yang memungkinkan CloudWatch Log mengumpulkan data log dari akun AWS Organizations anggota ke akun pusat untuk dianalisis. Untuk informasi selengkapnya, lihat [Sentralisasi log lintas wilayah lintas akun](#).

September 17, 2025

[CloudWatch Log deteksi anomali diperbarui contoh kebijakan](#)

CloudWatch Log memperbarui kebijakan contoh KMS untuk mengurangi cakupannya dan meningkatkan keamanan dengan menambahkan kondisi untuk `aws:SourceAccount` dan `aws:SourceArn`. Untuk informasi selengkapnya, lihat [Mengkripsi detektor anomali dan hasilnya](#) dengan KMS AWS

Juli 14, 2025

[CloudWatch Logs Insights menambahkan dukungan untuk log Amazon VPC Route Server](#)

CloudWatch Logs Insights menambahkan dukungan untuk log Amazon VPC Route Server. Untuk informasi selengkapnya, lihat [Mengaktifkan pencatatan dari AWS layanan](#). Sumber log baru untuk Amazon VPC Route Server, `EVENT_LOGS` didokumentasikan di [PutDeliverySource](#)

Juni 12, 2025

[CloudWatch Logs Insights
menambahkan dukungan
untuk AWS PCS log](#)

CloudWatch Logs Insights menambahkan dukungan untuk AWS PCS log. Untuk informasi selengkapnya, lihat [Mengaktifkan pencatatan dari AWS layanan](#). Sumber log baru untuk AWS PCS, PCS_SCHEDULER_LOGS dan PCS_JOBCOMP_LOGS didokumentasikan di [PutDelive rySource](#)

Juni 12, 2025

[CloudWatch Logs Insights
menambahkan dukungan
untuk Resolusi Entitas AWS
log](#)

CloudWatch Logs Insights menambahkan dukungan untuk Resolusi Entitas AWS log. Untuk informasi selengkapnya, lihat [Mengaktif kan pencatatan dari AWS layanan](#). Sumber log baru untuk Resolusi Entitas AWS, WORKFLOW_LOGS didokumen tasikan di [PutDeliverySource](#)

22 Mei 2025

[CloudWatch Log pembaruan kebijakan terkelola untuk mendukung ringkasan bahasa alami](#)

Izin untuk `cloudwatch:GenerateQueryResultsSummary` ditambahkan ke `CloudWatchLogsFullAccess` dan `CloudWatchLogsReadOnlyAccess` memungkinkan pembuatan ringkasan bahasa alami dari hasil kueri. Untuk melihat isi kebijakan, lihat [CloudWatchLogsFullAccess](#) dan [CloudWatchLogsReadOnlyAccess](#) di Panduan Referensi Kebijakan AWS Terkelola.

Mei 20, 2025

[CloudWatch Logs Insights menambahkan dukungan untuk menghasilkan ringkasan bahasa alami dari hasil kueri CloudWatch Logs Insights](#)

Menambahkan dukungan untuk ringkasan bahasa alami di Wawasan CloudWatch Log. Fitur ini menghasilkan ringkasan hasil kueri yang dapat dibaca manusia, saat ini tersedia di AS Timur (Virginia N.). Untuk informasi selengkapnya, lihat [Menghasilkan ringkasan bahasa alami dari hasil kueri Wawasan CloudWatch Log](#).

Mei 20, 2025

[CloudWatchOpenSearchDashboardAccessdanCloudWatchOpenSearchDashboardsFullAccess](#) merupakan kebijakan IAM baru

CloudWatch Log menambahkan dua kebijakan IAM baru, CloudWatchOpenSearchDashboardsFullAccessdanCloudWatchOpenSearchDashboardAccess. CloudWatchOpenSearchDashboardsFullAccessmemberikan izin untuk membuat dan mengelola integrasi dengan OpenSearch Layanan. CloudWatchOpenSearchDashboardAccessmemberikan akses untuk melihat dasbor log vendid yang dibuat dalam integrasi ini. Untuk informasi selengkapnya, lihat [Dasbor log penjual yang didukung oleh Layanan Amazon OpenSearch](#).

Desember 1, 2024

[CloudWatchLogsFullAccess](#) kebijakan diperbarui

CloudWatch Log menambahkan izin untuk Amazon OpenSearch Service dan IAM ke CloudWatchLogsFullAccesskebijakan, untuk mengaktifkan integrasi CloudWatch Log dengan OpenSearch Layanan untuk beberapa fitur.

Desember 1, 2024

[CloudWatch Logs Insights
menambahkan tipe struktur
baru ke kueri syntax](#)

CloudWatch Logs Insights menambahkan `unnest` perintah dan dua fungsi JSON, yang memungkinkan Anda mengoperasikan string JSON sebagai peta dan daftar. Untuk informasi selengkapnya, lihat [Jenis struktur](#).

November 21, 2024

[CloudWatch Log mendukung
transformasi log selama
konsumsi log](#)

Anda dapat membuat transformator log yang dapat memodifikasi peristiwa log pada saat konsumsi, membantu Anda menormalkan log Anda dalam format yang berbeda dan dari sumber yang berbeda ke dalam format yang konsisten dan kaya konteks. Untuk informasi selengkapnya, lihat [Mengubah log selama konsumsi](#).

November 20, 2024

[CloudWatch Logs Insights
menambahkan pengindeksan
bidang](#)

CloudWatch Logs Insights telah menambahkan dukungan untuk pengindeksan bidang log. Saat Anda kemudian menggunakan indeks bidang dalam kueri Wawasan CloudWatch Log, kueri akan mencoba melewati proses peristiwa log yang diketahui tidak menyertakan bidang yang diindeks. Untuk informasi selengkapnya, lihat [Membuat indeks bidang untuk meningkatkan kinerja kueri dan mengurangi volume pemindaian](#).

November 20, 2024

[CloudWatch Dukungan
Wawasan Log untuk
pembuatan kueri bahasa alami
umumnya tersedia](#)

CloudWatch Logs Insights mendukung bahasa alami untuk menghasilkan dan memperbarui kueri. Untuk informasi selengkapnya, lihat [Menggunakan bahasa alami untuk membuat dan memperbarui kueri Wawasan CloudWatch Log](#).

Juni 20, 2024

[CloudWatchLogsRead
OnlyAccesskebijakan
diperbarui](#)

CloudWatch Log menambahkan `cloudwatch:GenerateQuery` izin ke `CloudWatchLogsReadOnlyAccess`, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri [Wawasan CloudWatch Log](#) dari prompt bahasa alami.

26 November 2023

[CloudWatchLogsFull Accesskebijakan diperbarui](#)

CloudWatch Log menambahkan `cloudwatch:GenerateQuery` izin ke `CloudWatchLogsFullAccess`, sehingga pengguna dengan kebijakan ini dapat menghasilkan string kueri [Wawasan CloudWatch Log](#) dari prompt bahasa alami.

26 November 2023

[CloudWatch Log menambahkan analisis pola log](#)

CloudWatch Log sekarang memindai pola dalam peristiwa log setiap kali Anda melakukan kueri Wawasan CloudWatch Log. Untuk informasi lebih lanjut, lihat [Analisis pola](#).

26 November 2023

[CloudWatch Log menambahkan deteksi anomali log](#)

Anda dapat membuat detektor anomali log untuk grup log. Detektor anomali memindai peristiwa log yang dicerna ke dalam grup log dan menemukan anomali dalam data log. Untuk informasi lebih lanjut, lihat [Deteksi anomali log](#).

26 November 2023

[CloudWatch Log menambahkan fitur bandingkan](#)

Anda sekarang dapat menggunakan Wawasan CloudWatch Log untuk membandingkan perubahan dalam peristiwa log Anda dari waktu ke waktu. Untuk informasi selengkapnya, lihat [Bandingkan \(diff\) dengan rentang waktu sebelumnya](#).

26 November 2023

[CloudWatch Log menambahkan kelas log baru](#)

CloudWatch Log mendukung dua kelas grup log sehingga Anda dapat memiliki opsi hemat biaya untuk log yang jarang Anda akses, dan Anda juga memiliki opsi fitur lengkap untuk log yang memerlukan pemantauan waktu nyata atau fitur lainnya. Untuk informasi selengkapnya, lihat [Kelas log](#).

26 November 2023

[CloudWatch Logs Insights mendukung pembuatan kueri bahasa alami](#)

CloudWatch Logs Insights mendukung bahasa alami untuk menghasilkan dan memperbarui kueri. Untuk informasi selengkapnya, lihat [Menggunakan bahasa alami untuk membuat dan memperbarui kueri Wawasan CloudWatch Log](#).

26 November 2023

[CloudWatch Log menambahkan dukungan sintaks pola filter ekspresi reguler untuk Live Tail](#)

Sekarang Anda dapat menyesuaikan operasi pencarian dan pencocokan lebih lanjut untuk memenuhi kebutuhan Anda dengan ekspresi reguler yang fleksibel dalam pola filter Live Tail. Untuk informasi selengkapnya, lihat [Memfilter sintaks pola](#) di Panduan Pengguna CloudWatch Log Amazon.

13 November 2023

[CloudWatch Log menambahkan dukungan sintaks pola filter ekspresi reguler untuk filter metrik, filter langganan, dan peristiwa log filter](#)

Anda sekarang dapat menyesuaikan operasi pencarian dan pencocokan lebih lanjut untuk memenuhi kebutuhan Anda dengan ekspresi reguler yang fleksibel dalam pola filter. Untuk informasi selengkapnya, lihat [Memfilter sintaks pola](#) di Panduan Pengguna CloudWatch Log Amazon.

5 September 2023

[CloudWatch Log Insights menambahkan perintah pola](#)

Sekarang Anda dapat menggunakan pola dalam kueri Wawasan CloudWatch Log untuk secara otomatis mengelompokkan data log Anda ke dalam pola. Pola adalah struktur teks bersama yang berulang di antara bidang log Anda. Untuk informasi selengkapnya, lihat [pola](#) di Panduan Pengguna CloudWatch Log Amazon.

Juli 17, 2023

[CloudWatch Logs Insights menambahkan perintah dedup](#)

Sekarang Anda dapat menggunakan dedup dalam kueri Wawasan CloudWatch Log untuk menghapus hasil duplikat berdasarkan nilai tertentu di bidang yang Anda tentukan. Untuk informasi selengkapnya, lihat [dedup](#) di Panduan Pengguna Amazon CloudWatch Logs.

20 Juni 2023

[Kebijakan perlindungan data tingkat akun](#)

Anda sekarang dapat menetapkan kebijakan perlindungan data di tingkat akun. Kebijakan tingkat akun ini dapat mengaudit dan menutupi informasi sensitif dalam peristiwa log di semua grup log di akun. Untuk informasi selengkapnya, lihat [Membantu melindungi data log sensitif dengan masking](#) di Panduan Pengguna Amazon CloudWatch Logs.

8 Juni 2023

[Fitur Live Tail ditambahkan](#)

CloudWatch Log menambahkan kemampuan Live Tail, sehingga Anda dapat memindai log saat tertelan untuk membantu pemecahan masalah. Anda dapat secara opsional memfilter aliran peristiwa log yang ditampilkan berdasarkan istilah yang ditentukan, dan juga menyorot peristiwa log yang memiliki istilah tertentu. Untuk informasi selengkapnya, silakan lihat [Menggunakan live tail untuk melihat log mendekati waktu nyata](#).

6 Juni 2023

[CloudWatchLogsReadOnlyAccesskebijakan diperbarui](#)

CloudWatch Log menambahkan izin ke CloudWatchLogsReadOnlyAccess. Izin `logs:StartLiveTail` dan `logs:StopLiveTail` izin ditambahkan sehingga pengguna dengan kebijakan ini dapat menggunakan konsol untuk memulai dan menghentikan sesi ekor langsung CloudWatch Log. Untuk informasi selengkapnya, silakan lihat [Menggunakan live tail untuk melihat log mendekati waktu nyata](#).

6 Juni 2023

[CloudWatch Log Insights dirilis](#)

Anda dapat menggunakan Wawasan CloudWatch Log untuk mencari dan menganalisis data log secara interaktif. Untuk informasi selengkapnya lihat [Menganalisis Data CloudWatch Log dengan Wawasan Log](#) di Panduan Pengguna CloudWatch Log Amazon

27 November 2018

[Dukungan untuk titik akhir VPC Amazon VPC](#)

Anda sekarang dapat membuat koneksi pribadi antara VPC dan CloudWatch Log Anda. Untuk informasi selengkapnya, lihat [Menggunakan CloudWatch Log dengan Titik Akhir VPC Antarmuka di Panduan Pengguna Amazon CloudWatch Logs](#).

28 Juni 2018

Tabel berikut menjelaskan perubahan penting pada Panduan Pengguna Amazon CloudWatch Logs.

Ubah	Deskripsi	Tanggal rilis
Titik akhir VPC antarmuka	Di beberapa Wilayah, Anda dapat menggunakan titik akhir VPC antarmuka untuk menjaga lalu lintas antara VPC Amazon dan Log Anda CloudWatch agar tidak meninggalkan jaringan Amazon. Untuk mengetahui informasi selengkapnya, lihat Menggunakan CloudWatch Log dengan titik akhir VPC antarmuka .	Selasa, 07 Maret 2018
Log kueri DNS Route 53	Anda dapat menggunakan CloudWatch Log untuk menyimpan log tentang kueri DNS yang diterima oleh Route 53. Untuk informasi selengkapnya, lihat Apa itu Amazon CloudWatch Logs? atau Mencatat Log Kueri DNS dalam Panduan Developer Amazon Route 53.	7 September 2017
Menandai grup log	Anda dapat menggunakan tanda untuk mengategorikan grup log Anda. Untuk informasi selengkapnya, lihat Tandai grup log di Amazon CloudWatch Logs .	13 Desember 2016

Ubah	Deskripsi	Tanggal rilis
Penyempurnaan konsol	Anda dapat menavigasi dari grafik metrik ke grup log terkait. Untuk informasi selengkapnya, lihat Pivot dari metrik ke log .	Selasa, 07 Nopember 2016
Penyempurnaan kegunaan konsol	Meningkatkan pengalaman agar lebih mudah mencari, memfilter, dan memecahkan masalah. Misalnya, Anda sekarang dapat memfilter data log Anda berdasarkan rentang tanggal dan waktu. Untuk informasi selengkapnya, lihat Lihat data log yang dikirim ke CloudWatch Log .	Selasa, 29 Agustus 2016
Menambahkan AWS CloudTrail dukungan untuk Amazon CloudWatch Log dan metrik CloudWatch Log baru	Ditambahkan AWS CloudTrail dukungan untuk CloudWatch Log. Untuk informasi selengkapnya, lihat Logging CloudWatch Logs API dan operasi konsol di AWS CloudTrail .	10 Maret 2016
Menambahk an dukungan untuk ekspor CloudWatch Log ke Amazon S3	Menambahkan dukungan untuk mengekspor data CloudWatch Log ke Amazon S3. Untuk informasi selengkapnya, lihat Mengekspor data log ke Amazon S3 .	Selasa, 07 Desember 2015
Menambahkan dukungan untuk peristiwa yang AWS CloudTrail dicatat di Amazon CloudWatch Logs	Anda dapat membuat alarm CloudWatch dan menerima notifikasi aktivitas API tertentu seperti yang ditangkap oleh CloudTrail dan menggunakan notifikasi untuk melakukan pemecahan masalah.	10 November 2014

Ubah	Deskripsi	Tanggal rilis
Ditambahk an dukungan untuk Amazon CloudWatch Logs	Anda dapat menggunakan Amazon CloudWate h Logs untuk memantau, menyimpan, dan mengakses sistem, aplikasi, dan file log kustom Anda dari instans Amazon Elastic Compute Cloud (Amazon EC2) atau sumber lain. Anda kemudian dapat mengambil data log terkait dari CloudWate h Log menggunakan CloudWatch konsol Amazon, perintah CloudWatch Log di AWS CLI, atau CloudWatch Logs SDK. Untuk informasi selengkap nya, lihat Apa itu Amazon CloudWatch Logs? .	10 Juli 2014

AWS Glosarium

Untuk AWS terminologi terbaru, lihat [AWS glosarium di Referensi](#).Glosarium AWS

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