Manual do usuário

CloudWatch Registros da Amazon



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

CloudWatch Registros da Amazon: Manual do usuário

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

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é Amazon CloudWatch Logs? ... 1
Recursos ... 1
AWS Serviços relacionados ... 3
Preços ... 4
Conceitos ... 5
Faturamento e custos ... 6
Classes de logs ... 7
Recursos compatíveis ... 7
Otimize os custos de armazenamento com o Amazon CloudWatch Logs Intelligent Tiering ... 11
CloudWatch Registra os níveis de acesso do Intelligent Tiering ... 11
Ações que constituem acesso ... 12
Habilitando a hierarquização inteligente de CloudWatch registros ... 13
Permissões obrigatórias ... 13
Habilitar o usando o console do ... 13
Habilitar o uso do AWS CLI ... 14
Visualizando sua configuração atual ... 14
Monitoramento do armazenamento por nível ... 14
Preços do CloudWatch Logs Intelligent Tiering ... 15
Introdução ... 16
Pré-requisitos ... 16
Inscreva-se em um Conta da AWS ... 16
Configurar a interface de linha de comando ... 17
Usando o CloudWatch agente unificado ... 17
Usando o CloudWatch agente anterior ... 17
CloudWatch Pré-requisitos do agente de registros ... 18
Início rápido: Instalar o agente em uma instância do EC2 do Linux em execução ... 19
Início rápido: instalar o agente em uma instância do EC2 do Linux na inicialização ... 26
Início rápido: use CloudWatch registros com instâncias do Windows Server 2016 ... 30
Início rápido: use CloudWatch registros com instâncias do Windows Server 2012 e do Windows Server 2008 ... 42
Relate o status do agente do CloudWatch Logs ... 52
Inicie o agente CloudWatch Logs ... 53
Pare o agente CloudWatch de registros ... 53
CloudWatch Referência do agente de registros ... 54

Início rápido com CloudFormation 66
Ingestão de registros por meio de endpoints HTTP 68
Comportamento comum 68
Autenticação do token do portador 69
Opção 1: início rápido usando o AWS console 70
Opção 2: configuração manual 72
Controle as permissões para gerar e usar CloudWatch as chaves da API Logs 75
Chaves de API rotativas 77
Respondendo a uma chave de API comprometida 78
Práticas recomendadas de segurança para chaves de API 79
Registrando o uso da chave de API com CloudTrail 80
Endpoint OTLP 81
Formato de solicitação 81
Exemplo de solicitação 82
Respostas 83
Comportamentos específicos do OTLP 83
Endpoint HLC 83
Modos de entrada 84
Campo do evento (obrigatório) 84
Campo de hora (opcional) 84
Content-Type 85
Tipos de valores JSON aceitos 85
Formato do endpoint 85
Formato de solicitação 86
Exemplo de solicitação 87
Práticas recomendadas 88
Limitações 88
Endpoint NDJSON 89
Formato de solicitação 89
Tipos de valores JSON aceitos 89
Campo de carimbo de data/hora 91
Linhas inválidas 91
Exemplo de solicitação 91
Respostas 91
Práticas recomendadas 92
Limitações 93

Endpoint JSON estruturado 93
Formato de solicitação 93
Tipos de valores JSON aceitos 94
Campo de carimbo de data/hora 94
Exemplo de solicitação 95
Respostas 95
Práticas recomendadas 96
Limitações 96
Comparação dos endpoints de ingestão de HTTP 97
Escolhendo um endpoint 98
Ingestão de Syslog 99
Protocolos e portas compatíveis 99
Formatos de syslog suportados 100
Campos extraídos do RFC 5424 100
Campos extraídos do RFC 3164 101
Campos FTD/ASA extraídos da Cisco 102
Entrega de mensagens 102
Cotas e limites 103
Configurar 104
Pré-requisitos 104
Etapa 1: criar ou identificar uma VPC 105
Etapa 2: criar um grupo de segurança 107
Etapa 3: Criar o VPC endpoint. 108
Etapa 4: criar um grupo de registros 110
Etapa 5: adicionar uma política de recursos 110
Etapa 6: Criar a configuração do syslog 112
Etapa 7: Verificar a entrega e a extração do campo 113
Configuração do dispositivo 114
rsyslog 114
syslog-ng 116
Validação rápida 117
Dispositivos de rede 118
Confiança do certificado TLS 119
Monitoramento 119
Métricas do Syslog 119
Motivos da queda 120

Motivos de rejeição de conexão 120
Alarmes recomendados 121
Visualizando métricas de syslog no painel automático 122
Exemplos de consultas 122
Monitoramento de segurança e acesso 122
Visibilidade operacional 123
Consultas de firewall e dispositivos de rede 124
Políticas de VPC endpoint 125
Chaves de condição compatíveis 125
Restringir a um grupo de registros específico 126
Negar todo o tráfego 127
Redefinir para a política padrão 127
Propagação de políticas 127
Solução de problemas 128
Mensagens que não aparecem no grupo de registros 128
Falhas do handshake TLS 129
Mensagens UDP perdidas 130
Conexão recusada 130
Mensagens interrompidas devido à limitação de taxa 130
Trabalhando com AWS SDKs 132
Gerenciamento de logs 134
Descoberta e gerenciamento de fontes de dados 134
O que são fontes de dados de CloudWatch registros? 135
Como começar 135
Campos do sistema 137
Acessar fontes de dados 137
Relacionamento com grupos de registros 138
Recursos habilitados por fontes de dados 138
Fonte de dados compatíveis 139
Compatível Serviços da AWS para fontes de dados 139
Fontes de terceiros compatíveis para fontes de dados 144
Fontes personalizadas 162
Análise de dados de registro com o CloudWatch Log Analytics 163
O que você pode fazer com o Log Analytics 163
Comece a usar o Log Analytics 163
Análise de dados de registro com o CloudWatch Logs Insights 165

Linguagens de consulta compatíveis 168
CloudWatch Linguagem de consulta do Logs Insights (Logs Insights QL) 169
OpenSearch Linguagem de processamento canalizado (PPL) 264
OpenSearch Linguagem de consulta estruturada (SQL) 277
Use linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights 290
Consultas de exemplo 291
Optar por não usar seus dados para melhorar o serviço 293
Logs compatíveis e campos descobertos 294
Campos em logs JSON 297
Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento 299
Campos indexados automaticamente 303
Sintaxe e cotas do índice de campo 304
Crie uma política de índice de campo ao nível da conta 309
Crie uma política de índice de campo em nível de grupo de logs 310
Opções de seleção de grupos de logs ao criar uma consulta 311
Efeitos da exclusão de uma política de índice de campo 312
Use facetas para agrupar e explorar registros 313
Para executar uma consulta baseada em facetas 314
Para salvar uma consulta baseada em facetas 314
Para criar uma faceta no nível da conta 314
Gerenciamento de facetas usando APIs 315
Visualizando registros ao redor no CloudWatch Logs Insights 315
Como funcionam os troncos circundantes 316
Veja os registros ao redor 316
Pesquise nos registros ao redor 317
Análise de padrões 318
Introdução à análise de padrões 319
Detalhes sobre o comando de padrão 322
Salvar e executar consultas novamente 322
Usando consultas salvas com parâmetros 328
Adicionar consulta ao painel ou exportar os resultados da consulta 331
Exibir as consultas em execução ou o histórico de consultas 331
Criptografe os resultados da consulta com AWS Key Management Service 332
Limites 333
Etapa 1: criar um AWS KMS key 333

Etapa 2: definir permissões na chave do KMS 334
Etapa 3: associar uma chave do KMS aos resultados da consulta 336
Etapa 4: desassociar uma chave dos resultados da consulta na conta 336
Gere um resumo em linguagem natural a partir dos resultados da consulta do CloudWatch Logs Insights 336
Como funciona 336
Disponibilidade regional e processamento de dados 337
Introdução 338
Permissões 338
Privacidade de dados 338
Automatizando a análise de registros com consultas agendadas 339
Entendendo os conceitos de consultas agendadas 339
Separação de funções do IAM 340
Cross-region e uso entre contas 342
Expressões de cronograma e tratamento de fuso horário 344
Escolhendo um idioma de consulta 344
Seleção de destinos e casos de uso 345
Formato e estrutura do resultado da consulta 347
Compreendendo os registros digitalizados, os registros combinados e a contagem de resultados 350
Referência de expressão de cronograma 352
Práticas recomendadas 356
Começando com consultas agendadas 357
Criação de uma consulta agendada 358
Visualizando e gerenciando consultas agendadas 362
Visualizando o histórico de execução de consultas agendadas 364
Atualizando uma consulta agendada 366
Configurando destinos do S3 para consultas agendadas 368
Entrega de resultados para um bucket do Amazon S3 na mesma conta 368
Entrega de resultados para um bucket do Amazon S3 em outra conta 369
Criptografando resultados com um cliente gerenciado AWS KMS key 370
Configurando destinos da tabela de pesquisa para consultas agendadas 373
Solução de problemas de consultas agendadas 374
A execução da consulta falha com erros de permissão 374
O tempo limite da consulta 375
O processamento de destino falha 375

Erros de consulta inválidos 376
Erros de simultaneidade de consultas 376
Detecção de anomalias de log 378
Gravidade e prioridade de anomalias e padrões 379
Tempo de visibilidade das anomalias 380
Supressão de uma anomalia 380
Perguntas frequentes 380
Usando a detecção de anomalias no CloudWatch Logs Insights 381
Habilitar a detecção de anomalias em um grupo de logs 382
Visualizar as anomalias que foram encontradas 383
Criar alarmes em detectores de anomalias de log 386
Métricas publicadas por detectores de anomalias de log 389
Criptografe um detector de anomalias e seus resultados com AWS KMS 389
Limites 390
Solução de problemas com o CloudWatch Logs Live Tail 395
Inicie uma sessão do Live Tail usando o AWS CLI 396
somente impressão 396
interativo 396
Iniciar uma sessão do Live Tail no console 398
Cross-account Centralização de registros entre regiões 402
Conceitos da centralização de dados 402
Configurando a centralização de logs 404
Pré-requisitos 404
Personalizando nomes de grupos de registros de destino 407
Criação de uma regra de centralização 408
Modificando uma regra de centralização 412
Visualizar uma regra de centralização 412
Excluindo uma regra de centralização 413
Monitoramento e solução de problemas de centralização 413
Status de integridade da regra de centralização 414
Monitoramento de chamadas de API de centralização com AWS CloudTrail 414
Recomendações de monitoramento 415
Trabalhar com grupos de logs e fluxos de logs 418
Criar um grupo de logs 418
Enviar logs a um grupo de logs 419
Visualizar dados de log 419

Pesquisar dados de log usando padrões de filtro ... 420
Pesquisar entradas de log usando o console ... 420
Pesquisar entradas de registro usando o AWS CLI ... 421
Passar de métricas para logs ... 421
Solução de problemas ... 422
Alterar a retenção do log de dados ... 423
Proteja grupos de registros contra exclusão ... 424
Proteja grupos de registros contra exclusão ... 424
Ativando a proteção contra exclusão ... 424
Etiquetar grupos de logs ... 425
Conceitos Básicos de Tags ... 425
Monitorar custos usando a marcação ... 426
Restrições de tags ... 426
Marcando grupos de registros usando o AWS CLI ... 427
Como marcar grupos de registros usando a API CloudWatch Logs ... 428
Criptografe dados de registro usando AWS KMS ... 428
Limites ... 429
Etapa 1: criar uma AWS KMS chave ... 333
Etapa 2: definir permissões na chave do KMS ... 431
Etapa 3: definir permissões para o diretor do IAM chamador ... 433
Etapa 4: associar uma chave KMS a um grupo de registros ... 394
Etapa 5: Desassociar a chave de um grupo de registros ... 394
Chaves do KMS e contexto de criptografia ... 437
Ajude a proteger dados de log confidenciais com mascaramento ... 440
Noções básicas sobre políticas de proteção de dados ... 443
Permissões de IAM necessárias para criar ou trabalhar com uma política de proteção de dados ... 446
Criar uma política de proteção de dados para toda a conta ... 451
Criar uma política de proteção de dados para um único grupo de logs ... 455
Exibir dados não mascarados ... 458
Relatórios de descobertas de auditoria ... 459
Tipos de dados que você pode proteger ... 460
Transformar logs durante a ingestão ... 505
Crie e gerencie transformadores de log ... 507
Crie uma política de transformador no nível da conta ... 508
Editar ou excluir uma política de transformador em nível de conta ... 509

Crie um transformador de log-group-level toras do zero 510
Crie um log-group-level transformador copiando um existente 511
Editar um log-group-level transformador 512
Excluir um log-group-level transformador 513
Processadores configuráveis do tipo analisador 513
parseJSON 514
grok 515
csv 547
parseKeyValue 550
Processadores integrados para AWS registros vendidos 552
parseWAF 552
parsePostgres 555
parseCloudfront 556
parseRoute53 557
parseVPC 558
parseToOCSF 560
Processadores de mutação de string 561
lowerCaseString 562
upperCaseString 563
splitString 564
substituteString 566
trimString 570
Processadores de mutação do JSON 571
addKeys 571
deleteKeys 573
moveKeys 574
renameKeys 576
copyValue 577
listToMap 579
Processadores de conversão de tipos de dados 585
typeConverter 585
datetimeConverter 587
Métricas e erros de transformação 589
Registros de acesso com integração de tabelas do S3 590
Entendendo a integração de tabelas do S3 590
Componentes principais 590

Fluxo de dados para tabelas do S3 591
Quando usar a integração de tabelas do S3 591
Pré-requisitos 592
Permissões do IAM 592
Política de chaves do KMS (para dados criptografados) 594
Conceitos básicos 595
Integrando tabelas do Amazon Amazon S3 com AWS serviços de análise - Usando o console Amazon S3 (Link) 596
Configurar as permissões do Lake Formation 597
Conecte-se com as ferramentas de análise 598
Filtros de métrica 599
Conceitos 600
Sintaxe de padrões de filtros para filtros de métricas 602
Configurando valores métricos para um filtro de métrica 603
Como publicar dimensões com métricas de eventos de log 604
Usando valores em eventos de log para incrementar o valor de uma métrica 607
Criar filtros de métrica 608
Criar um filtro de métrica para um grupo de logs 608
Exemplo: contar eventos de log 610
Exemplo: contar as ocorrências de um termo 611
Exemplo: contar códigos HTTP 404 613
Exemplo: contar códigos HTTP 4xx 616
Exemplo: Extrair campos de um log Apache e atribuir dimensões 617
Listagem de filtros de métrica 619
Excluir um filtro de métrica 620
Filtros de assinatura 621
Conceitos 623
Filtros de assinatura ao nível do grupo de logs 625
Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams 625
Exemplo 2: filtros de assinatura com AWS Lambda 632
Exemplo 3: filtros de assinatura com o Amazon Data Firehose 635
Exemplo 4: filtros de assinatura com o Amazon OpenSearch Service 643
Account-level filtros de assinatura 644
Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams 645
Exemplo 2: filtros de assinatura com AWS Lambda 651
Exemplo 3: filtros de assinatura com o Amazon Data Firehose 656

Cross-account Assinaturas entre regiões 663
Cross-account Compartilhamento de dados de log entre regiões usando o Amazon Kinesis Data Streams 665
Cross-account Compartilhamento de dados de registro entre regiões usando Firehose 685
Cross-account Assinaturas em nível de conta entre regiões usando o Amazon Kinesis Data Streams 699
Cross-account Assinaturas em nível de conta entre regiões usando Firehose 718
Prevenção de 'confused deputy' 731
Prevenção de repetição de logs 732
Sintaxe de padrões de filtros 734
Expressões regulares compatíveis 735
Fazer a correspondência de termos usando expressões regulares 738
Fazer a correspondência de termos em eventos de log não estruturados 738
Fazer correspondência de termos em eventos de log JSON 742
Fazer a correspondência de termos em eventos de log delimitados por espaços 751
Ativar o registro de AWS serviços 756
Destinos e permissões de registro suportados 759
Registro em log que requer permissões [v1] adicionais 774
Registros enviados para CloudWatch Logs 775
Logs enviados ao Amazon S3 778
Logs enviados ao Firehose 783
Registro em log que requer permissões [v2] adicionais 785
Exemplos de configuração de entrega de registros 786
Registros enviados para CloudWatch Logs 789
Logs enviados ao Amazon S3 793
Logs enviados ao Firehose 800
Traços enviados para X-Ray 803
Service-specific permissões 807
Console-specific permissões 807
Cross-account exemplo de entrega 809
Criar origem de entrega 810
Configurar entrega para um bucket do Amazon S3 810
Configurar entrega em um fluxo do Firehose 813
Cross-service prevenção delegada confusa 816
Automatize a ativação de registros com regras de ativação de telemetria 817
Hierarquia de avaliação das regras 817

Criação de uma regra de capacitação 818
Ativar o registro de fontes de terceiros 819
Integrações diretas de terceiros 819
Fontes adicionais via AWS Security Hub CSPM 819
Exportar dados de log para o Amazon S3 825
Conceitos 826
Exportação de dados de log para o Amazon S3 usando o console. 827
Same-account exportação (console) 828
Cross-account exportação (console) 835
Exporte dados de log para o Amazon S3 usando o AWS CLI 845
Same-account exportação (CLI) 845
Cross-account exportação (CLI) 853
Descrever as tarefas de exportação (CLI) 862
Cancelar uma tarefa de exportação (CLI) 864
Streaming de dados para o OpenSearch serviço 865
Pré-requisitos 865
Inscrever um grupo de registros no OpenSearch Serviço 866
Exemplos de código 868
Conceitos básicos 869
Olá, CloudWatch Logs 870
Ações 871
Cenários 947
Configurar a conectividade do serviço de contêiner 947
Como criar sua primeira função com tecnologia sem servidor 964
Aprenda as noções básicas sobre a ingestão de syslog 976
Executar uma consulta grande 987
Usar eventos programados para invocar uma função do Lambda 1025
Criptografe tabelas de pesquisa usando AWS KMS 1028
Como o CloudWatch Logs usa AWS KMS para tabelas de pesquisa 1028
Permissões obrigatórias 1029
Etapa 1: criar uma AWS KMS chave 1030
Etapa 2: definir permissões na chave do KMS 1030
Etapa 3: associar uma chave KMS a uma tabela de pesquisa 1032
Considerações 1033
Segurança 1034
Proteção de dados 1035

Criptografia em repouso 1036
Criptografia em trânsito 1036
Gerenciamento de identidade e acesso 1036
Autenticação 1037
Controle de acesso 1037
Visão geral do gerenciamento de acesso 1038
Usar políticas baseadas em identidade (políticas do IAM) 1043
Validação de conformidade 1073
Resiliência 1073
Segurança da infraestrutura 1074
Endpoints da VPC de interface 1074
Disponibilidade 1075
Criação de um VPC endpoint para registros CloudWatch 1075
Testando a conexão entre sua VPC e Logs CloudWatch 1075
Controle do acesso ao seu endpoint CloudWatch VPC do Logs 1076
Compatibilidade com chaves de contexto da VPC 1077
Registrando operações de API e console com AWS CloudTrail 1078
Informações de geração de consultas em CloudTrail 1083
Noções básicas sobre entradas de arquivos de log do 1084
Monitorando o uso com CloudWatch métricas 1086
CloudWatch Métricas de registros 1086
Dimensões para métricas CloudWatch de registros 1091
Métricas e dimensões do transformador de log 1092
Métricas e dimensões de centralização 1093
CloudWatch Registra métricas de uso do serviço 1097
Cotas de serviço 1100
Gerenciando suas cotas do serviço CloudWatch Logs 1112
Histórico do documento 1114
AWS Glossário 1132
..... mcxxxiii

O que é Amazon CloudWatch Logs?

Você pode usar o Amazon CloudWatch Logs para monitorar, armazenar e acessar seus arquivos de log das instâncias do Amazon Elastic Compute Cloud (Amazon EC2) AWS CloudTrail, do Route 53 e de outras fontes.

CloudWatch O Logs permite que você centralize os registros de todos os seus sistemas, aplicativos e AWS serviços que você usa, em um único serviço altamente escalável. Em seguida, você pode facilmente visualizá-los, pesquisá-los em busca de códigos de erro ou padrões específicos, filtrá-los com base em campos específicos ou arquivá-los com segurança para futuras análises. CloudWatch Os registros permitem que você veja todos os seus registros, independentemente da origem, como um fluxo único e consistente de eventos ordenados por horário.

CloudWatch O Logs também permite consultar seus registros com uma linguagem de consulta poderosa, auditar e mascarar dados confidenciais em registros e gerar métricas a partir de registros usando filtros ou um formato de registro incorporado.

CloudWatch O Logs oferece suporte a duas classes de log. Os grupos de CloudWatch registros na classe de registros padrão são compatíveis com todos os recursos de CloudWatch registros. Os grupos de CloudWatch registros na classe de registros Logs Infrequent Access incorrem em taxas de ingestão mais baixas e oferecem suporte a um subconjunto dos recursos da classe Standard. Para obter mais informações, consulte Classes de logs.

Recursos

- Duas classes de registro para flexibilidade — o CloudWatch Logs oferece duas classes de registro para que você possa ter uma opção econômica para registros que você acessa com pouca frequência. Você também tem uma opção completa para os logs que requerem monitoramento em tempo real ou outros atributos. Para obter mais informações, consulte Classes de logs.
- Consulte seus dados de registro — Você pode usar o CloudWatch Logs Insights para pesquisar e analisar interativamente seus dados de registro. Você pode realizar consultas para ajudá-lo a responder com mais eficiência e eficácia aos problemas operacionais. CloudWatch O Logs Insights inclui uma linguagem de consulta específica com alguns comandos simples, mas poderosos. Fornecemos exemplos de consultas, descrições de comandos, preenchimento automático de consultas e descoberta de campo de log para ajudar você a começar a usar. Exemplos de consultas estão incluídos para vários tipos de registros de AWS serviço. Para começar, consulte o Análise de dados de registro com o CloudWatch Logs Insights.

- Crie índices de campo para tornar as consultas mais eficientes – Você pode criar índices de campo em seus eventos de logs. Quando você usa um índice de campo em uma consulta do CloudWatch Logs Insights, a consulta tenta ignorar o processamento de eventos de registro que, sabidamente, não incluem o campo indexado. Essa consulta reduz o volume de escaneamento de suas consultas, possibilitando o retorno mais rápido dos resultados. Para começar, consulte o Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.
- Detectar e depurar usando o Live Tail: use o Live Tail para solucionar incidentes rapidamente, visualizando uma lista de streaming de novos eventos de log à medida que eles são consumidos. É possível visualizar, filtrar e realçar logs consumidos quase em tempo real, o que ajuda a detectar e resolver problemas rapidamente. Os logs podem ser filtrados com base em termos especificados, e você também pode realçar logs que contêm termos específicos para ajudar a encontrar rapidamente o que está procurando. Para obter mais informações, consulte Solução de problemas com o CloudWatch Logs Live Tail.
- Monitore registros de EC2 instâncias da Amazon — Você pode usar o CloudWatch Logs para monitorar aplicativos e sistemas usando dados de log. Por exemplo, o CloudWatch Logs pode rastrear o número de erros que ocorrem nos registros do seu aplicativo e enviar uma notificação sempre que a taxa de erros exceder um limite especificado por você. CloudWatch O Logs usa seus dados de registro para monitoramento; portanto, nenhuma alteração no código é necessária. Por exemplo, você pode monitorar registros de aplicativos em busca de termos literais específicos (como "NullReferenceException“) ou contar o número de ocorrências de um termo literal em uma posição específica nos dados de registro (como códigos de status “404" em um registro de acesso do Apache). Quando o termo que você está procurando é encontrado, o CloudWatch Logs reporta os dados para uma CloudWatch métrica especificada por você. Os dados de log são criptografados em trânsito e em repouso. Para começar, consulte o Introdução ao CloudWatch Logs.
- Monitore eventos AWS CloudTrail registrados — Você pode criar alarmes CloudWatch e receber notificações de atividades específicas da API, conforme capturadas por, CloudTrail e usar a notificação para solucionar problemas. Para começar, consulte Enviar CloudTrail eventos para CloudWatch registros no Guia do AWS CloudTrail usuário.
- Auditar e mascarar dados confidenciais: se você tiver dados confidenciais em seus logs, poderá ajudar a protegê-los com políticas de proteção de dados. Essas políticas permitem auditar e mascarar dados confidenciais. Se você ativar a proteção de dados, por padrão, os dados confidenciais que corresponderem aos identificadores de dados selecionados serão mascarados. Para obter mais informações, consulte Ajude a proteger dados de log confidenciais com mascaramento.

- Retenção de logs - por padrão, os logs são mantidos por tempo indeterminado e nunca saem da validade. Você pode ajustar a política de retenção para cada grupo de logs, mantendo a retenção indefinida ou escolhendo um período de retenção entre 10 anos e um dia.
- Proteção contra exclusão — Uma proteção que evita a exclusão acidental de grupos de registros e seus fluxos de registros. Quando ativada em um grupo de registros, a proteção contra exclusão bloqueia todas as operações de exclusão até que seja explicitamente desativada. Por padrão, a proteção contra exclusão não está ativada. Esse recurso opcional ajuda a proteger dados operacionais e de conformidade essenciais contra a remoção não intencional, como grupos de registros que contêm dados de auditoria e registros de aplicativos de produção para solução de problemas e análise.
- Arquivar dados de registro — Você pode usar o CloudWatch Logs para armazenar seus dados de log em um armazenamento altamente durável. O agente de CloudWatch registros facilita o envio rápido de dados de registro rotativos e não rotacionados de um host para o serviço de registro. Em seguida, você poderá acessar os dados de log brutos quando forem necessários.
- Registrar consultas DNS do Route 53 — Você pode usar CloudWatch os registros para registrar informações sobre as consultas de DNS que o Route 53 recebe. Para mais informações, consulte Registrar consultas de DNS no Guia do desenvolvedor do Amazon Route 53.
- Centralize os registros entre contas e regiões — Você pode usar a centralização de CloudWatch registros para definir regras de centralização entre contas e regiões que replicam os dados de registro ingeridos em ambientes de várias contas e regiões em uma região e conta centrais. Você pode configurar uma região de backup na conta central para aumentar a resiliência, definir configurações de criptografia para grupos de logs recém-criados na conta central e criar filtros métricos e de assinatura em eventos de logs de regiões e contas de origem específicas com recursos de filtragem aprimorados.

AWS Serviços relacionados

Os seguintes serviços são usados em conjunto com o CloudWatch Logs:

- AWS CloudTrailé um serviço da web que permite monitorar as chamadas feitas para a API CloudWatch Logs da sua conta, incluindo chamadas feitas pelo Console de gerenciamento da AWS, AWS Command Line Interface (AWS CLI) e outros serviços. Quando o CloudTrail registro está ativado, CloudTrail captura chamadas de API em sua conta e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Cada arquivo de log pode conter um ou mais registros, dependendo de quantas ações devem ser realizadas para atender a uma solicitação. Para obter mais informações sobre AWS CloudTrail, consulte O que é AWS CloudTrail? no

Guia do AWS CloudTrail usuário. Para obter um exemplo do tipo de dados que são CloudWatch gravados em arquivos de CloudTrail log, consulteOperações da API CloudWatch Logging Logs e do console em AWS CloudTrail.

- AWS Identity and Access Management (IAM) é um serviço web que ajuda você a controlar com segurança o acesso aos AWS recursos para seus usuários. Use o IAM para controlar quem pode usar os recursos da AWS (autenticação) e quais recursos os usuários podem usar e de que maneira (autorização). Para obter mais informações, consulte O que é o IAM? no Manual do usuário do IAM.
- O Amazon Kinesis Data Streams é um serviço da Web que você pode usar para entrada e agregação de dados rápidas e contínuas. O tipo de dados usados inclui dados de log de infraestrutura de TI, logs de aplicação, mídias sociais, feeds de dados de mercado e dados de sequência de cliques da Web. Como o tempo de resposta para a entrada e o processamento de dados é em tempo real, o processando geralmente é leve. Para obter mais informações, consulte O que é o Amazon Kinesis Data Streams? no Guia do desenvolvedor do Amazon Kinesis Data Streams.
- O AWS Lambda é um serviço da Web que você pode usar para criar aplicações que respondam rapidamente a novas informações. Carregue código da aplicação como funções Lambda, e o Lambda executará seu código em uma infraestrutura de computação de alta disponibilidade e efetuará toda a administração de recursos de computação, incluindo manutenção do servidor e do sistema operacional, provisionamento da capacidade e escalabilidade automática, implantação de códigos e patches de segurança e monitoramento do código e registro em log. Tudo o que você precisa fazer é fornecer o código em uma das linguagens compatíveis com o Lambda. Para obter mais informações, consulte O que é AWS Lambda? no Guia do AWS Lambda desenvolvedor.

Preços

Ao se inscrever AWS, você pode começar a usar o CloudWatch Logs gratuitamente usando o nível AWS gratuito.

As tarifas padrão se aplicam a registros armazenados por outros serviços usando CloudWatch registros (por exemplo, registros de fluxo do Amazon VPC e registros do Lambda).

Para obter mais informações sobre preços, consulte Amazon CloudWatch Pricing.

Para obter mais informações sobre como analisar seus custos e uso do CloudWatch Logs e CloudWatch sobre as melhores práticas sobre como reduzir seus custos, consulte CloudWatch faturamento e custo.

Conceitos do Amazon CloudWatch Logs

A terminologia e os conceitos fundamentais para sua compreensão e uso dos CloudWatch registros estão descritos abaixo.

Classe de logs

CloudWatch O Logs oferece duas classes de grupos de registros. A classe de logs Padrão é uma opção completa para os logs que exigem monitoramento em tempo real ou os logs que você acessa com frequência. A classe de logs de Acesso infrequente é uma opção de baixo custo para os logs que você acessa com menos frequência. Ele é compatível com um subconjunto dos atributos da classe de logs Padrão.

Eventos de log

Evento de log é um registro de alguma atividade registrada pela aplicação ou recurso que está sendo monitorado. O registro de eventos de log que o CloudWatch Logs compreende contém duas propriedades: a data e hora de quando o evento ocorreu e a mensagem bruta do evento. As mensagens de eventos devem estar codificadas por UTF-8.

Fluxos de log

Fluxo de logs é uma sequência de eventos de log que compartilham a mesma origem. Mais especificamente, um stream de log geralmente representa a sequência de eventos que vem da instância da aplicação ou do recurso que está sendo monitorado. Por exemplo, um stream de log pode estar associado a um log de acesso do Apache em um host específico. Quando você não precisar mais de um stream de log, poderá excluí-lo usando o delete-log-stream comando aws logs.

Grupos de logs

Os grupos de logs definem grupos de streams de log que compartilham as mesmas configurações de retenção, monitoramento e controle de acesso. Cada stream de log precisa pertencer a um grupo de logs. Por exemplo, se você tiver um stream de log separado para os logs de acesso do Apache a partir de cada host, poderá agrupar esses fluxos de log em um único grupo de log chamado `MyWebsite.com/Apache/access_log`.

Não há limite para o número de streams de log que podem pertencer a um grupo de logs.

Filtros de métrica

Você pode usar filtros de métrica para extrair as observações de métrica de eventos ingeridos e transformá-las em pontos de dados em uma métrica do CloudWatch . Filtros de métrica são

atribuídos a grupos de logs, e todos os filtros atribuídas a um grupo de logs são aplicados a seus streams de log.

Configurações de retenção

As configurações de retenção podem ser usadas para especificar por quanto tempo os eventos de registro são mantidos nos CloudWatch registros. Os eventos de log expirados serão excluídos automaticamente. Assim como os filtros de métrica, as configurações de retenção também são atribuídas a grupos de logs, e a retenção atribuída a um grupo de logs é aplicada aos seus streams de log.

Deletion protection (Proteção contra exclusão)

A proteção contra exclusão é uma proteção que evita a exclusão acidental de grupos de registros e seus fluxos de registros. Quando ativada em um grupo de registros, a proteção contra exclusão bloqueia todas as operações de exclusão até que seja explicitamente desativada. Por padrão, a proteção contra exclusão não está ativada. Esse recurso opcional ajuda a proteger dados operacionais e de conformidade essenciais contra a remoção não intencional, como grupos de registros que contêm dados de auditoria e registros de aplicativos de produção para solução de problemas e análise.

Cobrança e custo do Amazon CloudWatch Logs

Para obter informações detalhadas sobre como analisar seus custos e uso do CloudWatch Logs e CloudWatch sobre as melhores práticas sobre como reduzir seus custos, consulte CloudWatch faturamento e custo.

Para obter mais informações sobre preços, consulte Amazon CloudWatch Pricing.

Ao se inscrever AWS, você pode começar a usar o CloudWatch Logs gratuitamente usando o nível AWS gratuito.

As tarifas padrão se aplicam a registros armazenados por outros serviços usando CloudWatch registros (por exemplo, registros de fluxo do Amazon VPC e registros do Lambda).

Classes de logs

CloudWatch O Logs oferece duas classes de grupos de registros:

- A classe de registro CloudWatch Logs Standard é uma opção completa para registros que exigem monitoramento em tempo real ou registros que você acessa com frequência.
- A classe de registro CloudWatch Logs Infrequent Access é uma nova classe de registro que você pode usar para consolidar seus registros de forma econômica. Essa classe de registro oferece um subconjunto de recursos de CloudWatch registros, incluindo ingestão gerenciada, armazenamento, análise de registros entre contas e criptografia, com um preço de ingestão menor por GB. A classe de registro de acesso infrequente é ideal para consultas ad-hoc e análise after-the-fact forense em registros acessados com pouca frequência.

> **Note**
>
> Em termos de custos, as classes de logs de Acesso infrequente e Padrão só diferem nos custos de ingestão. As cobranças de armazenamento e CloudWatch as cobranças do Logs Insights são as mesmas em cada classe de registro.

Para obter mais informações sobre CloudWatch os preços do Logs, consulte Amazon CloudWatch Pricing.

> **Important**
>
> Depois que um grupo de logs é criado, sua classe de logs não pode mais ser alterada.

Recursos compatíveis

A tabela a seguir lista os atributos para cada classe de logs.

Recurso	Standard	Acesso infrequente	
Ingestão e armazenamento de logs totalmente gerenciados	Sim ✓	Sim ✓	
Atributos entre contas	Sim ✓	Sim ✓	
Criptografia com AWS KMS	Sim ✓	Sim ✓	
CloudWatch Comandos de consulta do Logs Insights	Sim ✓	Sim ✓ (A maioria dos comandos: consulte Comandos do Logs Insights QL compatíveis com as classes de log.)	
CloudWatch Campos descobertos do Logs Insights	Sim ✓	Sim ✓	
Facetas	Sim ✓	Não	
Usando o CloudWatch Pipeline para transformar registros	Sim ✓	Sim ✓	
Exportar para o Amazon S3	Sim ✓	Sim ✓	
Integração de tabelas S3	Sim ✓	Sim ✓	
Consultas agendadas	Sim ✓	Sim ✓	
Usando OpenSearch PPL ou OpenSearch SQL para consultar no CloudWatch Logs Insights;	Sim ✓	Sim ✓	

Recurso	Standard	Acesso infrequente
Assistência de consulta em linguagem natural	Sim ✓	Não
CloudWatch Detecção de anomalias de registros	Sim ✓	Não
Live Tail	Sim ✓	Não
Indexação de campo	Sim ✓	Não
Comparação com intervalo de tempo anterior	Sim ✓	Não
Filtros de assinatura	Sim ✓	Não
GetLogEvents e operações FilterLogEvents de API	Sim ✓	Não compatível. Use o CloudWatch Logs Insights para visualizar eventos de registro armazenados em grupos de registros na classe de registros de acesso infrequente.
Filtros de métrica	Sim ✓	Não
Ingestão de logs do Container Insights	Sim ✓	Não
Ingestão de logs do Lambda Insights	Sim ✓	Não

Recurso	Standard	Acesso infrequente	
Proteção de dados confidenciais com mascaramento	Sim ✓	Sim ✓	
Formato de métricas incorporadas	Sim ✓	Não	

> **Note**
>
> Além dessas duas classes de log, há uma classe de log `Delivery`. Use a classe `Delivery` log somente para entregar AWS Lambda registros para armazenar em Amazon S3 ou Amazon Data Firehose. Os eventos de registro em grupos de registros na classe de entrega são mantidos CloudWatch Logs por dois dias. Esse período de retenção é fixo e não pode ser alterado. Essa classe de log não oferece CloudWatch Logs recursos avançados, como consultas do CloudWatch Logs Insights.

Otimize os custos de armazenamento com o Amazon CloudWatch Logs Intelligent Tiering

O Amazon CloudWatch Logs Intelligent Tiering classifica automaticamente seus dados de log em três níveis de acesso com base nos padrões de acesso: padrão, acesso ocasional e acesso instantâneo de arquivamento. Com o Intelligent Tiering, você pode reter registros de alto volume e longa retenção em níveis de menor custo sem nenhuma sobrecarga operacional. Em vez de filtrar os registros ou transferi-los para soluções de armazenamento separadas, você pode mantê-los nativamente nos CloudWatch registros e se beneficiar da mesma experiência de consulta, independentemente da camada em que seus dados residam.

CloudWatch O Logs monitora os padrões de acesso e reclassifica automaticamente os dados não acessados por 30 dias para a camada Acesso Infrequente e os dados não acessados por 90 dias para a camada Archive Instant Access. Quando você acessa dados mais antigos, o CloudWatch Logs os promove automaticamente de volta ao nível Padrão por 30 dias. Ao consolidar todos os seus CloudWatch registros no Logs, você obtém visibilidade total em uma única ferramenta, eliminando a sobrecarga operacional do gerenciamento de várias soluções de armazenamento e reduzindo o tempo médio de resolução (MTTR) consultando, analisando e alertando sobre todos os seus registros em um único local.

Para começar a otimizar os custos de armazenamento, ative a hierarquização inteligente no nível da conta em uma região usando o CloudWatch console ou ligando. `PutStorageTierPolicy` Depois de ativar o Intelligent Tiering, ele se aplica a todos os grupos de registros em sua conta nessa região.

O Amazon CloudWatch Logs Intelligent Tiering está disponível em todas as regiões AWS comerciais, exceto nas regiões do Oriente Médio (Bahrein) e Oriente Médio (EAU).

CloudWatch Registra os níveis de acesso do Intelligent Tiering

Sua experiência de consulta permanece a mesma, independentemente do nível de acesso em que seus dados de log residem.

Padrão (padrão)

Esse é o nível padrão. Todos os dados de registro recém-ingeridos começam no nível Padrão. Os dados de registro permanecem no nível Padrão enquanto são acessados ativamente (0 a

30 dias desde o último acesso). O nível Standard oferece desempenho de baixa latência e alto rendimento para consultas, exportações, acompanhamento em tempo real e recuperação de registros.

Acesso infrequente

Se os dados de registro não forem acessados por 30 dias consecutivos, eles passarão para o nível Acesso ocasional. Essa camada contém dados que foram acessados pela última vez há 31 a 90 dias. O nível de acesso ocasional fornece a mesma latência de consulta e taxa de transferência do nível Standard, com um custo de armazenamento menor.

Acesso instantâneo ao arquivamento

Se os dados de log não forem acessados por 90 dias consecutivos, eles passarão para o nível Archive Instant Access. Essa camada contém dados que foram acessados pela última vez há mais de 90 dias. O nível Archive Instant Access fornece a mesma latência de consulta de milissegundos que os outros níveis, com o menor custo de armazenamento.

Ações que constituem acesso

Os padrões de acesso a seguir promovem automaticamente os dados de registro do nível Acesso ocasional ou do nível Archive Instant Access de volta ao nível Padrão. Quando qualquer uma dessas operações lê eventos de log em um nível de custo mais baixo, esses eventos de log voltam para o nível Standard por 30 dias. O cronômetro é reiniciado em cada acesso subsequente.

> **Classificação em camadas para centralização de registros**
>
> Acessar registros replicados em uma conta de destino por meio da centralização de registros entre regiões promove apenas os dados de registro na conta de destino.

- Saída com configuração de consulta ou alarme — Use `StartQuery` ou `CreateScheduledQuery` para consultar dados de log com o CloudWatch Logs Insights ou para configurar alarmes em registros
- Filtrando e recuperando eventos de log — Use `FilterLogEvents` ou `GetLogEvents` para ler eventos de log
- Exportar — Exporte dados de log para o Amazon S3 com `CreateExportTask`

Habilitando a hierarquização inteligente de CloudWatch registros

Permissões obrigatórias

Para gerenciar o nível de armazenamento da sua conta, você precisa das seguintes permissões do IAM:

- `logs:PutStorageTierPolicy`— Necessário para definir ou alterar o nível de armazenamento da sua conta.
- `logs:GetStorageTierPolicy`— Necessário para recuperar o nível de armazenamento atualmente configurado. Também é necessário habilitar o Intelligent Tiering no console.

Os usuários com a política `CloudWatchLogsFullAccess` gerenciada já têm as duas permissões. Os usuários com a política `CloudWatchLogsReadOnlyAccess` gerenciada têm a `logs:GetStorageTierPolicy` permissão.

O exemplo a seguir mostra uma declaração de política que concede as duas permissões:

```
{
    "Effect": "Allow",
    "Action": [
        "logs:PutStorageTierPolicy",
        "logs:GetStorageTierPolicy"
    ],
    "Resource": "*"
}
```

Habilitar o usando o console do

Para ativar o Intelligent Tiering usando o CloudWatch console:

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Configurações.
3. Na página Configurações, escolha a guia Registros.
4. Na seção Classificação inteligente por níveis, escolha Ativar.

Habilitar o uso do AWS CLI

Para ativar o Intelligent Tiering, use o `put-storage-tier-policy` comando:

```
aws logs put-storage-tier-policy --storage-tier INTELLIGENT_TIERING
```

O comando retorna a camada de armazenamento que foi definida e a hora da última atualização:

```
{
    "storageTier": "INTELLIGENT_TIERING",
    "lastUpdatedTime": 1751328000000
}
```

Para desativar o Intelligent Tiering, redefina o nível de armazenamento para`STANDARD`:

```
aws logs put-storage-tier-policy --storage-tier STANDARD
```

Quando você redefine o nível de armazenamento para`STANDARD`, todos os grupos de registros em sua conta são cobrados de acordo com a taxa de armazenamento padrão.

Visualizando sua configuração atual

Para visualizar o nível de armazenamento atual da sua conta, use o `get-storage-tier-policy` comando:

```
aws logs get-storage-tier-policy
```

O comando retorna o nível de armazenamento atual e a hora em que foi atualizado pela última vez:

```
{
    "storageTier": "INTELLIGENT_TIERING",
    "lastUpdatedTime": 1751328000000
}
```

Monitoramento do armazenamento por nível

Quando o Intelligent Tiering está ativado, o CloudWatch Logs publica uma métrica vendida que mostra a quantidade de dados armazenados em cada camada para cada grupo de registros:

- Namespace: `AWS/Logs`
- Nome da métrica: `StoredBytes`
- Dimensões: `LogGroupName`, `StorageType` (valores:`Standard`,`Infrequent_Access`,`Archival_Instant_Access`)

Você pode usar essa métrica para criar CloudWatch painéis que mostrem a divisão do armazenamento por camada, definam alarmes sobre os tamanhos dos níveis ou acompanhem a otimização de custos ao longo do tempo.

> **Note**
>
> Essa métrica é publicada uma vez por dia e pode não refletir as transições de camadas em tempo real.

Preços do CloudWatch Logs Intelligent Tiering

Com o Intelligent Tiering, você paga taxas de armazenamento diferentes com base no nível em que seus dados residem. Não há cobrança adicional pela ativação do Intelligent Tiering nem pelas transições de níveis. Você paga somente pelo armazenamento consumido em cada nível.

Para obter mais informações sobre preços, consulte CloudWatch Preços da Amazon.

Para visualizar os custos de armazenamento por camada, use o Athena e filtre pelos tipos de uso `TimedStorage-ByteHrs` (Padrão), `TimedStorage-IA-ByteHrs` (Acesso ocasional) e `TimedStorage-AIA-ByteHrs` (Acesso instantâneo de arquivamento). Para obter mais informações sobre CloudWatch os tipos de uso de registros em sua fatura, consulte Análise, otimização e redução de CloudWatch custos.

Introdução ao CloudWatch Logs

Para coletar registros de suas instâncias do Amazon EC2 e servidores locais em CloudWatch Logs, use o agente unificado. CloudWatch Ele permite que você colete logs e métricas avançadas com um agente. Ele oferece suporte em sistemas operacionais, incluindo servidores que executam o Windows Server. Esse agente também proporciona melhor performance.

Se você estiver usando o CloudWatch agente unificado para coletar CloudWatch métricas, ele permite a coleta de métricas adicionais do sistema, para visibilidade dos hóspedes. Ele também oferece suporte à coleta de métricas personalizadas usando `StatsD` ou `collectd`.

Para obter mais informações, consulte Instalando o CloudWatch agente no Guia CloudWatch do usuário da Amazon.

O agente CloudWatch Logs mais antigo, que suporta somente a coleta de registros de servidores que executam Linux, está obsoleto e não é mais suportado. Para obter informações sobre a migração do agente CloudWatch Logs antigo para o agente unificado, consulte Criar o arquivo de configuração do CloudWatch agente com o assistente.

Conteúdo

- Pré-requisitos
- Use o CloudWatch agente unificado para começar a usar o CloudWatch Logs
- Use o CloudWatch agente anterior para começar a usar o CloudWatch Logs
- Início rápido: use CloudFormation para começar a usar o CloudWatch Logs

Pré-requisitos

Para usar o Amazon CloudWatch Logs, você precisa de uma AWS conta. Sua AWS conta permite que você use serviços (por exemplo, Amazon EC2) para gerar registros que você pode visualizar no CloudWatch console, uma interface baseada na web. Além disso, você pode instalar e configurar o AWS Command Line Interface (AWS CLI).

Inscreva-se em um Conta da AWS

Para começar AWS, você precisa de um Conta da AWS. Para obter informações sobre como criar um Conta da AWS, consulte Introdução a um Conta da AWS no Guia de AWS Gerenciamento de contas referência.

Configurar a interface de linha de comando

Você pode usar o AWS CLI para realizar operações CloudWatch de registros.

Para obter informações sobre como instalar e configurar o AWS CLI, consulte Como configurar a interface de linha de AWS comando no Guia do AWS Command Line Interface usuário.

Use o CloudWatch agente unificado para começar a usar o CloudWatch Logs

Para obter mais informações sobre o uso do CloudWatch agente unificado para começar a usar o CloudWatch Logs, consulte Coletar métricas e registros de instâncias do Amazon EC2 e servidores locais com o CloudWatch agente no Guia do usuário da Amazon CloudWatch . Você conclui as etapas listadas nesta seção para instalar, configurar e iniciar o agente. Se você não estiver usando o agente para também coletar CloudWatch métricas, você pode ignorar qualquer seção que se refira a métricas.

Se você estiver usando o agente CloudWatch Logs antigo e quiser migrar para o novo agente unificado, recomendamos que você use o assistente incluído no novo pacote do agente. Esse assistente pode ler o arquivo de configuração atual do agente do CloudWatch Logs e configurar o CloudWatch agente para coletar os mesmos registros. Para obter mais informações sobre o assistente, consulte Criar o arquivo de configuração do CloudWatch agente com o assistente no Guia CloudWatch do usuário da Amazon.

Use o CloudWatch agente anterior para começar a usar o CloudWatch Logs

Important

CloudWatch inclui um CloudWatch agente unificado que pode coletar registros e métricas de instâncias do EC2 e servidores locais. O agente somente para logs mais antigo está obsoleto e não é mais compatível.

Para obter informações sobre a migração do antigo agente somente de registros para o agente unificado, consulte Criar o arquivo de configuração do CloudWatch agente com o assistente.

O restante desta seção explica o uso do agente CloudWatch Logs antigo para clientes que ainda o usam.

Usando o agente CloudWatch Logs, você pode publicar dados de log de instâncias do Amazon EC2 executando Linux ou Windows Server e eventos registrados de. AWS CloudTrail Em vez disso, recomendamos usar o agente CloudWatch unificado para publicar seus dados de registro. Para obter mais informações sobre o novo agente, consulte Coletar métricas e registros de instâncias do Amazon EC2 e servidores locais com o CloudWatch agente no Guia do usuário da Amazon CloudWatch .

Conteúdo

- CloudWatch Pré-requisitos do agente de registros
- Início rápido: instale e configure o agente CloudWatch Logs em uma instância Linux do EC2 em execução
- Início rápido: instale e configure o agente CloudWatch Logs em uma instância do EC2 Linux na inicialização
- Início rápido: habilite suas instâncias do Amazon EC2 executando o Windows Server 2016 para enviar registros para Logs usando o CloudWatch agente CloudWatch Logs
- Início rápido: habilite suas instâncias do Amazon EC2 executando o Windows Server 2012 e o Windows Server 2008 para enviar registros para o Logs CloudWatch
- Relate o status do agente do CloudWatch Logs
- Inicie o agente CloudWatch Logs
- Pare o agente CloudWatch de registros
- CloudWatch Referência do agente de registros

CloudWatch Pré-requisitos do agente de registros

O agente CloudWatch Logs exige Python versão 2.7, 3.0 ou 3.3 e qualquer uma das seguintes versões do Linux:

- Amazon Linux versão 2014.03.02 ou posterior. O Amazon Linux 2 não é compatível
- Ubuntu Server versão 12.04, 14.04 ou 16.04
- CentOS versão 6, 6.3, 6.4, 6.5 ou 7.0

- Red Hat Enterprise Linux (RHEL) versão 6.5 ou 7.0
- Debian 8.0

Início rápido: instale e configure o agente CloudWatch Logs em uma instância Linux do EC2 em execução

Important

O agente de registros mais antigo está obsoleto. CloudWatch inclui um agente unificado que pode coletar registros e métricas de instâncias do EC2 e servidores locais. Para obter mais informações, consulte Introdução ao CloudWatch Logs.
Para obter informações sobre a migração do agente CloudWatch Logs antigo para o agente unificado, consulte Criar o arquivo de configuração do CloudWatch agente com o assistente.
O agente de logs mais antigo é compatível somente com as versões 2.6 a 3.5 do Python. Além disso, o agente CloudWatch Logs mais antigo não é compatível com o Instance Metadata Service versão 2 (IMDSv2). Se seu servidor usa IMDSv2, você deve usar o agente unificado mais novo em vez do agente CloudWatch Logs mais antigo.
O restante desta seção explica o uso do agente CloudWatch Logs antigo para clientes que ainda o usam.

Tip

CloudWatch inclui um novo agente unificado que pode coletar registros e métricas de instâncias do EC2 e servidores locais. Se você ainda não estiver usando o agente CloudWatch Logs antigo, recomendamos usar o CloudWatch agente unificado mais novo. Para obter mais informações, consulte Introdução ao CloudWatch Logs.
Além disso, o agente mais antigo não é compatível com o Instance Metadata Service versão 2 (IMDSv2). Se seu servidor usa IMDSv2, você deve usar o agente unificado mais novo em vez do agente CloudWatch Logs mais antigo.
O restante desta seção explica o uso do antigo agente CloudWatch Logs.

Configurar o agente CloudWatch Logs antigo em uma instância Linux do EC2 em execução

Você pode usar o instalador do agente CloudWatch Logs em uma instância EC2 existente para instalar e configurar o agente CloudWatch Logs. Depois que a instalação é concluída, os logs vão automaticamente da instância para o fluxo de logs que você cria enquanto instala o agente. O agente confirma que ele foi iniciado e permanece em execução até que você o desative.

Além de usar o agente, você também pode publicar dados de registro usando o AWS CLI SDK do CloudWatch Logs ou a API CloudWatch Logs. O AWS CLI é mais adequado para publicar dados na linha de comando ou por meio de scripts. O SDK de CloudWatch registros é mais adequado para publicar dados de registro diretamente de aplicativos ou criar seu próprio aplicativo de publicação de registros.

Etapa 1: configurar sua função ou usuário do IAM para CloudWatch Logs

O agente CloudWatch Logs é compatível com funções e usuários do IAM. Se sua instância já tiver uma função do IAM associada, certifique-se de incluir a política do IAM abaixo. Se você ainda não tiver uma função do IAM atribuída à sua instância, poderá usar suas credenciais do IAM para as próximas etapas ou atribuir uma função do IAM a essa instância. Para obter mais informações, consulte Como associar uma função do IAM a uma instância.

Para configurar sua função ou usuário do IAM para CloudWatch Logs

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Escolha a função selecionando o nome (não marque a caixa de seleção ao lado do nome).
4. Escolha Attach Policies (Anexar políticas), Create Policy (Criar política).

   Uma nova guia ou janela de navegação é aberta.
5. Escolha a guia JSON e digite o seguinte documento de política JSON.

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

```
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
    ],
      "Resource": [
        "*"
    ]
  }
 ]
}
```

6. Ao concluir, selecione Revisar política. O Validador de política indica se há qualquer erro de sintaxe.
7. Na página Review Policy (Revisar política), digite um Name (Nome) e uma Description (Descrição) (opcional) para a política que você está criando. Revise o Resumo da política para ver as permissões que são concedidas pela política. Em seguida, escolha Criar política para salvar seu trabalho.
8. Feche a guia ou janela de navegador e retorne à página Add permissions (Adicionar permissões) da sua função. Escolha Refresh (Atualizar) e, em seguida, escolha a nova política para anexá-la à sua função.
9. Escolha Attach Policy.

Etapa 2: instalar e configurar CloudWatch registros em uma instância existente do Amazon EC2

O processo de instalação do agente CloudWatch Logs difere dependendo se sua instância do Amazon EC2 está executando Amazon Linux, Ubuntu, CentOS ou Red Hat. Use as etapas apropriadas para a versão do Linux na sua instância.

Para instalar e configurar CloudWatch registros em uma instância existente do Amazon Linux

A partir do Amazon Linux AMI 2014.09, o agente CloudWatch Logs está disponível como uma instalação RPM com o pacote awslogs. As versões anteriores do Amazon Linux podem acessar o pacote awslogs atualizando sua instância com o comando `sudo yum update -y`. Ao instalar o pacote awslogs como um RPM em vez de usar o instalador do CloudWatch Logs, sua instância recebe atualizações e patches regulares do pacote AWS sem precisar reinstalar manualmente o CloudWatch agente do Logs.

> **Warning**
>
> Não atualize o agente CloudWatch Logs usando o método de instalação RPM se você já usou o script Python para instalar o agente. Isso pode causar problemas de configuração que impedem que o agente do CloudWatch Logs envie seus registros para CloudWatch o.

1. Conecte-se à sua instância do Amazon Linux. Para obter mais informações, consulte Conectar-se à sua instância no Manual do usuário do Amazon EC2.

   Para obter mais informações sobre problemas de conexão, consulte Solução de problemas de conexão à sua instância no Manual do usuário do Amazon EC2.
2. Atualize sua instância do Amazon Linux para receber as alterações mais recentes nos repositórios de pacote.

   ```
   sudo yum update -y
   ```
3. Instale o pacote `awslogs`. Este é o método recomendado para instalar awslogs nas instâncias do Amazon Linux.

   ```
   sudo yum install -y awslogs
   ```
4. Edite o arquivo `/etc/awslogs/awslogs.conf` para configurar os logs a serem monitorados. Para obter mais informações sobre a edição desse arquivo, consulte CloudWatch Referência do agente de registros.
5. Por padrão, o `/etc/awslogs/awscli.conf` aponta para a região us-east-1. Para enviar seus logs a uma região diferente, edite o arquivo `awscli.conf` e especifique essa região.
6. Inicie o serviço `awslogs`.

   ```
   sudo service awslogs start
   ```

   Se você está executando o Amazon Linux 2, inicie o serviço `awslogs` com o comando a seguir.

   ```
   sudo systemctl start awslogsd
   ```
7. (Opcional) Verifique o arquivo `/var/log/awslogs.log` para ver se há erros registrados ao iniciar o serviço.

8. (Opcional) Execute o comando a seguir para iniciar o serviço `awslogs` em cada inicialização do sistema.

   ```
   sudo chkconfig awslogs on
   ```

   Se você estiver executando o Amazon Linux 2, use o comando a seguir para iniciar o serviço a cada inicialização do sistema.

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

9. Você deve ver o grupo de logs recém-criado e o fluxo de logs no console do CloudWatch depois de alguns minutos de execução do agente.

   Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch Logs.

Para instalar e configurar CloudWatch registros em uma instância existente do Ubuntu Server, CentOS ou Red Hat

Se você estiver usando uma AMI executando o Ubuntu Server, CentOS ou Red Hat, use o procedimento a seguir para instalar manualmente o agente CloudWatch Logs na sua instância.

1. Conecte-se à sua instância do EC2. Para obter mais informações, consulte Conectar-se à sua instância no Manual do usuário do Amazon EC2.

   Para obter mais informações sobre problemas de conexão, consulte Solução de problemas de conexão à sua instância no Manual do usuário do Amazon EC2.

2. Execute o instalador do agente CloudWatch Logs usando uma das duas opções. Você pode executá-lo diretamente na internet ou fazer download dos arquivos e executá-lo de forma autônoma.

   > **Note**
   >
   > Se você estiver executando CentOS 6.x, Red Hat 6.x ou Ubuntu 12.04, use as etapas para baixar e executar o instalador autônomo. A instalação do agente CloudWatch Logs diretamente da Internet não é compatível com esses sistemas.

> **Note**
>
> No Ubuntu, execute `apt-get update` antes de executar os comandos a seguir.

Para executá-lo diretamente na internet, use os seguintes comandos e siga as instruções:

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

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

Se o comando anterior não funcionar, tente o seguinte:

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

Para fazer download e executá-lo de forma independente, use os estes comandos e siga as instruções:

```
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
```

Você pode instalar o agente CloudWatch Logs especificando us-east-1, us-west-1, us-west-2, ap-south-1, ap-northeast-2, ap-northeast-2, ap-southeast-1, ap-southeast-1, ap-southeast-2, ap-northeast-1, eutheast-2 regiões eu-central-1, eu-west-1 ou sa-east-1.

> **Note**
>
> Para obter mais informações sobre a versão atual e o histórico das versões de `awslogs-agent-setup`, consulte CHANGELOG.txt.

O instalador do agente CloudWatch Logs exige certas informações durante a configuração. Antes de começar, você precisa saber qual arquivo de log monitorar e seu formato do time stamp. Você também deve ter as seguintes informações à mão.

Item	Description
AWS ID da chave de acesso	Aperte Enter se estiver usando uma função do IAM. Caso contrário , insira o ID da chave de AWS acesso.
AWS chave de acesso secreta	Aperte Enter se estiver usando uma função do IAM. Caso contrário , insira sua chave de acesso AWS secreta.
Nome da região padrão	Pressione Enter. A região padrão é us-east-2. É possível definir para 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 ou sa-east-1.
Formato de saída padrão	Deixe em branco e pressione Enter.
Caminho do arquivo de log para upload	A localização do arquivo que contém os dados do log a ser enviado. O instalador sugere um caminho para você.
Nome do grupo de logs de destino	O nome para o seu grupo de logs. O instalador sugere um nome de grupo de logs para você.
Nome do stream de log de destino	Por padrão, esse é o nome do host. O instalador sugere um nome de host para você.
Formato de time stamp	Especifique o formato do time stamp no arquivo de log especific ado. Escolha Personalizado para especificar seu próprio formato.

Item	Description
Posição inicial	Como é feito upload dos dados. Defina isso como start_of_file para fazer upload de tudo no arquivo de dados. Defina como end_of_file para fazer upload somente dos dados recém-acrescentados.

Depois de concluir essas etapas, o instalador pergunta se você deseja configurar outro arquivo de log. Você pode executar o processo quantas vezes quiser para cada arquivo de log. Se você não tiver mais arquivos de log para monitorar, escolha N quando o instalador solicitar a configuração de outro log. Para obter mais informações sobre as configurações no arquivo de configuração do agente, consulte CloudWatch Referência do agente de registros.

> **Note**
>
> A configuração de várias fontes de log para enviar dados a um único stream de logs não é suportada.

3. Você deve ver o grupo de logs recém-criado e o fluxo de logs no console do CloudWatch depois de alguns minutos de execução do agente.

   Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch Logs.

Início rápido: instale e configure o agente CloudWatch Logs em uma instância do EC2 Linux na inicialização

> **Tip**
>
> O antigo agente do CloudWatch Logs discutido nesta seção está prestes a ser descontinuado. É altamente recomendável que você use o novo CloudWatch agente unificado que pode coletar registros e métricas. Além disso, o agente CloudWatch Logs mais antigo exige o Python 3.3 ou anterior, e essas versões não são instaladas em novas instâncias do EC2 por padrão. Para obter mais informações sobre o CloudWatch agente unificado, consulte Instalando o CloudWatch agente.
> O restante desta seção explica o uso do antigo agente CloudWatch Logs.

Instalando o agente CloudWatch Logs antigo em uma instância Linux do EC2 na inicialização

Você pode usar os dados do usuário do Amazon EC2, um recurso do Amazon EC2 que permite que informações paramétricas sejam passadas para a instância na inicialização, para instalar e configurar CloudWatch o agente Logs nessa instância. Para passar as informações de instalação e configuração do agente CloudWatch Logs para o Amazon EC2, você pode fornecer o arquivo de configuração em um local de rede, como um bucket do Amazon S3.

A configuração de várias fontes de log para enviar dados a um único stream de logs não é suportada.

Pré-requisito

Crie um arquivo de configuração do agente que descreva todos os seus grupos e streams de logs. Trata-se de um arquivo de texto que descreve os arquivos de log a serem monitorados, bem como os grupos e os streams de logs para os quais será feito upload deles. O agente consome esse arquivo de configuração e inicia o monitoramento e o upload de todos os arquivos de log descritos nele. Para obter mais informações sobre as configurações no arquivo de configuração do agente, consulte CloudWatch Referência do agente de registros.

Veja a seguir o exemplo de um arquivo de configuração do agente para o 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
```

Veja a seguir uma amostra de um arquivo de configuração do agente para o 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
```

Para configurar sua função do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Políticas, Criar política.
3. Na página Criar política, em Criar sua própria política, escolha Selecionar. Para obter mais informações sobre a criação de políticas personalizadas, consulte Políticas do IAM para o Amazon EC2 no Manual do usuário do Amazon EC2.
4. Na página Revisar política, em Nome da política, digite um nome para a política.
5. Em Documento da política, cole a política a seguir:

   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. Escolha Criar política.
7. No painel de navegação, escolha Funções, Criar nova função.
8. Na página Definir nome da função, digite um nome para a função e escolha Próxima etapa.
9. Na página Selecionar tipo de função, escolha Selecionar ao lado de Amazon EC2.
10. Na página Anexar política, no cabeçalho da tabela escolha Tipo de política, Gerenciado pelo cliente.
11. Selecione a política do IAM que você acabou de criar e escolha Next Step (Próxima etapa).
12. Selecione Criar função.

    Para obter mais informações sobre usuários e políticas, consulte Usuários e grupos do IAM e Gerenciar políticas do IAM no Guia do usuário do IAM.

Para iniciar uma nova instância e ativar CloudWatch os registros

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Executar instância.

   Para obter mais informações, consulte Iniciar uma instância no Manual do usuário do Amazon EC2.
3. Na página Etapa 1: escolher uma Imagem de Máquina da Amazon (AMI), selecione o tipo de instância Linux para iniciar e, em seguida, na página Etapa 2: escolher um tipo de instância, selecione Próximo: configurar detalhes da instância.

   Certifique-se de que cloud-init seja incluído na sua Imagem de máquina da Amazon (AMI). O Amazon Linux AMIs e AMIs o Ubuntu e o RHEL já incluem o cloud-init, mas o CentOS e outros talvez não. AMIs AWS Marketplace
4. Na página Etapa 3: Configurar detalhes da instância, em IAM role (Função do IAM), selecione a função do IAM que você criou.
5. Em Detalhes avançados, em Dados do usuário, cole o script a seguir na caixa. Em seguida, atualize esse script alterando o valor da opção -c para o local do seu arquivo de configuração do agente:

```
#!/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. Faça todas as outras alterações para a instância, revise suas configurações de execução e, em seguida, escolha Iniciar.
7. Você deve ver o grupo de registros e o stream de registros recém-criados no CloudWatch console depois que o agente estiver em execução por alguns instantes.

   Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch Logs.

Início rápido: habilite suas instâncias do Amazon EC2 executando o Windows Server 2016 para enviar registros para Logs usando o CloudWatch agente CloudWatch Logs

> **Tip**
>
> CloudWatch inclui um novo agente unificado que pode coletar registros e métricas de instâncias do EC2 e servidores locais. Recomendamos que você o mais recente agente unificado do CloudWatch. Para obter mais informações, consulte Introdução ao CloudWatch Logs.
> O restante desta seção explica o uso do antigo agente CloudWatch Logs.

Habilite suas instâncias do Amazon EC2 executando o Windows Server 2016 para enviar registros para Logs usando o agente de CloudWatch Logs mais antigo CloudWatch

Há vários métodos que você pode usar para permitir que instâncias que executam o Windows Server 2016 enviem registros para o CloudWatch Logs. As etapas desta seção usam o Run Command do Systems Manager. Para obter informações sobre os outros métodos possíveis, consulte Envio de registros, eventos e contadores de desempenho para a Amazon CloudWatch.

Etapas

- Baixe o arquivo de configuração de exemplo
- Configure o arquivo JSON para CloudWatch
- Criar um perfil do IAM para o Systems Manager
- Verifique os pré-requisitos do Systems Manager

- Verificar o acesso à Internet
- Ativar CloudWatch registros usando o comando de execução do Systems Manager

Baixe o arquivo de configuração de exemplo

Baixe o seguinte arquivo de amostra em seu computador: `AWS.EC2.Windows.CloudWatch.json`.

Configure o arquivo JSON para CloudWatch

Você determina para quais registros enviar CloudWatch especificando suas opções em um arquivo de configuração. O processo de criação desse arquivo e a especificação de suas opções pode levar 30 minutos ou mais para serem concluídos. Após concluir essa tarefa uma vez, você pode reutilizar o arquivo de configuração em todas as instâncias.

Etapas

- Etapa 1: ativar CloudWatch registros
- Etapa 2: definir as configurações para CloudWatch
- Etapa 3: Configurar os dados a serem enviados
- Etapa 4: Configurar o controle de fluxo
- Etapa 5: Salvar conteúdo JSON

Etapa 1: ativar CloudWatch registros

Na parte superior do arquivo JSON, altere "false" para "true" em `IsEnabled`:

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

Etapa 2: definir as configurações para CloudWatch

Especifique as credenciais, a região, um nome de grupo de logs e um namespace de fluxo de logs. Isso permite que a instância envie dados de registro para o CloudWatch Logs. Para enviar os mesmos dados de registro para locais diferentes, você pode adicionar seções adicionais com regiões exclusivas IDs (por exemplo, “CloudWatchLogs2" e CloudWatchLogs 3") e diferentes para cada ID.

Para definir as configurações para enviar dados de registro para o CloudWatch Logs

1. No arquivo JSON, localize a seção `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. Deixe os campos `AccessKey` e `SecretKey` em branco. Você configura as credenciais usando uma função do IAM.
3. Em `Region`, digite a região para a qual enviar os dados de log (por exemplo, `us-east-2`).
4. Em `LogGroup`, digite o nome do grupo de logs. Esse nome aparece na tela Log Groups (Grupos de logs) no console do CloudWatch.
5. Em `LogStream`, digite o fluxo de log de destino. Esse nome aparece na tela Grupos de registros > Streams no CloudWatch console.

   Se você usar `{instance_id}`, o padrão, o nome do fluxo de logs será o ID dessa instância.

   Se você especificar um nome de fluxo de registros que ainda não existe, o CloudWatch Logs o cria automaticamente para você. Você pode definir um nome de fluxo de log usando uma sequência literal, as variáveis predefinidas `{hostname}`, `{instance_id}` e `{ip_address}` ou uma combinação delas.

Etapa 3: Configurar os dados a serem enviados

Você pode enviar dados de registro de eventos, dados de rastreamento de eventos para Windows (ETW) e outros dados de registro para CloudWatch o Logs.

Para enviar dados do registro de eventos do aplicativo Windows para o CloudWatch Logs

1. No arquivo JSON, localize a seção `ApplicationEventLog`.

```
{
    "Id": "ApplicationEventLog",
```

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

2. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar dados do registro de segurança para o CloudWatch Logs

1. No arquivo JSON, localize a seção `SecurityEventLog`.

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

2. Em `Levels`, digite **7** para fazer upload de todas as mensagens.

Para enviar dados do registro de eventos do sistema para o CloudWatch Logs

1. No arquivo JSON, localize a seção `SystemEventLog`.

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

2. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar outros tipos de dados de registro de eventos para o CloudWatch Logs

1. No arquivo JSON, adicione uma nova seção. Cada seção deve ter um `Id` exclusivo.

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

2. Em `Id`, digite o nome do log para fazer upload (por exemplo, **`WindowsBackup`**).
3. Em `LogName`, digite o nome do log do qual fazer upload. Você pode localizar o nome do log da seguinte forma.

a. Abra o Visualizador de eventos.

b. No painel de navegação, escolha Applications and Services Logs.

c. Navegue até o log e escolha Actions, Properties.

4. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar dados do Event Tracing for Windows para o Logs CloudWatch

O ETW (Rastreamento de Eventos para Windows) fornece um mecanismo de registro eficiente e detalhado no qual os aplicativos podem gravar logs. Cada ETW é controlado por um gerente de sessão que pode iniciar e parar a sessão de registro. Cada sessão tem um provedor e um ou mais consumidores.

1. No arquivo JSON, localize a seção `ETW`.

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

2. Em `LogName`, digite o nome do log do qual fazer upload.

3. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

- **1** – fazer upload apenas de mensagens de erro.
- **2** – fazer upload apenas de mensagens de aviso.
- **4** – fazer upload apenas de mensagens informativas.

Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar registros personalizados (qualquer arquivo de registro baseado em texto) para o Logs CloudWatch

1. No arquivo JSON, localize a seção `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. Em `LogDirectoryPath`, digite o caminho onde os logs estão armazenados na instância.
3. Em `TimestampFormat`, digite o formato do time stamp a ser usado. Para obter mais informações sobre os valores compatíveis, consulte o tópico Cadeias de caracteres de formato de data e hora personalizado no MSDN.

> **Important**
>
> Seu arquivo de log de origem deve ter o time stamp no início de cada linha do log e deve haver um espaço depois do time stamp.

4. Em `Encoding`, digite a codificação de arquivo a ser usada (por exemplo, UTF-8). Para obter uma lista de valores compatíveis, consulte o tópico Classe Encoding no MSDN.

   > **Note**
   >
   > Use o nome da codificação, não o nome da exibição.

5. (Opcional) Em `Filter`, digite o prefixo de nomes de logs. Deixe esse parâmetro em branco para monitorar todos os arquivos. Para obter mais informações sobre os valores suportados, consulte o tópico FileSystemWatcherFilter Propriedade no MSDN.

6. (Opcional) Em `CultureName`, digite a localidade na qual o time stamp é registrado no log. Se `CultureName` ficar em branco, será adotada como padrão a mesma localidade usada atualmente pela instância do Windows. Para obter mais informações a respeito, consulte a coluna `Language tag` na tabela no tópico Product Behavior no MSDN.

   > **Note**
   >
   > Os valores `div`, `div-MV`, `hu` e `hu-HU` não têm suporte.

7. (Opcional) Em `TimeZoneKind`, digite `Local` ou `UTC`. Você pode definir isso para fornecer informações de fuso horário quando essas informações não estiverem incluídas no time stamp do log. Se esse parâmetro for deixado em branco e se seu carimbo de data/hora não incluir informações de fuso horário, o CloudWatch Logs usará como padrão o fuso horário local. Esse parâmetro será ignorado se o time stamp já incluir informações sobre o fuso horário.

8. (Opcional) Em `LineCount`, digite o número de linhas do cabeçalho para identificar o arquivo de log. Por exemplo, os arquivos de log do IIS têm cabeçalhos praticamente idênticos. Você pode especificar **5**, que determina a leitura das três primeiras linhas do cabeçalho do arquivo de log para identificá-lo. Nos arquivos de log do IIS, a terceira linha é o carimbo de data/hora, mas nem sempre é garantido que o carimbo será diferente entre os arquivos de log. Por esse motivo, recomendamos incluir pelo menos uma linha de dados de log reais para identificar o arquivo de log de forma exclusiva.

Para enviar dados de log do IIS para CloudWatch Logs

1. No arquivo JSON, localize a seção `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. Em `LogDirectoryPath`, digite a pasta onde os logs do IIS são armazenados para um site individual (por exemplo, `C:\inetpub\logs\LogFiles\W3SVC`*n*).

> **Note**
>
> Há suporte apenas para o formato de log W3C. Não há suporte para os formatos IIS, NCSA e Personalizado.

3. Em `TimestampFormat`, digite o formato do time stamp a ser usado. Para obter mais informações sobre os valores compatíveis, consulte o tópico Cadeias de caracteres de formato de data e hora personalizado no MSDN.
4. Em `Encoding`, digite a codificação de arquivo a ser usada (por exemplo, UTF-8). Para obter mais informações sobre os valores compatíveis, consulte o tópico Encoding Class no MSDN.

> **Note**
>
> Use o nome da codificação, não o nome da exibição.

5. (Opcional) Em `Filter`, digite o prefixo de nomes de logs. Deixe esse parâmetro em branco para monitorar todos os arquivos. Para obter mais informações sobre os valores suportados, consulte o tópico FileSystemWatcherFilter Propriedade no MSDN.
6. (Opcional) Em `CultureName`, digite a localidade na qual o time stamp é registrado no log. Se `CultureName` ficar em branco, será adotada como padrão a mesma localidade usada atualmente pela instância do Windows. Para obter mais informações sobre os valores compatíveis, consulte a coluna `Language tag` na tabela no tópico Product Behavior no MSDN.

> **Note**
>
> Os valores `div`, `div-MV`, `hu` e `hu-HU` não têm suporte.

7. (Opcional) Em `TimeZoneKind`, digite `Local` ou `UTC`. Você pode definir isso para fornecer informações de fuso horário quando essas informações não estiverem incluídas no time stamp do log. Se esse parâmetro for deixado em branco e se seu carimbo de data/hora não incluir informações de fuso horário, o CloudWatch Logs usará como padrão o fuso horário local. Esse parâmetro será ignorado se o time stamp já incluir informações sobre o fuso horário.
8. (Opcional) Em `LineCount`, digite o número de linhas do cabeçalho para identificar o arquivo de log. Por exemplo, os arquivos de log do IIS têm cabeçalhos praticamente idênticos. Você pode especificar **5**, que leria as cinco primeiras linhas do cabeçalho do arquivo de log para identificá-lo. Nos arquivos de log do IIS, a terceira linha é o carimbo de data/hora, mas nem sempre é garantido que o carimbo será diferente entre os arquivos de log. Por esse motivo, recomendamos incluir pelo menos uma linha de dados de log reais para identificar o arquivo de log de forma exclusiva.

Etapa 4: Configurar o controle de fluxo

Cada tipo de dados deve ter um destino correspondente na seção `Flows`. Por exemplo, para enviar o log personalizado, o log ETW e o log do sistema para CloudWatch Logs, adicione `(CustomLogs,ETW,SystemEventLog),CloudWatchLogs` à `Flows` seção.

> **Warning**
>
> A adição de uma etapa inválida bloqueia o fluxo. Por exemplo, se você adicionar uma etapa de métrica de disco, mas a instância não tiver um disco, todas as etapas do fluxo serão bloqueadas.

Você pode enviar o mesmo arquivo de log a mais de um destino. Por exemplo, para enviar o log do aplicativo a dois destinos diferentes que você definiu na seção `CloudWatchLogs`, adicione `ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)` à seção `Flows`.

Para configurar o controle de fluxo

1. No arquivo `AWS.EC2.Windows.CloudWatch.json`, localize a seção `Flows`.

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

2. Em `Flows`, adicione cada tipo de dados os quais serão feito upload (por exemplo, `ApplicationEventLog`) e seu destino (por exemplo, `CloudWatchLogs`).

Etapa 5: Salvar conteúdo JSON

Agora você concluiu a edição do arquivo JSON. Salve-o e cole o conteúdo do arquivo em um editor de texto em outra janela. Você precisará do conteúdo do arquivo em uma etapa posterior deste procedimento.

Criar um perfil do IAM para o Systems Manager

É necessário ter uma função do IAM para credenciais de instância ao usar o Run Command do Systems Manager. Essa função permite que o Systems Manager execute ações na instância. Para obter mais informações, consulte Configurar funções de segurança para o Systems Manager no Manual do usuário do AWS Systems Manager . Para obter informações sobre como anexar esse perfil do IAM a uma instância existente, consulte Anexar uma função do IAM a uma instância, no Manual do usuário do Amazon EC2.

Verifique os pré-requisitos do Systems Manager

Antes de usar o Systems Manager Run Command para configurar a integração com o CloudWatch Logs, verifique se suas instâncias atendem aos requisitos mínimos. Para obter mais informações, consulte Pré-requisitos do Systems Manager no Manual do usuário do AWS Systems Manager .

Verificar o acesso à Internet

Suas instâncias Windows Server e instâncias gerenciadas do Amazon EC2 devem ter acesso de saída à Internet para enviar dados de log e eventos para. CloudWatch Para obter mais informações sobre como configurar o acesso à Internet, consulte Gateways da Internet no Manual do usuário da Amazon VPC.

Ativar CloudWatch registros usando o comando de execução do Systems Manager

O Run Command permite gerenciar a configuração de suas instâncias sob demanda. Você especifica um documento do Systems Manager, especifica parâmetros e executa o comando em uma ou mais instâncias. O SSM Agent na instância processa o comando e configura a instância conforme especificado.

Para configurar a integração com o CloudWatch Logs usando o comando Executar

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Abra o console SSM em. https://console.aws.amazon.com/systems-manager/
3. No painel de navegação, selecione Executar comando.
4. Escolha Executar um comando.
5. Para o documento de comando, escolha AWS- ConfigureCloudWatch.
6. Para instâncias do Target, escolha as instâncias a serem integradas com o CloudWatch Logs. Se você não vir uma instância nessa lista, ela poderá não estar configurada para Run Command. Para obter mais informações, consulte Pré-requisitos do Systems Manager no Manual do usuário do Amazon EC2.
7. Em Status, escolha Enabled.
8. Em Propriedades, copie e cole o conteúdo JSON que você criou nas tarefas anteriores.
9. Preencha os campos opcionais restantes e escolha Run.

Use o seguinte procedimento para visualizar os resultados da execução do comando no console do Amazon EC2.

Para visualizar a saída do comando no console

1. Selecione um comando.
2. Escolha a guia Output.
3. Escolha View Output. A página de saída do comando mostra os resultados da execução do comando.

Início rápido: habilite suas instâncias do Amazon EC2 executando o Windows Server 2012 e o Windows Server 2008 para enviar registros para o Logs CloudWatch

> **Tip**
>
> CloudWatch inclui um novo agente unificado que pode coletar registros e métricas de instâncias do EC2 e servidores locais. Recomendamos que você o mais recente agente unificado do CloudWatch. Para obter mais informações, consulte Introdução ao CloudWatch Logs.
> O restante desta seção explica o uso do antigo agente CloudWatch Logs.

Habilite suas instâncias do Amazon EC2 executando o Windows Server 2012 e o Windows Server 2008 para enviar registros para Logs CloudWatch

Use as etapas a seguir para permitir que suas instâncias que executam o Windows Server 2012 e o Windows Server 2008 enviem CloudWatch registros para o Logs.

Baixe o arquivo de configuração de exemplo

Baixe os seguintes arquivos JSON em seu computador: `AWS.EC2.Windows.CloudWatch.json`. Você o edita nas etapas a seguir.

Configure o arquivo JSON para CloudWatch

Você determina para quais registros enviar CloudWatch especificando suas opções no arquivo de configuração JSON. O processo de criação desse arquivo e a especificação de suas opções pode levar 30 minutos ou mais para serem concluídos. Após concluir essa tarefa uma vez, você pode reutilizar o arquivo de configuração em todas as instâncias.

Etapas

- Etapa 1: ativar CloudWatch registros
- Etapa 2: definir as configurações para CloudWatch
- Etapa 3: Configurar os dados a serem enviados
- Etapa 4: Configurar o controle de fluxo

Etapa 1: ativar CloudWatch registros

Na parte superior do arquivo JSON, altere "false" para "true" em `IsEnabled`:

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

Etapa 2: definir as configurações para CloudWatch

Especifique as credenciais, a região, um nome de grupo de logs e um namespace de fluxo de logs. Isso permite que a instância envie dados de registro para o CloudWatch Logs. Para enviar os mesmos dados de registro para locais diferentes, você pode adicionar seções adicionais com regiões exclusivas IDs (por exemplo, “CloudWatchLogs2" e CloudWatchLogs 3") e diferentes para cada ID.

Para definir as configurações para enviar dados de registro para o CloudWatch Logs

1. No arquivo JSON, localize a seção `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. Deixe os campos `AccessKey` e `SecretKey` em branco. Você configura as credenciais usando uma função do IAM.
3. Em `Region`, digite a região para a qual enviar os dados de log (por exemplo, `us-east-2`).
4. Em `LogGroup`, digite o nome do grupo de logs. Esse nome aparece na tela Log Groups (Grupos de logs) no console do CloudWatch.
5. Em `LogStream`, digite o fluxo de log de destino. Esse nome aparece na tela Grupos de registros > Streams no CloudWatch console.

   Se você usar `{instance_id}`, o padrão, o nome do fluxo de logs será o ID dessa instância.

Se você especificar um nome de fluxo de registros que ainda não existe, o CloudWatch Logs o cria automaticamente para você. Você pode definir um nome de fluxo de log usando uma sequência literal, as variáveis predefinidas `{hostname}`, `{instance_id}` e `{ip_address}` ou uma combinação delas.

Etapa 3: Configurar os dados a serem enviados

Você pode enviar dados de registro de eventos, dados de rastreamento de eventos para Windows (ETW) e outros dados de registro para CloudWatch o Logs.

Para enviar dados do registro de eventos do aplicativo Windows para o CloudWatch Logs

1. No arquivo JSON, localize a seção `ApplicationEventLog`.

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

2. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar dados do registro de segurança para o CloudWatch Logs

1. No arquivo JSON, localize a seção `SecurityEventLog`.

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

2. Em `Levels`, digite **7** para fazer upload de todas as mensagens.

Para enviar dados do registro de eventos do sistema para o CloudWatch Logs

1. No arquivo JSON, localize a seção `SystemEventLog`.

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

2. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar outros tipos de dados de registro de eventos para o CloudWatch Logs

1. No arquivo JSON, adicione uma nova seção. Cada seção deve ter um `Id` exclusivo.

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

2. Em `Id`, digite o nome do log para fazer upload (por exemplo, **`WindowsBackup`**).
3. Em `LogName`, digite o nome do log do qual fazer upload. Você pode localizar o nome do log da seguinte forma.

   a. Abra o Visualizador de eventos.

   b. No painel de navegação, escolha Applications and Services Logs.

   c. Navegue até o log e escolha Actions, Properties.
4. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar dados do Event Tracing for Windows para o Logs CloudWatch

O ETW (Rastreamento de Eventos para Windows) fornece um mecanismo de registro eficiente e detalhado no qual os aplicativos podem gravar logs. Cada ETW é controlado por um gerente de sessão que pode iniciar e parar a sessão de registro. Cada sessão tem um provedor e um ou mais consumidores.

1. No arquivo JSON, localize a seção `ETW`.

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

2. Em `LogName`, digite o nome do log do qual fazer upload.
3. Em `Levels`, especifique o tipo de mensagens das quais fazer upload. É possível especificar um dos seguintes valores:

   - **1** – fazer upload apenas de mensagens de erro.
   - **2** – fazer upload apenas de mensagens de aviso.
   - **4** – fazer upload apenas de mensagens informativas.

   Você pode combinar valores para incluir mais de um tipo de mensagem. Por exemplo, o valor **3** faz upload de mensagens de erro (**1**) e de mensagens de aviso (**2**). Um valor **7** faz upload de mensagens de erro (**1**), de mensagens de aviso (**2**) e de mensagens informativas (**4**).

Para enviar registros personalizados (qualquer arquivo de registro baseado em texto) para o Logs CloudWatch

1. No arquivo JSON, localize a seção `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. Em `LogDirectoryPath`, digite o caminho onde os logs estão armazenados na instância.
3. Em `TimestampFormat`, digite o formato do time stamp a ser usado. Para obter mais informações sobre os valores compatíveis, consulte o tópico Cadeias de caracteres de formato de data e hora personalizado no MSDN.

> **Important**
>
> Seu arquivo de log de origem deve ter o time stamp no início de cada linha do log e deve haver um espaço depois do time stamp.

4. Em `Encoding`, digite a codificação de arquivo a ser usada (por exemplo, UTF-8). Para obter mais informações sobre os valores compatíveis, consulte o tópico Encoding Class no MSDN.

> **Note**
>
> Use o nome da codificação, não o nome da exibição.

5. (Opcional) Em `Filter`, digite o prefixo de nomes de logs. Deixe esse parâmetro em branco para monitorar todos os arquivos. Para obter mais informações sobre os valores suportados, consulte o tópico FileSystemWatcherFilter Propriedade no MSDN.
6. (Opcional) Em `CultureName`, digite a localidade na qual o time stamp é registrado no log. Se `CultureName` ficar em branco, será adotada como padrão a mesma localidade usada atualmente pela instância do Windows. Para obter mais informações sobre os valores compatíveis, consulte a coluna `Language tag` na tabela no tópico Product Behavior no MSDN.

> **Note**
>
> Os valores `div`, `div-MV`, `hu` e `hu-HU` não têm suporte.

7. (Opcional) Em `TimeZoneKind`, digite `Local` ou `UTC`. Você pode definir isso para fornecer informações de fuso horário quando essas informações não estiverem incluídas no time stamp do log. Se esse parâmetro for deixado em branco e se seu carimbo de data/hora não incluir

informações de fuso horário, o CloudWatch Logs usará como padrão o fuso horário local. Esse parâmetro será ignorado se o time stamp já incluir informações sobre o fuso horário.

8. (Opcional) Em `LineCount`, digite o número de linhas do cabeçalho para identificar o arquivo de log. Por exemplo, os arquivos de log do IIS têm cabeçalhos praticamente idênticos. Você pode especificar **5**, que determina a leitura das três primeiras linhas do cabeçalho do arquivo de log para identificá-lo. Nos arquivos de log do IIS, a terceira linha é o carimbo de data/hora, mas nem sempre é garantido que o carimbo será diferente entre os arquivos de log. Por esse motivo, recomendamos incluir pelo menos uma linha de dados de log reais para identificar o arquivo de log de forma exclusiva.

Para enviar dados de log do IIS para CloudWatch Logs

1. No arquivo JSON, localize a seção `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. Em `LogDirectoryPath`, digite a pasta onde os logs do IIS são armazenados para um site individual (por exemplo, `C:\inetpub\logs\LogFiles\W3SVC`*n*).

> **Note**
>
> Há suporte apenas para o formato de log W3C. Não há suporte para os formatos IIS, NCSA e Personalizado.

3. Em `TimestampFormat`, digite o formato do time stamp a ser usado. Para obter mais informações sobre os valores compatíveis, consulte o tópico Cadeias de caracteres de formato de data e hora personalizado no MSDN.
4. Em `Encoding`, digite a codificação de arquivo a ser usada (por exemplo, UTF-8). Para obter mais informações sobre os valores compatíveis, consulte o tópico Encoding Class no MSDN.

   > **Note**
   >
   > Use o nome da codificação, não o nome da exibição.

5. (Opcional) Em `Filter`, digite o prefixo de nomes de logs. Deixe esse parâmetro em branco para monitorar todos os arquivos. Para obter mais informações sobre os valores suportados, consulte o tópico FileSystemWatcherFilter Propriedade no MSDN.
6. (Opcional) Em `CultureName`, digite a localidade na qual o time stamp é registrado no log. Se `CultureName` ficar em branco, será adotada como padrão a mesma localidade usada atualmente pela instância do Windows. Para obter mais informações sobre os valores compatíveis, consulte a coluna `Language tag` na tabela no tópico Product Behavior no MSDN.

   > **Note**
   >
   > Os valores `div`, `div-MV`, `hu` e `hu-HU` não têm suporte.

7. (Opcional) Em `TimeZoneKind`, digite `Local` ou `UTC`. Você pode definir isso para fornecer informações de fuso horário quando essas informações não estiverem incluídas no time stamp do log. Se esse parâmetro for deixado em branco e se seu carimbo de data/hora não incluir informações de fuso horário, o CloudWatch Logs usará como padrão o fuso horário local. Esse parâmetro será ignorado se o time stamp já incluir informações sobre o fuso horário.
8. (Opcional) Em `LineCount`, digite o número de linhas do cabeçalho para identificar o arquivo de log. Por exemplo, os arquivos de log do IIS têm cabeçalhos praticamente idênticos. Você pode especificar **5**, que leria as cinco primeiras linhas do cabeçalho do arquivo de log para identificá-lo. Nos arquivos de log do IIS, a terceira linha é o carimbo de data/hora, mas nem sempre é garantido que o carimbo será diferente entre os arquivos de log. Por esse motivo, recomendamos incluir pelo menos uma linha de dados de log reais para identificar o arquivo de log de forma exclusiva.

Etapa 4: Configurar o controle de fluxo

Cada tipo de dados deve ter um destino correspondente na seção `Flows`. Por exemplo, para enviar o log personalizado, o log ETW e o log do sistema para CloudWatch Logs, adicione `(CustomLogs,ETW,SystemEventLog),CloudWatchLogs` à `Flows` seção.

> **Warning**
>
> A adição de uma etapa inválida bloqueia o fluxo. Por exemplo, se você adicionar uma etapa de métrica de disco, mas a instância não tiver um disco, todas as etapas do fluxo serão bloqueadas.

Você pode enviar o mesmo arquivo de log a mais de um destino. Por exemplo, para enviar o log do aplicativo a dois destinos diferentes que você definiu na seção `CloudWatchLogs`, adicione `ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)` à seção `Flows`.

Para configurar o controle de fluxo

1. No arquivo `AWS.EC2.Windows.CloudWatch.json`, localize a seção `Flows`.

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

2. Em `Flows`, adicione cada tipo de dados os quais serão feito upload (por exemplo, `ApplicationEventLog`) e seu destino (por exemplo, `CloudWatchLogs`).

Agora você concluiu a edição do arquivo JSON. Você o usa em uma etapa posterior.

Inicie o agente

Para permitir que uma instância do Amazon EC2 executando o Windows Server 2012 ou o Windows Server 2008 envie CloudWatch registros para o Logs, use o serviço EC2 Config (. `EC2Config.exe`) Sua instância deve ter o EC2 Config 4.0 ou posterior, e você pode usar esse procedimento.

Para configurar CloudWatch usando o EC2 Config 4.x

1. Verifique a codificação do arquivo `AWS.EC2.Windows.CloudWatch.json` editado anteriormente neste procedimento. Só há suporte para a codificação UTF-8 sem BOM. Em seguida, salve o arquivo na pasta a seguir na instância do Windows Server 2008 - 2012 R2: `C:\Program Files\Amazon\SSM\Plugins\awsCloudWatch\`.
2. Inicie ou reinicie o agente SSM (`AmazonSSMAgent.exe`) usando o painel de controle dos Serviços do Windows ou usando o seguinte PowerShell comando:

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

Depois que o agente SSM é reiniciado, ele detecta o arquivo de configuração e configura a instância para integração. CloudWatch Se você alterar os parâmetros e as configurações no arquivo de configuração local, será necessário reiniciar o SSM Agent para que as alterações sejam efetivadas. Para desativar a CloudWatch integração na instância, `IsEnabled false` altere e salve suas alterações no arquivo de configuração.

Relate o status do agente do CloudWatch Logs

Use o procedimento a seguir para relatar o status do agente CloudWatch Logs na sua instância do EC2.

Para relatar o status do agente

1. Conecte-se à sua instância do EC2. Para obter mais informações, consulte Conectar-se à sua instância no Manual do usuário do Amazon EC2.

   Para obter mais informações sobre problemas de conexão, consulte Solução de problemas ao conectar-se à sua instância no Manual do usuário do Amazon EC2
2. Em um prompt de comando, digite o seguinte comando:

   ```
   sudo service awslogs status
   ```

   Se você está executando o Amazon Linux 2, digite o seguinte comando:

   ```
   sudo service awslogsd status
   ```

3. Verifique se há erros, avisos ou problemas com o agente var/log/awslogsLogs no arquivo/ CloudWatch .log.

Inicie o agente CloudWatch Logs

Se o agente CloudWatch Logs na sua instância do EC2 não foi iniciado automaticamente após a instalação, ou se você interrompeu o agente, você pode usar o procedimento a seguir para iniciar o agente.

Para iniciar o agente da

1. Conecte-se à sua instância do EC2. Para obter mais informações, consulte Conectar-se à sua instância no Manual do usuário do Amazon EC2.

   Para obter mais informações sobre problemas de conexão, consulte Solução de problemas de conexão à sua instância no Manual do usuário do Amazon EC2.
2. Em um prompt de comando, digite o seguinte comando:

   ```
   sudo service awslogs start
   ```

   Se você está executando o Amazon Linux 2, digite o seguinte comando:

   ```
   sudo service awslogsd start
   ```

Pare o agente CloudWatch de registros

Use o procedimento a seguir para interromper o agente CloudWatch Logs na sua instância do EC2.

Para interromper o agente da

1. Conecte-se à sua instância do EC2. Para obter mais informações, consulte Conectar-se à sua instância no Manual do usuário do Amazon EC2.

   Para obter mais informações sobre problemas de conexão, consulte Solução de problemas de conexão à sua instância no Manual do usuário do Amazon EC2.
2. Em um prompt de comando, digite o seguinte comando:

```
sudo service awslogs stop
```

Se você está executando o Amazon Linux 2, digite o seguinte comando:

```
sudo service awslogsd stop
```

CloudWatch Referência do agente de registros

> **Important**
>
> Esta seção é uma referência para quem usa o agente Logs obsoleto. CloudWatch Se você estiver usando o Instance Metadata Service Version 2 (IMDSv2), deverá usar o novo agente unificado. CloudWatch No entanto, mesmo que você não esteja usando o IMDSv2, é altamente recomendável usar o CloudWatch agente unificado mais recente em vez do agente de registros obsoleto. CloudWatch Para obter informações sobre o CloudWatch agente unificado mais recente, consulte Coleta de métricas e registros da instância do Amazon EC2 e de servidores locais com o agente. CloudWatch Para obter informações sobre a migração do agente de CloudWatch registros obsoleto para o agente unificado, crie o arquivo de configuração do CloudWatch agente com o assistente.

O agente CloudWatch Logs fornece uma forma automatizada de enviar dados de log para CloudWatch Logs a partir de instâncias do Amazon EC2. O agente inclui os seguintes componentes:

- Um plug-in para o AWS CLI que envia dados de registro para CloudWatch o Logs.
- Um script (daemon) que inicia o processo de envio de dados para o Logs. CloudWatch
- Um trabalho cron que garante que o daemon esteja sempre em execução.

Arquivo de configuração do agente

O arquivo de configuração do agente do CloudWatch Logs descreve as informações necessárias para o agente do CloudWatch Logs. A seção [general] do arquivo de configuração do agente define configurações comuns que se aplicam a todos os streams de log. A seção [logstream] define as informações necessárias para enviar um arquivo local para um stream de logs remoto. Você pode ter mais de uma seção [logstream], mas cada uma deve ter um nome exclusivo dentro do arquivo de

configuração, por exemplo, [logstream1], [logstream2] e assim por diante. O valor [logstream] junto com a primeira linha de dados no arquivo de log definem a identidade do arquivo de 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

Especifica onde o arquivo de estado está armazenado.

logging_config_file

(Opcional) Especifica a localização do arquivo de configuração de log do agente. Se você não especifica um arquivo de configuração de log do agente aqui, o arquivo padrão awslogs.conf é usado. A localização do arquivo padrão é `/var/awslogs/etc/awslogs.conf` se você instalou o agente com um script, e `/etc/awslogs/awslogs.conf` se você instalou o agente com RPM. O arquivo está no formato de arquivo de configuração do Python (https://docs.python.org/2/library/logging.config.html#logging -config-fileformat). Registradores com os seguintes nomes podem ser personalizados.

```
cwlogs.push
cwlogs.push.reader
cwlogs.push.publisher
```

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

O exemplo a seguir altera o nível de leitor e editor para AVISO enquanto o valor padrão é 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

Quando definido como verdadeiro (padrão), ativa a codificação de conteúdo gzip http para enviar cargas comprimidas para o Logs. CloudWatch Isso diminui o uso da CPU, diminui NetworkOut e diminui a latência de colocação. Para desativar esse recurso, adicione use_gzip_http_content_encoding = false à seção [general] do arquivo de configuração do agente do Logs e reinicie o CloudWatch agente.

> **Note**
>
> Essa configuração só está disponível no awscli-cwlogs versão 1.3.3 e posterior.

log_group_name

Especifica o grupo de logs de destino. Um grupo de logs é criado automaticamente, caso ele ainda não exista. Os nomes de grupos de log podem ter entre 1 e 512 caracteres. Os caracteres permitidos incluem a-z A-Z, 0-9, '_' (sublinhado), '-' (hífen), '/' (barra frontal) e '.' (período).

log_stream_name

Especifica o stream de logs de destino. Você pode usar uma cadeia de caracteres literal ou as variáveis predefinidas ({instance_id}, {hostname}, {ip_address}) ou uma combinação de ambas, para definir um nome de stream de log. Um stream de logs é criado automaticamente, caso ele ainda não exista.

datetime_format

Especifica como o carimbo de data e hora é extraído de logs. O timestamp é usado para recuperar eventos de log e gerar métricas. A hora atual será usada para todos os eventos de log se datetime_format não for fornecido. Se o valor datetime_format fornecido for inválido para uma determinada mensagem de log, o carimbo de data e hora do último evento de log com um carimbo de data/hora analisado com êxito será usado. Se não existir nenhum evento de log anterior, a hora atual será usada.

Os códigos datetime_format comuns estão listados a seguir. Você também pode usar qualquer código datetime_format compatíveis com Python, datetime.strptime(). O desvio de fuso horário

(%z) também é compatível, embora não seja suportado até o python 3.2, [+ -] HHMM sem dois-pontos (:). Para obter mais informações, consulte strftime() and strptime() Behavior.

%y: ano sem século preenchido como um número decimal preenchido com zeros. 00, 01, ..., 99

%Y: ano com século como número decimal.1970, 1988 2001, 2013

%b: mês como nome abreviado do local. Jan, Fev, ..., Dez (en_US);

%B: mês como nome completo do local. Janeiro, fevereiro,..., dezembro (en_US);

%m: mês como um número decimal preenchido com zeros. 01, 02, ..., 12

%d: dia do mês como um número decimal preenchido com zeros. 01, 02, ..., 31

%H: hora (formato de 24 horas) como um número decimal preenchido com zeros. 00, 01, ..., 23

%I: hora (formato de 12 horas) como um número decimal preenchido com zeros. 01, 02, ..., 12

%p: equivalente de local de AM ou PM.

%M: minuto como um número decimal preenchido com zeros. 00, 01, ..., 59

%S: segundo como um número decimal preenchido com zeros. 00, 01, ..., 59

%f: microssegundo como um número decimal preenchido com zeros à esquerda. 000000, ..., 999999

%z: deslocamento de UTC na forma+HHMM ou -HHMM. +0000, -0400, +1030

Exemplo de formatos:

```
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

Especifica o fuso horário do carimbo de data e hora do evento de log. Os dois valores aceitos são UTC e LOCAL. O padrão é LOCAL, que será usado se o fuso horário não puder ser considerado com base em datetime_format.

arquivo

Especifica os arquivos de log que você deseja enviar para o CloudWatch Logs. O arquivo pode apontar para um arquivo específico ou vários arquivos (usando curingas como/var/log/system.log *). Somente o arquivo mais recente é enviado para o CloudWatch Logs com base no horário de modificação do arquivo. Recomendamos usar caracteres curinga para especificar uma série de arquivos do mesmo tipo, como access_log.2014-06-01-01, access_log.2014-06-01-02, etc., mas não vários tipos de arquivos, como access_log_80 e access_log_443. Para especificar vários tipos de arquivos, adicione outra entrada de stream de log ao arquivo de configuração para que cada tipo de arquivo de log vá para um stream de log diferente. Arquivos compactados não têm suporte.

file_fingerprint_lines

Especifica o intervalo de linhas para identificar um arquivo. Os valores válidos são um ou dois números delimitados por traço, como "1", "2-5". O valor padrão é "1" para que a primeira linha seja usada para calcular a impressão digital. As linhas de impressão digital não são enviadas para o CloudWatch Logs, a menos que todas as linhas especificadas estejam disponíveis.

multi_line_start_pattern

Especifica o padrão para identificar o início de uma mensagem de log. Uma mensagem de log é feita de uma linha em conformidade com o padrão e as seguintes linhas que não correspondem ao padrão. Os valores válidos são expressão regular ou {datetime_format}. Ao usar {datetime_format}, a opção datetime_format deve ser especificada. O valor padrão é "^ [^\s]" para que qualquer linha que comece com um caractere diferente de espaço feche a mensagem de log anterior e inicie uma nova mensagem de log.

initial_position

Especifica onde começar a ler dados (start_of_file ou end_of_file). O padrão é start_of_file. É usado somente se não há estado persistente para esse stream de logs.

encoding

Especifica a codificação do arquivo de log para que o arquivo possa ser lido corretamente. O padrão é utf_8. As codificações suportadas pelo codecs.decode () Python podem ser usadas aqui.

> **Warning**
>
> A especificação incorreta da codificação pode causar a perda de dados porque os caracteres que não podem ser decodificados são substituídos por algum outro caractere.

Veja a seguir algumas codificações comuns:

```
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

Especifica o intervalo de tempo para o processamento em lote de eventos de log. O valor mínimo é 5000 ms e valor padrão é 5000 ms.

batch_count

Especifica o número máximo de eventos de log em um lote, até 10.000. O valor padrão é 10000.

batch_size

Especifica o tamanho máximo de eventos de log em um lote, em bytes, até 1.048.576 bytes. O valor de padrão é de 1048576 bytes. Esse tamanho é calculado como a soma de todas as mensagens de eventos recebidas UTF-8, mais 26 bytes para cada evento de log.

Usando o agente CloudWatch Logs com proxies HTTP

Você pode usar o agente CloudWatch Logs com proxies HTTP.

> **Note**
>
> Os proxies HTTP são aceitos no awslogs-agent-setup.py versão 1.3.8 ou posterior.

Para usar o agente CloudWatch Logs com proxies HTTP

1. Execute um destes procedimentos:

   a. Para uma nova instalação do agente CloudWatch Logs, execute os seguintes comandos:

   ```
   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
   ```

   Para manter o acesso ao serviço de metadados do Amazon EC2 em instâncias do EC2, use --no-proxy 169.254.169.254 (recomendado). Para obter mais informações, consulte Metadados de instância e dados do usuário no Manual do usuário do Amazon EC2.

   Nos valores de `http-proxy` e `https-proxy`, você especifica a URL completa.

   b. Para uma instalação existente do agente CloudWatch Logs, edite/var/awslogs/etc/proxy.conf e adicione seus proxies:

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

2. Reinicie o agente para que as alterações sejam aplicadas:

   ```
   sudo service awslogs restart
   ```

   Se você está usando o Amazon Linux 2, use o seguinte comando para reiniciar o agente:

   ```
   sudo service awslogsd restart
   ```

Compartimentalizando os arquivos de configuração do agente CloudWatch Logs

Se você estiver usando o awslogs-agent-setup.py versão 1.3.8 ou posterior com awscli-cwlogs 1.3.3 ou posterior, poderá importar diferentes configurações de stream para vários componentes, independentemente um do outro, criando arquivos de configuração adicionais no diretório///. var/awslogs etc/config Quando o agente CloudWatch Logs é iniciado, ele inclui todas as configurações de stream nesses arquivos de configuração adicionais. As propriedades de configuração na seção [general] devem ser definidas no arquivo de configuração principal (/var/awslogs/etc/awslogs.conf) e são ignoradas em qualquer arquivo de configuração adicional encontrado em//var/awslogs/. etc/config

Se você não tiver um diretório//var/awslogsetc/config/porque instalou o agente com rpm, você pode usar o diretório/etc/awslogs/config/ em vez disso.

Reinicie o agente para que as alterações sejam aplicadas:

```
sudo service awslogs restart
```

Se você está usando o Amazon Linux 2, use o seguinte comando para reiniciar o agente:

```
sudo service awslogsd restart
```

CloudWatch Perguntas frequentes sobre o agente de registros

Quais tipos de rotações de arquivos são compatíveis?

Os mecanismos de rotação de arquivos a seguir são suportados:

- Renomear arquivos de log existentes com um sufixo numérico e, em seguida, recriar o arquivo de log original em branco. Por exemplo,/var/log/syslog.log é renomeado para/var/log/syslog.log.1. Se/var/log/syslog.log.1 já existir de uma rotação anterior, ela será renomeada para//syslog.log.2. var/log
- Truncando o arquivo de log original após a criação de uma cópia. Por exemplo,/var/log/syslog.log é copiado para/var/log/syslog.log.1 e//syslog.log é truncado. var/log Pode haver perda de dados nesse caso, então, tome cuidado ao usar esse mecanismo de rotação de arquivos.
- Criando um novo arquivo com um padrão comum como o antigo. Por exemplo,/var/log/syslog.log.2014-01-01 permanece e//syslog.log.2014-01-02 é criado. var/log

A impressão digital (ID de origem) do arquivo é calculada aplicando hash à chave de stream de logs e à primeira linha do conteúdo do arquivo. Para substituir esse comportamento, a opção file_fingerprint_lines pode ser usada. Quando a rotação de arquivos acontece, o novo arquivo deve ter um novo conteúdo e o arquivo antigo não deve ter conteúdo anexado; o agente envia o novo arquivo depois de ler o arquivo antigo.

Como posso determinar qual versão do agente estou usando?

Se você usou um script de configuração para instalar o agente do CloudWatch Logs, poderá usar var/awslogs//bin/awslogs-version.sh para verificar qual versão do agente está usando. Ele imprime a versão do agente e suas principais dependências. Se você usou o yum para instalar o agente CloudWatch Logs, você pode usar “yum info awslogs” e “yum info aws-cli-plugin-cloudwatch-logs” para verificar a versão do agente e do plug-in do Logs. CloudWatch

Como ficam as entradas de log convertidas em eventos de log?

Os eventos de log contêm duas propriedades: o carimbo de data e hora de quando o evento ocorreu, e a mensagem de log bruta. Por padrão, qualquer linha que comece com um caractere diferente de espaço fecha a mensagem de log anterior, se houver, e inicia uma nova mensagem de log. Para substituir este comportamento, o multi_line_start_pattern pode ser usado, e qualquer linha que esteja em conformidade com o padrão inicia uma nova mensagem de log. O padrão pode ser qualquer regex ou "{datetime_format}". Por exemplo, se a primeira linha de cada mensagem de registro contiver um timestamp como T13:13:01Z '2014-01-02', o multi_line_start_pattern poderá ser definido como '\ d {4} -\ d {2} -\ d {2} T\ d {2}:\ d {2}:\ d {2} Z'. Para simplificar a configuração, a variável ‘{datetime_format}' poderá ser usada se a variável datetime_format option for especificada. Para o mesmo exemplo, se datetime_format estiver definido como '%Y-%m-%dT%H:%M:%S%z', então multi_line_start_pattern poderá ser simplesmente '{datetime_format}'.

A hora atual será usada para todos os eventos de log se datetime_format não for fornecido. Se datetime_format fornecido for inválido para uma determinada mensagem de log, será usado o carimbo de data e hora do último evento de log com um carimbo de data/hora analisado com êxito. Se não existir nenhum evento de log anterior, a hora atual será usada. Uma mensagem de aviso é registrada quando um evento de log retorna para a hora atual ou a hora do evento de log anterior.

Os carimbos de data e hora são usados para recuperar eventos de log e gerar métricas, portanto, se você especificar o formato incorreto, os eventos de log poderão se tornar não recuperáveis e gerar métricas incorretas.

Como os eventos de log são armazenadas em lote?

Um lote fica cheio e é publicado quando qualquer uma das seguintes condições é atendida:

1. O tempo de buffer_duration terminou desde que o primeiro evento de log foi adicionado.
2. Menos do que batch_size de eventos de log foram acumulados, mas adicionar o novo evento de log excede o batch_size.
3. O número de eventos de log atingiu batch_count.
4. Os eventos de log do lote não abrangem mais do que 24 horas, mas adicionar o novo evento de log excede a restrição de 24 horas.

O que faz com que entradas de log, eventos de log ou lotes sejam ignorados ou truncados?

Para acompanhar a restrição da operação `PutLogEvents`, os seguintes problemas podem fazer com que um evento de log ou lote seja ignorado.

> **Note**
>
> O agente CloudWatch Logs grava um aviso em seu registro quando os dados são ignorados.

1. Se o tamanho de um evento de log exceder 256 KB, ele será ignorado completamente.
2. Se o carimbo de data e hora do evento de log for de mais do que 2 horas no futuro, o evento de log será ignorado.
3. Se o carimbo de data e hora do evento de log for de mais do que 14 dias no passado, o evento de log será ignorado.
4. Se qualquer evento de log for mais antigo do que o período de retenção do grupo de logs, o lote inteiro será ignorado.
5. Se o lote de eventos de log em uma única solicitação `PutLogEvents` abranger mais de 24 horas, ocorrerá falha na operação `PutLogEvents`.

A interrupção do agente causa dados loss/duplicates?

Não, desde que o arquivo de estado esteja disponível e nenhuma rotação de arquivos tenha ocorrido desde a última execução. O agente de CloudWatch registros pode começar de onde parou e continuar enviando os dados de registro.

Posso apontar arquivos de log diferentes do mesmo host ou de hosts diferentes para o mesmo stream de logs?

A configuração de várias fontes de log para enviar dados a um único stream de logs não é suportada.

Quais chamadas de API o agente pode fazer (ou quais ações devo adicionar à minha política do IAM)?

O agente do CloudWatch Logs exige permissão para realizar `CreateLogGroupCreateLogStream`,`DescribeLogStreams`,`DescribeLogGroups`, `PutLogEvents` e `PutRetentionPolicy` ações. Se você estiver usando o agente mais recente, o `DescribeLogStreams` não será necessário. Consulte o exemplo de política do IAM a seguir.

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:*:*:*"
    ]
  }
 ]
}
```

Não quero que o agente de CloudWatch registros crie grupos ou fluxos de registros automaticamente. Como posso impedir que o agente recrie grupos e fluxos de log?

Em sua política do IAM, é possível restringir o agente apenas às seguintes operações: `DescribeLogStreams`, `PutLogEvents`.

Antes de revogar as permissões `CreateLogGroup` e `CreateLogStream` do agente, certifique-se de criar os grupos de log e fluxos de log que você deseja que o agente use. O agente de logs não pode criar fluxos de log em um grupo de log que você criou, a menos que ele tenha as permissões `CreateLogGroup` e `CreateLogStream`.

Quais logs devo procurar ao solucionar problemas?

O log de instalação do agente está em `/var/log/awslogs-agent-setup.log` e o log do agente está em `/var/log/awslogs.log`.

Início rápido: use CloudFormation para começar a usar o CloudWatch Logs

AWS CloudFormation permite que você descreva e provisione seus AWS recursos no formato JSON. As vantagens desse método incluem ser capaz de gerenciar uma coleção de AWS recursos como uma única unidade e replicar facilmente seus AWS recursos em todas as regiões.

Ao provisionar AWS usando CloudFormation, você cria modelos que descrevem os AWS recursos a serem usados. O exemplo a seguir é um trecho de modelo que cria um grupo de logs e um filtro de métrica que conta as ocorrências de 404 e envia essa contagem para o grupo de logs.

```
"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"
```

```
            }
        ]
    }
}
```

Este é um exemplo simples. Você pode configurar implantações muito mais avançadas do CloudWatch Logs usando. CloudFormation Para obter mais informações sobre exemplos de modelos, consulte Amazon CloudWatch Logs Template Snippets no Guia do AWS CloudFormation usuário. Para obter mais informações sobre como começar, consulte Conceitos básicos do AWS CloudFormation no Manual do usuário do AWS CloudFormation .

Ingestão de registros por meio de endpoints HTTP

O Amazon CloudWatch Logs fornece endpoints HTTP que permitem que você envie registros diretamente para o CloudWatch Logs usando solicitações HTTP POST simples. Esses endpoints oferecem suporte à autenticação SigV4 e ao token do portador.

> **Important**
>
> Recomendamos usar a autenticação SigV4 para todas as cargas de trabalho de produção em que a integração do AWS SDK é possível. O SigV4 usa credenciais de curto prazo e fornece a postura de segurança mais forte. A autenticação do token do portador (chave de API) é destinada a cenários em que o SigV4 não é viável, como encaminhadores de log de terceiros que não oferecem suporte à integração do SDK. AWS Para obter mais informações, consulte Alternativas às chaves de acesso de longo prazo no Guia do usuário do IAM.

CloudWatch O Logs é compatível com os seguintes endpoints de ingestão de HTTP:

Endpoint	Path	Content-Type	Formato
OpenTelemetry Logs	`/v1/logs`	`application/json` ou `application/x-protobuf`	OTLP JSON ou Protobuf
HLC Logs	`/services/collector/event`	`application/json`	Formato HLC
ND-JSON Logs	`/ingest/bulk`	`application/json` ou `application/x-ndjson`	JSON delimitado por Newline
Structured JSON Logs	`/ingest/json`	`application/json`	Objeto ou matriz JSON

Comportamento comum

Todos os endpoints de ingestão de HTTP compartilham o seguinte comportamento:

Autenticação

Todos os endpoints oferecem suporte à autenticação SigV4 e ao token do portador:

- SigV4 (recomendado) — AWS Assinatura padrão versão 4. Use o SigV4 sempre que seu aplicativo ou infraestrutura suportar o AWS SDK ou puder assinar solicitações. O SigV4 usa credenciais de curto prazo e é o método de autenticação mais seguro.
- Token do portador — Use o `Authorization: Bearer <ACWL token>` cabeçalho.
  - O token deve ser um token portador de ACWL válido. Para obter instruções de configuração, consultethe section called “Autenticação do token do portador”.
  - Requer as permissões `logs:CallWithBearerToken` do IAM `logs:PutLogEvents` e do IAM.

Grupo de registros e fluxo de registros

- Fornecido por meio de cabeçalhos: `x-aws-log-group` e `x-aws-log-stream`
- `?logGroup=<name>&logStream=<name>`Os parâmetros de consulta também são suportados em todos os endpoints, exceto OTLP.
- Você não pode usar parâmetros de consulta e cabeçalhos para o mesmo parâmetro.
- Tanto o grupo de registros quanto o fluxo de registros são obrigatórios.

Resposta

- Sucesso: `HTTP 200` com corpo `{}`
- Erros de validação: `HTTP 400`
- Falhas de autenticação: `HTTP 401`

Configurando a autenticação do token do portador

Antes de enviar registros usando a autenticação de token do portador com qualquer um dos endpoints de ingestão HTTP, você precisa:

- Crie um usuário do IAM com permissões CloudWatch de registros
- Gere credenciais específicas do serviço (token do portador)
- Crie um grupo de registros e um fluxo de registros

- Ativar a autenticação do token do portador no grupo de registros

> **Important**
>
> Recomendamos usar a autenticação SigV4 com credenciais de curto prazo para todas as cargas de trabalho, sempre que possível. O SigV4 fornece a postura de segurança mais forte. Restrinja o uso de chaves de API (tokens portadores) a cenários em que a autenticação de curto prazo baseada em credenciais não seja viável. Quando estiver pronto para incorporar CloudWatch registros em aplicativos com maiores requisitos de segurança, você deve mudar para credenciais de curto prazo. Para obter mais informações, consulte Alternativas às chaves de acesso de longo prazo no Guia do usuário do IAM.

Opção 1: início rápido usando o AWS console

O AWS Management Console fornece um fluxo de trabalho simplificado para gerar chaves de API para acesso ao endpoint HTTP.

Para configurar o acesso ao endpoint HTTP usando o console

1. Faça login no AWS Management Console.
2. Navegue até CloudWatch> Configurações > Registros.
3. Na seção Chaves de API, escolha Gerar chave de API.
4. Em Validade da chave de API, faça uma das seguintes opções:
   - Selecione uma duração de expiração da chave de API de 1, 5, 30, 90 ou 365 dias.
   - Escolha Duração personalizada para especificar uma data de validade personalizada para a chave de API.
   - Selecione Nunca expira (não recomendado).
5. Escolha Gerar chave de API.

   O console automaticamente:

   - Cria um novo usuário do IAM com as permissões apropriadas
   - Anexa a política gerenciada do CloudWatchLogsAPIKeyAccess (inclusões `logs:PutLogEvents` e `logs:CallWithBearerToken` permissões)

- Gera credenciais específicas do serviço (chave de API)

6. Copie e salve com segurança as credenciais exibidas:

   - ID da chave de API (ID de credencial específica do serviço)
   - Segredo da chave da API (token do portador)

   > **Important**
   >
   > Salve o segredo da chave de API imediatamente. Ela não poderá ser recuperada posteriormente. Se você a perder, precisará gerar uma nova chave de API.

7. Crie o grupo de registros e o fluxo de registros em que seus registros serão armazenados:

```
# 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. Ative a autenticação do token do portador no grupo de registros:

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

   Verifique a configuração:

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

Permissões incluídas: o usuário do IAM criado automaticamente terá as seguintes permissões:

- `logs:PutLogEvents`— Enviar eventos de registro para CloudWatch Logs
- `logs:CallWithBearerToken`— Autenticar usando o token do portador
- `kms:Describe*`,`kms:GenerateDataKey*`, `kms:Decrypt` — Acesse grupos de registros criptografados pelo KMS (com condição restrita ao serviço de registros)

Opção 2: configuração manual

Se você preferir ter mais controle sobre a configuração do IAM ou precisar personalizar as permissões, você pode configurar o acesso ao endpoint HTTP manualmente.

Etapa 1: criar um usuário do IAM

Crie um usuário do IAM que será usado para ingestão de registros:

1. Faça login no AWS Management Console e navegue até o IAM.
2. No painel de navegação à esquerda, escolha Usuários.
3. Selecione Criar usuário.
4. Insira um nome de usuário (por exemplo,`cloudwatch-logs-hlc-user`).
5. Escolha Próximo.
6. Anexe uma das seguintes políticas do IAM:

   Opção A: usar a política gerenciada (recomendada)

   Anexe a política gerenciada do CloudWatchLogsAPIKeyAccess.

   Opção B: criar uma política personalizada

   Crie e anexe a seguinte política do IAM:

```
{
    "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. Escolha Avançar e, em seguida, Criar usuário.

> **Note**
>
> As permissões do KMS são necessárias se você planeja enviar registros para grupos de registros criptografados pelo KMS. A condição restringe o acesso ao KMS somente às chaves usadas por meio do serviço de CloudWatch registros.

Etapa 2: gerar credenciais específicas do serviço (chave de API)

Gere a chave da API CloudWatch Logs usando a CreateServiceSpecificCredentialAPI. Você também pode usar o comando create-service-specific-credentialCLI. Para a idade da credencial, você pode especificar um valor entre 1 e 36600 dias. Se você não especificar o prazo de validade de uma credencial, a chave da API não expirará.

Para gerar uma chave de API com uma expiração de 30 dias:

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

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

A resposta é um ServiceSpecificCredentialobjeto. O `ServiceCredentialSecret` valor é sua chave da API CloudWatch Logs (token do portador).

> **Important**
>
> Armazene o valor `ServiceCredentialSecret` em segurança, pois não será possível recuperá-lo mais tarde. Se você a perder, precisará gerar uma nova chave de API.

Etapa 3: criar um grupo de registros e um fluxo de registros

Crie o grupo de registros e o fluxo de registros em que seus registros serão armazenados:

```
# 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
```

Etapa 4: ativar a autenticação do token do portador

Ative a autenticação do token do portador no grupo de registros:

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

Verifique a configuração:

```
aws logs describe-log-groups \
    --log-group-name-prefix /aws/hlc-logs/my-application \
```

```
    --region us-east-1
```

Controle as permissões para gerar e usar CloudWatch as chaves da API Logs

A geração e o uso das chaves da API CloudWatch Logs são controlados por ações e chaves de condição nos serviços do CloudWatch Logs e do IAM.

Controle da geração de chaves da API CloudWatch Logs

A CreateServiceSpecificCredential ação iam: controla a geração de uma chave específica do serviço (como uma chave da API CloudWatch Logs). Você pode definir o escopo dessa ação para usuários do IAM como um recurso para limitar os usuários para os quais uma chave pode ser gerada.

É possível usar as seguintes chaves de condição para impor condições à permissão para a ação `iam:CreateServiceSpecificCredential`:

- iam: ServiceSpecificCredentialAgeDays — Permite especificar, na condição, o tempo de expiração da chave em dias. Por exemplo, é possível usar essa chave de condição para permitir somente a criação de chaves de API que expirem em noventa dias.
- iam: ServiceSpecificCredentialServiceName — Permite especificar, na condição, o nome de um serviço. Por exemplo, você pode usar essa chave de condição para permitir somente a criação de chaves de API para CloudWatch Logs e não para outros serviços.

Controle do uso das chaves da API CloudWatch Logs

A `logs:CallWithBearerToken` ação controla o uso de uma chave da API CloudWatch Logs. Para evitar que uma identidade use as chaves da API CloudWatch Logs, anexe uma política que negue a `logs:CallWithBearerToken` ação ao usuário do IAM associado à chave.

Exemplo de política

Impedir que uma identidade gere e use CloudWatch as chaves da API Logs

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

```
            "Effect": "Deny",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "logs:CallWithBearerToken"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Warning**
>
> Essa política impedirá a criação de credenciais para todos os AWS serviços que oferecem suporte à criação de credenciais específicas do serviço. Para ter mais informações, consulte Credenciais específicas do serviço para usuários do IAM.

Impedir que uma identidade use CloudWatch as chaves da API Logs

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

Permita a criação de chaves de CloudWatch registros somente se elas expirarem em 90 dias

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

Chaves de API rotativas

A rotação regular de suas chaves de API reduz o risco de acesso não autorizado. Recomendamos estabelecer um cronograma de rotação que se alinhe às políticas de segurança da sua organização.

Processo de rotação

Para alternar uma chave de API sem interromper a entrega do registro, siga este procedimento:

1. Crie uma nova credencial (secundária) para o usuário do IAM:

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

2. (Opcional) Armazene a nova credencial AWS Secrets Manager para recuperação segura e rotação automatizada.
3. Importe a nova credencial para o portal do seu fornecedor ou atualize a configuração do seu aplicativo para usar a nova chave de API.
4. Defina a credencial original como inativa:

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

5. Verifique se a entrega de registros não é afetada monitorando a `IncomingBytes` métrica do seu grupo de registros em CloudWatch. Para obter mais informações, consulte Monitoramento com CloudWatch métricas.
6. Depois de confirmar a entrega bem-sucedida com a nova chave, exclua a credencial anterior:

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

Monitorando a expiração da chave

Para verificar a data de criação e o status de suas chaves de API existentes, use o list-service-specific-credentialscomando:

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

A resposta inclui `CreateDate` e `Status` para cada credencial. Use essas informações para identificar chaves que estão prestes a expirar ou que estão ativas há mais tempo do que sua política de rotação permite.

Respondendo a uma chave de API comprometida

Se você suspeitar que uma chave de API foi comprometida, execute as seguintes etapas imediatamente:

1. Desative a chave imediatamente para evitar mais uso não autorizado:

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

2. Analise CloudTrail os registros para determinar o escopo do acesso não autorizado. Veja the section called “Registrando o uso da chave de API com CloudTrail” como habilitar a auditoria do uso da chave de API.
3. Crie uma chave de substituição seguindo o processo de rotação descrito emthe section called “Processo de rotação”.
4. Exclua a chave comprometida depois que a substituição estiver em vigor:

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

5. Anexe uma política de negação se precisar bloquear imediatamente todo o acesso ao token portador do usuário do IAM durante a investigação:

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

> **Note**
>
> Para realizar essas ações por meio da API, você deve se autenticar com AWS credenciais e não com uma chave da API CloudWatch Logs.

Você também pode usar as seguintes operações de API do IAM para gerenciar chaves comprometidas:

- ResetServiceSpecificCredential— Redefina a chave para gerar uma nova senha sem excluir a credencial. A chave não deve ter expirado.

Práticas recomendadas de segurança para chaves de API

Siga estas práticas recomendadas para proteger suas chaves da API CloudWatch Logs:

- Nunca incorpore chaves de API no código-fonte. Não codifique as chaves de API no código do aplicativo nem as comprometa com sistemas de controle de versão. Se uma chave for transferida acidentalmente para um repositório público, o escaneamento AWS automático pode sinalizá-la e você deve girar a chave imediatamente.
- Use um gerenciador de segredos. Armazene as chaves de API em AWS Secrets Manageruma solução de gerenciamento de segredos equivalente. Isso permite controle de acesso centralizado, registro de auditoria e rotação automatizada.
- Defina uma expiração para todas as chaves. Sempre especifique um `--credential-age-days` valor ao criar chaves de API. Para impor uma vida útil máxima da chave em sua organização,

use a chave de condição `iam:ServiceSpecificCredentialAgeDays` do IAM. Para ver um exemplo, consulte the section called “Permita a criação de chaves de CloudWatch registros somente se elas expirarem em 90 dias”.

- Aplique permissões com privilégios mínimos. Defina o escopo das permissões do usuário do IAM somente para os grupos de registros e as ações necessárias. Use a política de CloudWatchLogsAPIKeyacesso gerenciado como ponto de partida e restrinja ainda mais conforme necessário.
- Ative o CloudTrail registro. Audite o uso da chave da API ativando eventos de CloudTrail dados para`AWS::Logs::LogGroupAuthorization`. Consulte the section called “Registrando o uso da chave de API com CloudTrail”.
- Monitore com o IAM Access Analyzer. Use o IAM Access Analyzer para identificar credenciais não utilizadas e políticas excessivamente permissivas associadas aos usuários da chave de API do IAM.
- Gire as teclas regularmente. Estabeleça um cronograma de rotação e siga o processo descrito emthe section called “Chaves de API rotativas”.

Registrando o uso da chave de API com CloudTrail

Você pode usar AWS CloudTrail para registrar eventos de dados para o uso da chave da API CloudWatch Logs. CloudWatch O Logs emite eventos de `AWS::Logs::LogGroupAuthorization` dados para `CallWithBearerToken` chamadas, permitindo que você audite quando e como as chaves de API são usadas para enviar registros.

Para ativar o CloudTrail registro para o uso da chave da API CloudWatch Logs:

> **Note**
>
> O bucket do S3 que você especifica para a trilha deve ter uma política de bucket que permita CloudTrail gravar arquivos de log nele. Para obter mais informações, consulte a política de bucket do Amazon S3 para. CloudTrail

1. Crie uma trilha:

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

```
    --region us-east-1
```

2. Configure seletores de eventos avançados para capturar eventos de autorização de grupos de CloudWatch registros de registros:

```
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. Inicie o registro de trilhas:

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

Enviando registros usando o endpoint OTLP (OpenTelemetry Logs)

O endpoint do OpenTelemetry Logs (`/v1/logs`) aceita dados de log do OpenTelemetry Protocolo (OTLP) na codificação JSON ou Protobuf. Para obter informações detalhadas sobre o endpoint OTLP, incluindo configuração e uso, consulte Enviar métricas e rastreamentos para com CloudWatch . OpenTelemetry

Se você estiver usando a autenticação do token do portador, conclua as etapas de configuração the section called “Autenticação do token do portador” antes de continuar.

Formato de solicitação

- Método: `POST`
- Tipo de conteúdo: ou `application/json` `application/x-protobuf`
- Grupo de registros: somente `x-aws-log-group` cabeçalho (parâmetro de consulta não suportado)

- Fluxo de log: `x-aws-log-stream` cabeçalho

Exemplo de solicitação

```
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" }
                }
              ]
            }
          ]
```

```
        }
      ]
    }
  ]
}'
```

Respostas

Sucesso (todos os eventos são aceitos):

```
HTTP 200 OK
{}
```

Sucesso parcial (alguns eventos foram rejeitados):

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

Quando a solicitação Content-Type é`application/x-protobuf`, a resposta é retornada como uma mensagem `ExportLogsServiceResponse` protobuf serializada com os mesmos campos.

Comportamentos específicos do OTLP

Os comportamentos a seguir são específicos do endpoint OTLP e não estão presentes nos outros endpoints de ingestão HTTP:

- Cabeçalho Retry-After — Incluído nas respostas 503 e 429 para indicar quando o cliente deve tentar novamente.

Enviando registros usando o endpoint HLC (HLC Logs)

O endpoint do HLC Logs (`/services/collector/event`) é baseado no formato HTTP Log Collector (HLC).

Se você estiver usando a autenticação do token do portador, conclua as etapas de configuração the section called “Autenticação do token do portador” antes de continuar.

Modos de entrada

Cada evento é um objeto JSON com um `"event"` campo obrigatório. Campos de metadados opcionais:`"time"`,`"host"`,`"source"`,`"sourcetype"`,`"index"`.

Evento único:

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

Matriz de eventos JSON:

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

Eventos concatenados/agrupados (sem invólucro de matriz):

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

Campo do evento (obrigatório)

O `"event"` campo é obrigatório. Seu valor pode ser qualquer tipo de JSON:

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

Objetos sem um `"event"` campo são ignorados silenciosamente:

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

Campo de hora (opcional)

O `"time"` campo está em segundos de época (não em milissegundos), com decimal opcional para precisão de menos de um segundo.

Formato	Exemplo	Interpretada como
Float	`"time":1486683865.500`	148683865.500 ms
Inteiro	`"time":1486683865`	1486683865.000 ms
Corda (flutuação)	`"time":"1486683865.500"`	148683865.500 ms
Cadeia de caracteres (inteiro)	`"time":"1486683865"`	1486683865.000 ms
Missing (Ausente)	(sem campo de hora)	Hora atual do servidor
Inválido	`"time":"invalid"`	Hora atual do servidor

Content-Type

Só `application/json` é aceito.

Tipos de valores JSON aceitos

Tipo de nível superior	Comportamento
Objeto com `"event"`	Aceito
Objeto sem `"event"`	Ignorado
Matriz de objetos	Cada elemento processado individualmente
Objetos concatenados	Cada objeto processado individualmente
Primitivo (string, número, booleano, nulo)	Ignorado

Formato do endpoint

O URL do endpoint do HLC segue este formato:

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

Parâmetros necessários:

- `<region>`— AWS Região (por exemplo,`us-east-1`,`eu-west-1`)
- `logGroup`— Nome do grupo de registros codificado por URL
- `logStream`— Nome do fluxo de log codificado por URL

Parâmetros opcionais:

Opcionalmente, você pode associar seus eventos de log a uma `Service` entidade incluindo os seguintes parâmetros de consulta. Como os registros enviados pelo endpoint do HLC são telemetria personalizada, eles não são associados automaticamente a uma entidade. Ao fornecer esses parâmetros, o CloudWatch Logs cria uma entidade com `KeyAttributes.Type` set to `Service` e a associa aos seus eventos de registro. Isso permite que o recurso relacionado ao Explore correlacione esses registros com outras telemetrias (métricas, rastreamentos e registros) do mesmo serviço, facilitando a solução de problemas e o monitoramento de seus aplicativos em diferentes tipos de sinal. CloudWatch Para obter mais informações sobre entidades e telemetria relacionada, consulte Adicionar informações relacionadas à telemetria personalizada.

- `entityName`— O nome da entidade de serviço a ser associada aos eventos de log. Esse valor é armazenado como a entidade `KeyAttributes.Name` (por exemplo, `my-application` ou`api.myservice.com`).
- `entityEnvironment`— O ambiente em que o serviço está hospedado ou ao qual ele pertence. Esse valor é armazenado como a entidade `KeyAttributes.Environment` (por exemplo`production`,`ec2:default`, ou`eks:my-cluster/default`).

Formato de solicitação

Envie registros usando HTTP POST com os seguintes cabeçalhos e corpo:

Cabeçalhos:

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

Formato do corpo:

O corpo da solicitação deve estar no formato JSON com uma matriz de eventos:

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

Descrições de campo:

- `time`— Timestamp da época do Unix em segundos, com decimal opcional para precisão de menos de um segundo (opcional)
- `event`— A mensagem de registro ou os dados do evento (obrigatório)
- `host`— Nome do host ou identificador de origem (opcional)
- `source`— Identificador da fonte de log (opcional)

Campos personalizados adicionais podem ser incluídos conforme necessário.

Exemplo de solicitação

```
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"}'
```

Práticas recomendadas

eventos em lotes

Para melhor desempenho e eficiência:

- Agrupe vários eventos em uma única solicitação, quando possível
- Tamanho de lote recomendado: 10 a 100 eventos por solicitação
- Tamanho máximo da solicitação: 1 MB

Tratamento de erros

Implemente o tratamento adequado de erros em seu aplicativo. Códigos de status HTTP comuns:

- `200 OK`— Registros ingeridos com sucesso
- `400 Bad Request`— Formato ou parâmetros de solicitação inválidos
- `401 Unauthorized`— Token de portador inválido ou expirado
- `403 Forbidden`— Permissões insuficientes
- `404 Not Found`— O grupo de log ou stream não existe
- `429 Too Many Requests`— Limite de taxa excedido
- `500 Internal Server Error`— Erro de serviço (tente novamente com recuo exponencial)

Limitações

- Tamanho máximo do evento: 256 KB por evento
- Tamanho máximo da solicitação: 1 MB
- Máximo de eventos por solicitação: 10.000
- Os nomes dos grupos de registros devem seguir as convenções de nomenclatura de CloudWatch registros
- A autenticação do token do portador deve ser ativada no grupo de registros se a autenticação do token do portador for usada.

Enviando registros usando o endpoint NDJSON (ND-JSON Logs)

O endpoint ND-JSON Logs (`/ingest/bulk`) aceita registros no formato NDJSON (Newline Delimited JSON). Cada linha contém exatamente um valor JSON, separado por caracteres de nova linha.

Se você estiver usando a autenticação do token do portador, conclua as etapas de configuração the section called “Autenticação do token do portador” antes de continuar.

Formato de solicitação

Envie um valor JSON por linha, separado por `\n` (LF) ou `\r\n` (CRLF). As linhas vazias são ignoradas silenciosamente.

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

Ambos `application/json` `application/x-ndjson` são aceitos como o tipo de conteúdo.

Tipos de valores JSON aceitos

De acordo com a especificação NDJSON (RFC 8259), qualquer valor JSON válido é aceito em cada linha.

Objetos JSON (mais comuns):

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

Matrizes JSON (agrupadas em eventos individuais):

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

Essa única linha produz 2 eventos. Cada elemento da matriz se torna um evento de log separado.

Valores primitivos:

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

```
null
```

Cada primitivo se torna seu próprio evento com a data e hora atual do servidor.

Tipos mistos:

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

Todas as 4 linhas são aceitas como eventos válidos.

Conteúdo da linha	Comportamento
Objeto JSON	Aceito, carimbo de data/hora extraído, se presente
matriz JSON	Achatado — cada elemento se torna um evento separado
Matriz vazia `[]`	Aceito, produz 0 eventos
Cadeia de caracteres JSON	Aceito como mensagem do evento
Número JSON	Aceito como mensagem do evento
Booleano JSON	Aceito como mensagem do evento
JSON nulo	Aceito como mensagem do evento
JSON inválido	Ignorado (contado, o processamento continua)
Linha vazia	Ignorado (não contado como ignorado)

Campo de carimbo de data/hora

O `"timestamp"` campo está em milissegundos de época (não segundos).

Formato	Exemplo	Interpretada como
Numérico (millis)	`"timestamp":1771007942000`	1771007942000 ms
Missing (Ausente)	(sem campo de carimbo de data/hora)	Hora atual do servidor
Não numérico	`"timestamp":"invalid"`	Hora atual do servidor
Linha sem objeto	`"hello"`, `42`, `true`	Hora atual do servidor

Linhas inválidas

As linhas que não são JSON válidas são ignoradas e contadas silenciosamente. O processamento continua com a próxima linha.

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

Resultado: 2 eventos ingeridos, 1 ignorado. Retorna `HTTP 200`.

Se todas as linhas forem inválidas, retorna `HTTP 400` com`"All events were invalid"`.

Exemplo de solicitação

```
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"}'
```

Respostas

Sucesso (todos os eventos são aceitos):

```
HTTP 200 OK
{}
```

Sucesso parcial (alguns eventos foram rejeitados):

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

O `rejectedLogRecords` campo é o número total de eventos rejeitados. O `errorMessage` campo contém um detalhamento codificado em JSON por motivo de rejeição:

- `tooOldLogEventCount`— Eventos com carimbos de data/hora anteriores ao período de retenção
- `tooNewLogEventCount`— Eventos com timestamps muito distantes no futuro
- `expiredLogEventCount`— Eventos que expiraram durante o processamento

Práticas recomendadas

eventos em lotes

Para melhor desempenho e eficiência:

- Agrupe vários eventos em uma única solicitação, quando possível
- Tamanho de lote recomendado: 10 a 100 eventos por solicitação
- Tamanho máximo da solicitação: 1 MB

Tratamento de erros

Implemente o tratamento adequado de erros em seu aplicativo. Códigos de status HTTP comuns:

- `200 OK`— Registros ingeridos com sucesso
- `400 Bad Request`— Formato ou parâmetros de solicitação inválidos

- `401 Unauthorized`— Token de portador inválido ou expirado
- `403 Forbidden`— Permissões insuficientes
- `404 Not Found`— O grupo de log ou stream não existe
- `429 Too Many Requests`— Limite de taxa excedido
- `500 Internal Server Error`— Erro de serviço (tente novamente com recuo exponencial)

Limitações

- Tamanho máximo do evento: 256 KB por evento
- Tamanho máximo da solicitação: 1 MB
- Máximo de eventos por solicitação: 10.000
- Os nomes dos grupos de registros devem seguir as convenções de nomenclatura de CloudWatch registros
- A autenticação do token do portador deve ser ativada no grupo de registros se a autenticação do token do portador for usada.

Envio de registros usando o endpoint JSON estruturado (Structured JSON Logs)

O endpoint (`/ingest/json`) do Structured JSON Logs aceita JSON padrão — um único objeto JSON ou uma matriz JSON de objetos. Esse endpoint foi projetado para dados de log estruturados em que cada evento é um objeto JSON.

Se você estiver usando a autenticação do token do portador, conclua as etapas de configuração the section called “Autenticação do token do portador” antes de continuar.

Formato de solicitação

Só `application/json` é aceito como o tipo de conteúdo.

Objeto JSON único:

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

Matriz JSON de objetos:

```
[
  {"timestamp":1771007942000,"message":"event one","level":"INFO"},
  {"timestamp":1771007943000,"message":"event two","level":"ERROR"}
]
```

Tipos de valores JSON aceitos

Esse endpoint é estrito — somente objetos JSON são aceitos como eventos.

Input	Comportamento
Objeto JSON único	Aceito como um evento
Matriz JSON de objetos	Cada objeto se torna um evento separado
Matriz vazia `[]`	Aceito, produz 0 eventos
Não objeto na matriz (string, número, etc.)	Ignorado
Primitivo de nível superior (`"hello"`, `42`)	Ignorado
Objetos concatenados `{...}{...}`	Somente o primeiro objeto analisado

Exemplo — matriz com tipos mistos:

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

Resultado: 2 eventos ingeridos (os objetos), 2 ignorados (a string e o número).

Campo de carimbo de data/hora

O `"timestamp"` campo está em milissegundos de época, o mesmo que o endpoint NDJSON.

Formato	Exemplo	Interpretada como
Numérico (millis)	`"timestamp":1771007942000`	1771007942000 ms
Missing (Ausente)	(sem campo de carimbo de data/hora)	Hora atual do servidor
Não numérico	`"timestamp":"invalid"`	Hora atual do servidor

Exemplo de solicitação

```
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"}]'
```

Respostas

Sucesso (todos os eventos são aceitos):

```
HTTP 200 OK
{}
```

Sucesso parcial (alguns eventos foram rejeitados):

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

O `rejectedLogRecords` campo é o número total de eventos rejeitados. O `errorMessage` campo contém um detalhamento codificado em JSON por motivo de rejeição:

- `tooOldLogEventCount`— Eventos com carimbos de data/hora anteriores ao período de retenção

- `tooNewLogEventCount`— Eventos com timestamps muito distantes no futuro
- `expiredLogEventCount`— Eventos que expiraram durante o processamento

Práticas recomendadas

eventos em lotes

Para melhor desempenho e eficiência:

- Agrupe vários eventos em uma única solicitação, quando possível
- Tamanho de lote recomendado: 10 a 100 eventos por solicitação
- Tamanho máximo da solicitação: 1 MB

Tratamento de erros

Implemente o tratamento adequado de erros em seu aplicativo. Códigos de status HTTP comuns:

- `200 OK`— Registros ingeridos com sucesso
- `400 Bad Request`— Formato ou parâmetros de solicitação inválidos
- `401 Unauthorized`— Token de portador inválido ou expirado
- `403 Forbidden`— Permissões insuficientes
- `404 Not Found`— O grupo de log ou stream não existe
- `429 Too Many Requests`— Limite de taxa excedido
- `500 Internal Server Error`— Erro de serviço (tente novamente com recuo exponencial)

Limitações

- Tamanho máximo do evento: 256 KB por evento
- Tamanho máximo da solicitação: 1 MB
- Máximo de eventos por solicitação: 10.000
- Os nomes dos grupos de registros devem seguir as convenções de nomenclatura de CloudWatch registros
- A autenticação do token do portador deve ser ativada no grupo de registros se a autenticação do token do portador for usada.

Comparação dos endpoints de ingestão de HTTP

Recurso	Registros do HLC	Registros ND-JSON	Registros JSON estruturados	OpenTelemetry Registros
Path	`/services/collector/event`	`/ingest/bulk`	`/ingest/json`	`/v1/logs`
Content-Type	`application/json`	`application/json` ou `application/x-ndjson`	`application/json`	`application/json` ou `application/x-protobuf`
Campo de carimbo de data/hora	`"time"` (segundos)	`"timestamp"` (milissegundos)	`"timestamp"` (milissegundos)	`"timeUnixNano"` (nanossegundos)
Campos obrigatórios	`"event"`	Nenhum	Nenhum	Estrutura OTLP () `"resourceLogs"`
Resposta parcial de sucesso	Não	Sim	Sim	Sim
Suporte a parâmetros de consulta	Sim	Sim	Sim	Não (somente cabeçalhos)
Metadados da entidade	Sim	Sim	Sim	Não
Aceita primitivos	Não	Sim	Não	Não
Análise baseada em linha	Não	Sim	Não	Não

Recurso	Registros do HLC	Registros ND-JSON	Registros JSON estruturados	OpenTelemetry Registros
Suporte ao Protobuf	Não	Não	Não	Sim
Cabeçalho Retry-After	Não	Não	Não	Sim

Escolhendo um endpoint

- Usando o formato HLC? Use os registros do HLC. Suas cargas úteis de HLC existentes funcionam com o mínimo de alterações.
- line-by-lineRegistros de streaming? Use registros ND-JSON. Ideal para pipelines de log que emitem um evento por linha. Mais flexível — aceita qualquer tipo de valor JSON.
- Enviando cargas JSON estruturadas? Use registros JSON estruturados. Ideal para aplicativos que produzem objetos ou matrizes JSON bem formados.
- Já está usando OpenTelemetry? Use OpenTelemetry registros. Aceita o formato OTLP JSON ou Protobuf e oferece suporte a respostas parciais de sucesso com semântica de repetição.

Ingestão de Syslog

A ingestão gerenciada de syslog do Amazon CloudWatch Logs permite que você envie mensagens de syslog (RFC 5424, RFC 3164 e Cisco FTD/ASA) de firewalls, roteadores, switches e servidores Linux diretamente para o Logs, sem instalar ou gerenciar nenhum agente. CloudWatch Suas fontes de syslog enviam mensagens por TCP, TCP+TLS ou UDP para um endpoint VPC em sua conta. O tráfego é canalizado AWS PrivateLink para o serviço syslog CloudWatch Logs, que analisa automaticamente as mensagens recebidas e extrai campos estruturados, como instalação, gravidade, nome do host e nome do aplicativo, eliminando a necessidade de pipelines de análise personalizados. Em seguida, você pode consultar esses campos usando o CloudWatch Logs Analytics para investigar eventos de segurança ou solucionar problemas de conectividade. Isso ajuda você a centralizar a visibilidade dos registros da infraestrutura, simplificar os fluxos de trabalho operacionais e reduzir a sobrecarga de implantação e manutenção de agentes de coleta de registros em ambientes distribuídos.

Se suas fontes de syslog já estiverem na sua Amazon VPC, elas poderão enviar mensagens diretamente para o endpoint da VPC. Se suas fontes estiverem fora AWS (data centers locais, filiais ou instalações de co-localização), elas poderão acessar o endpoint VPC por meio de sua conexão VPN ou Direct Connect.

Protocolos e portas compatíveis

Protocolo	Porta	Observações
TCP + TLS	6514	Criptografado em trânsito. Recomendado para requisitos de conformidade.
Texto simples TCP	1514	Texto sem formatação AWS PrivateLink (isolado em rede).
UDP	514	Best-effort entrega.

O TLS na porta 6514 é encerrado no balanceador de carga de rede usando um certificado AWS gerenciado emitido pela Amazon Trust Services. Seus clientes syslog confiam nesse certificado automaticamente sem nenhuma configuração especial.

> **Note**
>
> O UDP é o melhor protocolo. As mensagens podem ser perdidas devido às condições da rede. Use o TCP para uma entrega confiável.

Formatos de syslog suportados

CloudWatch O Logs detecta e analisa automaticamente os seguintes formatos de syslog:

- RFC 5424 (o formato mais recente) — Inclui dados estruturados, registros de data e hora ISO 8601 e campos explícitos de nome do aplicativo e ID do processo.
- RFC 3164 (syslog BSD, o formato legado) — Inclui BSD-style timestamps e um campo TAG. Ainda é amplamente usado por dispositivos de rede, como firewalls, roteadores e switches.
- Cisco FTD/ASA — O formato syslog usado pelos dispositivos Cisco Firepower Threat Defense (FTD) e Adaptive Security Appliance (ASA). As mensagens são identificadas pela `%ASA-` tag `%FTD-` ou no corpo da mensagem.

As mensagens são armazenadas em seu grupo de registros em seu formato bruto original. CloudWatch O Logs extrai automaticamente campos estruturados de cada formato, que você pode consultar usando o CloudWatch Logs Insights.

Campos extraídos do RFC 5424

Campo	Description
`facility`	Nome da categoria de registro (por exemplo`kern`,,`auth`,`local0`).
`facilityCode`	Código numérico da instalação (0—23).
`severity`	Nome do nível de severidade (por exemplo`emerg`,`err`,,`info`).
`severityCode`	Código numérico de severidade (0—7).
`timestamp`	Carimbo de data/hora da mensagem no formato ISO 8601.
`hostname`	Nome do host do dispositivo de origem.

Campo	Description
`appName`	Application name (Nome da aplicação).
`procId`	ID do processo.
`msgId`	Identificador de mensagem.
`structuredData`	Elementos de dados estruturados RFC 5424 (metadados de valor-cha ve).
`message`	Corpo da mensagem.

Campos extraídos do RFC 3164

Campo	Description
`facility`	Nome da categoria de registro.
`facilityCode`	Código numérico da instalação (0—23).
`severity`	Nome do nível de severidade.
`severityCode`	Código numérico de severidade (0—7).
`timestamp`	Carimbo de data/hora da mensagem (convertido do formato BSD para ISO 8601).
`hostname`	Nome do host do dispositivo de origem.
`appName`	Nome do aplicativo (extraído do campo TAG).
`procId`	ID do processo (extraído do campo TAG, se houver).
`message`	Corpo da mensagem.

Campos FTD/ASA extraídos da Cisco

Campo	Description
`device`	Tipo de dispositivo (`FTD`,`ASA`, ou`FMC-AUDIT-LOG`).
`timestamp`	Carimbo de data/hora da mensagem (formato RFC 3164 ou RFC 5424, dependendo da configuração do dispositivo).
`deviceId`	Nome do host do dispositivo (presente quando o `device-id` registro é configurado no dispositivo).
`severity`	Nome do nível de severidade (por exemplo`informati onal` ,`warning`,,`critical`).
`severityLevel`	Nível de severidade numérico (0—7).
`messageId`	Identificador numérico de mensagem da Cisco (por exemplo,`106023`,`302013`).
`subsystem`	Nome do subsistema (presente para certos tipos de mensagens).
`message`	Corpo da mensagem (texto sem formatação) ou campos de valores-c have individuais quando o corpo usa o formato estruturado da Cisco.

Entrega de mensagens

Diferentemente da HTTP-based ingestão, em que o servidor retorna um código de status para cada solicitação, o syslog não fornece uma confirmação de sucesso por mensagem ao remetente. Depois que as mensagens são recebidas pelo serviço, elas são armazenadas em buffer e entregues ao seu grupo de registros com novas tentativas de erros transitórios. As garantias de entrega dependem do protocolo de transporte que você escolher:

- TCP (portas 6514 e 1514) — Fornece entrega confiável em condições operacionais normais.

  Quando a entrega não é possível, o serviço redefine a conexão TCP para sinalizar a falha ao seu cliente. As mensagens em andamento nessa conexão podem ser descartadas, mas a reinicialização da conexão fornece contrapressão imediata para que seu cliente possa detectar o

problema e armazenar as mensagens localmente até que a condição seja resolvida. Sob pressão de capacidade, o serviço rejeita novas conexões TCP mais cedo, fornecendo o mesmo sinal de contrapressão.

As condições que causam a redefinição da conexão incluem:

- A política de endpoint da VPC nega acesso
- A `PutLogEvents` cota da sua conta foi excedida
- O grupo de registros de destino não existe
- A política de recursos no grupo de registros nega acesso

- UDP (porta 514) — Best-effort entrega. As mensagens que não podem ser entregues são descartadas sem feedback para o remetente. As mensagens também podem ser perdidas devido ao congestionamento da rede ou a restrições de capacidade. Use o TCP se a entrega confiável for importante para seu caso de uso.

Para detectar e responder a problemas de entrega, monitore a `SyslogMessagesDropped` métrica em CloudWatch. A `Reason` dimensão indica por que as mensagens foram descartadas para que você possa tomar medidas corretivas. Para obter mais informações, consulte the section called “Monitoramento ”.

Cotas e limites

Limite	Valor	Observações
Tamanho máximo da mensagem (TCP)	64 KB	As mensagens de syslog padrão geralmente estão bem abaixo desse limite. Se você tiver um caso de uso que exija mensagens maiores, entre em contato com o AWS Suporte.
Tamanho máximo da mensagem (UDP)	8 KB	As mensagens de syslog padrão geralmente estão bem abaixo desse limite. Se você tiver um caso de uso que exija mensagens maiores, entre em contato com o AWS Suporte.

Limite	Valor	Observações
Taxa de transferência de ingestão	Compartilhado com `PutLogEvents`	A ingestão de Syslog conta com base na `PutLogEvents` cota da sua conta (5.000 solicitações por segundo por conta por região, por padrão).

A cota de `PutLogEvents` é ajustável. Se seu tráfego de syslog exigir maior taxa de transferência, solicite um aumento de cota por meio de cotas de serviço. https://docs.aws.amazon.com/servicequotas/latest/userguide/request-quota-increase.html

Configurando a ingestão de syslog

Esta seção mostra as etapas para configurar a ingestão de syslog no Logs. CloudWatch Você criará um VPC endpoint para o serviço syslog, um grupo de logs para receber as mensagens, uma política de recursos para autorizar o serviço syslog e uma configuração de syslog que roteará o tráfego do seu VPC endpoint para seu grupo de logs.

Você pode executar todas essas etapas usando o AWS Management Console AWS CLI, o ou os AWS SDKs. As instruções a seguir fornecem console e AWS CLI exemplos.

Pré-requisitos

A identidade do IAM (usuário ou função) que você usa para configurar a ingestão de syslog deve ter permissões para criar endpoints de VPC, grupos de registros, políticas de recursos e configurações de syslog. O exemplo de política a seguir mostra as permissões mínimas necessárias:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:PutResourcePolicy",
        "logs:DeleteResourcePolicy",
        "logs:PutSyslogConfiguration",
```

```
        "logs:ListSyslogConfigurations",
        "logs:DeleteSyslogConfiguration"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateVpc",
        "ec2:ModifyVpcAttribute",
        "ec2:CreateSubnet",
        "ec2:CreateSecurityGroup",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:CreateVpcEndpoint",
        "ec2:ModifyVpcEndpoint",
        "ec2:DescribeVpcEndpoints"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Se você já tem uma VPC, uma sub-rede e um grupo de segurança, você só precisa das `ec2:DescribeVpcEndpoints` permissões`ec2:CreateVpcEndpoint`,`ec2:ModifyVpcEndpoint`, e para as ações do EC2.

Etapa 1: criar ou identificar uma VPC

Você precisa de uma VPC que possa ser acessada pela sua rede local (via VPN ou Direct Connect) onde residem seus dispositivos geradores de syslog. Se você já tem uma VPC conectada ao seu data center, pule esta etapa e use sua VPC e IDs de sub-rede existentes.

Console

Para criar uma VPC (console)

1. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.

2. No painel de navegação, escolha Suas VPCs e, em seguida, escolha Criar VPC.
3. Em Recursos a serem criados, escolha Somente VPC.
4. Para o bloco CIDR IPv4, digite `10.0.0.0/16` (ou um CIDR que não entre em conflito com sua rede local).
5. Escolha Criar VPC.
6. Selecione a VPC recém-criada, escolha Ações, Editar configurações de VPC. Ative a resolução DNS e os nomes de host DNS e escolha Salvar.
7. Crie uma sub-rede para o VPC endpoint:

    a. No painel de navegação, escolha Sub-redes e, em seguida, escolha Criar sub-rede.
    b. Para VPC ID, selecione a VPC que você criou.
    c. Em Zona de disponibilidade, escolha uma Zona de disponibilidade.
    d. Para o bloco CIDR de sub-rede IPv4, digite. `10.0.1.0/24`
    e. Escolha Criar sub-rede.

AWS CLI

```
REGION=us-east-1

# Create VPC
VPC_ID=$(aws ec2 create-vpc \
  --cidr-block 10.0.0.0/16 \
  --region $REGION \
  --query 'Vpc.VpcId' --output text)

aws ec2 modify-vpc-attribute --vpc-id $VPC_ID --enable-dns-support --region $REGION
aws ec2 modify-vpc-attribute --vpc-id $VPC_ID --enable-dns-hostnames --region
 $REGION

# Create a subnet for the VPC endpoint
SUBNET_ID=$(aws ec2 create-subnet \
  --vpc-id $VPC_ID \
  --cidr-block 10.0.1.0/24 \
  --availability-zone ${REGION}a \
  --region $REGION \
  --query 'Subnet.SubnetId' --output text)

echo "VPC: $VPC_ID, Subnet: $SUBNET_ID"
```

> **Note**
>
> O VPC endpoint cria uma interface de rede elástica (ENI) com um IP privado em sua sub-rede. Seus dispositivos locais alcançam esse IP por meio de sua conexão VPN ou Direct Connect. Certifique-se de que seu roteamento de rede permita o tráfego de seus dispositivos para o CIDR da sub-rede.

Etapa 2: criar um grupo de segurança

Crie um grupo de segurança para o VPC endpoint que permita o tráfego syslog de entrada da sua VPC. Isso controla quais recursos podem enviar syslog para o endpoint.

Console

Para criar um grupo de segurança (console)

1. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel de navegação, escolha Grupos de segurança e, em seguida, escolha Criar grupo de segurança.
3. Em Nome do grupo de segurança, insira `syslog-vpce-sg`.
4. Em Descrição, insira `Allow syslog traffic to VPC endpoint`.
5. Para VPC, selecione a VPC que você criou ou identificou na Etapa 1.
6. Na seção Regras de entrada, escolha Adicionar regra e adicione as seguintes regras:
   - Regra 1: Tipo = TCP personalizado, intervalo de portas =`6514`, Fonte = `10.0.0.0/16` (seu CIDR de VPC)
   - Regra 2: Tipo = TCP personalizado, Intervalo de portas =`1514`, Fonte = `10.0.0.0/16`
   - Regra 3: Tipo = UDP personalizado, Intervalo de portas =`514`, Fonte = `10.0.0.0/16`
7. Escolha Criar grupo de segurança.

AWS CLI

```
VPCE_SG_ID=$(aws ec2 create-security-group \
  --group-name syslog-vpce-sg \
  --description "Allow syslog traffic to VPC endpoint" \
  --vpc-id $VPC_ID \
```

```
  --region $REGION \
  --query 'GroupId' --output text)

# Allow TCP+TLS (port 6514), TCP plaintext (port 1514), and UDP (port 514)
aws ec2 authorize-security-group-ingress --group-id $VPCE_SG_ID \
  --protocol tcp --port 6514 --cidr 10.0.0.0/16 --region $REGION
aws ec2 authorize-security-group-ingress --group-id $VPCE_SG_ID \
  --protocol tcp --port 1514 --cidr 10.0.0.0/16 --region $REGION
aws ec2 authorize-security-group-ingress --group-id $VPCE_SG_ID \
  --protocol udp --port 514 --cidr 10.0.0.0/16 --region $REGION
```

> **Tip**
>
> Se você planeja usar apenas um protocolo (por exemplo, TCP+TLS na porta 6514), você só precisa abrir essa porta no grupo de segurança.

Etapa 3: Criar o VPC endpoint.

Crie uma interface VPC endpoint apontando para o serviço syslog. AWS PrivateLink Isso dá à sua VPC um ponto de entrada privado para o serviço syslog do CloudWatch Logs.

> **Note**
>
> Você pode especificar várias IDs de sub-rede em diferentes zonas de disponibilidade ao criar o endpoint. O endpoint cria uma ENI em cada sub-rede, fornecendo maior disponibilidade sem precisar de VPC endpoints separados por zona de disponibilidade. Um único VPC endpoint com sub-redes em várias zonas de disponibilidade é suficiente.

Console

Para criar o VPC endpoint (console)

1. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel de navegação, escolha Endpoints e, em seguida, escolha Create endpoint.
3. Em Etiqueta de nome, insira um nome para o endpoint (por exemplo,`syslog-vpce`).
4. Em Categoria do serviço, escolha Serviços do AWS .

5. No campo Pesquisa de serviços, insira `syslog-logs` e selecione o serviço`com.amazonaws.Region.syslog-logs`.
6. Para VPC, selecione a VPC que você criou ou identificou na Etapa 1.
7. Na seção Sub-redes, selecione uma ou mais zonas de disponibilidade e escolha as sub-redes nas quais você deseja criar as interfaces de rede do endpoint.
8. Para Grupos de segurança, selecione o grupo de segurança que você criou na Etapa 2 (`syslog-vpce-sg`).
9. (Opcional) Se você quiser restringir o tráfego permitido pelo endpoint, configure uma política de VPC endpoint. Para obter mais informações, consulte the section called “Políticas de VPC endpoint”.
10. Escolha Criar endpoint.
11. Depois que o estado do endpoint mudar para Disponível, selecione o endpoint e anote o valor dos nomes DNS. Este é o endereço para o qual seus dispositivos syslog enviarão.

AWS CLI

```
VPCE_ID=$(aws ec2 create-vpc-endpoint \
  --vpc-id $VPC_ID \
  --service-name com.amazonaws.${REGION}.syslog-logs \
  --vpc-endpoint-type Interface \
  --subnet-ids $SUBNET_ID \
  --security-group-ids $VPCE_SG_ID \
  --region $REGION \
  --query 'VpcEndpoint.VpcEndpointId' --output text)

echo "VPC Endpoint: $VPCE_ID"
```

Aguarde até que o endpoint fique disponível (aproximadamente 60 segundos) e, em seguida, recupere o nome do DNS:

```
VPCE_DNS=$(aws ec2 describe-vpc-endpoints --vpc-endpoint-ids $VPCE_ID \
  --region $REGION --query 'VpcEndpoints[0].DnsEntries[0].DnsName' --output text)

echo "Endpoint DNS: $VPCE_DNS"
```

Salve o `VPCE_DNS` valor — você configurará seus dispositivos syslog para enviar para esse endereço.

Etapa 4: criar um grupo de registros

Crie o grupo de CloudWatch registros de registros em que suas mensagens de syslog serão entregues. Você pode usar qualquer nome de grupo de registros. Recomendamos usar um `/syslog/` prefixo para maior clareza organizacional.

Console

Para criar um grupo de registros (console)

1. Abra o console CloudWatch de registros em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, em Gerenciamento de registros, escolha Grupos de registros.
3. Escolha a opção Criar grupo de logs.
4. Em Nome do grupo de registros, insira`/syslog/my-devices`.
5. (Opcional) Defina as configurações de retenção e criptografia conforme necessário.
6. Escolha Criar.

AWS CLI

```
aws logs create-log-group \
  --log-group-name /syslog/my-devices \
  --region $REGION
```

Você não precisa criar um fluxo de log. O serviço syslog cria automaticamente um fluxo de log chamado *VPCE_ID*_Syslog_*Region* (por exemplo,`vpce-0abc123def456_Syslog_us-east-1`) quando a primeira mensagem é entregue.

Etapa 5: adicionar uma política de recursos

O serviço syslog do CloudWatch Logs grava no seu grupo de registros usando o `syslog.logs.amazonaws.com` Service Principal. Você deve conceder permissão por meio de uma política de recursos em seu grupo de registros. A `aws:SourceArn` condição garante que somente o tráfego do seu VPC endpoint específico possa gravar nesse grupo de registros.

```
ACCOUNT_ID=$(aws sts get-caller-identity --query 'Account' --output text)
```

```
aws logs put-resource-policy \
  --policy-name syslog-ingestion \
  --policy-document '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "syslog.logs.amazonaws.com"
      },
      "Action": [
        "logs:PutLogEvents",
        "logs:CreateLogStream"
      ],
      "Resource": "arn:aws:logs:'$REGION':'$ACCOUNT_ID':log-group:/syslog/my-
devices:*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "'$ACCOUNT_ID'"
        },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:ec2:'$REGION':'$ACCOUNT_ID':vpc-
endpoint/'$VPCE_ID'"
        }
      }
    }
  ]
}' \
  --region $REGION
```

As condições na política de recursos fornecem as seguintes proteções:

- `aws:SourceAccount`— Evita ataques confusos de deputados. Somente o tráfego da sua conta é aceito.
- `aws:SourceArn`— Escopo o acesso a um VPC endpoint específico. Se você tiver vários VPC endpoints, adicione cada um como um ARN separado na condição.

Para permitir que vários VPC endpoints gravem no mesmo grupo de registros, use o operador de `ArnLike` condição com um caractere curinga:

```
"ArnLike": {
    "aws:SourceArn": "arn:aws:ec2:us-east-1:123456789012:vpc-endpoint/*"
```

```
}
```

Etapa 6: Criar a configuração do syslog

Essa etapa informa ao serviço de syslog do CloudWatch Logs que o tráfego que chega do seu VPC endpoint deve ser roteado para seu grupo de registros. Sem essa configuração, o tráfego do seu endpoint é rejeitado.

Console

Para criar a configuração do syslog (console)

1. Abra o console CloudWatch de registros em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, em Gerenciamento de registros, escolha Grupos de registros.
3. Escolha o grupo de registros que você criou na Etapa 4 (por exemplo,`/syslog/my-devices`).
4. Nos detalhes do grupo de registros, localize a seção Ingestão de Syslog.
5. Selecione Configurar.
6. No menu suspenso VPC endpoint, selecione o VPC endpoint que você criou na Etapa 3.
7. Escolha Criar.

AWS CLI

```
aws logs put-syslog-configuration \
  --log-group-identifier /syslog/my-devices \
  --vpc-endpoint-id $VPCE_ID \
  --region $REGION
```

Verifique a configuração:

```
aws logs list-syslog-configurations \
  --log-group-identifier /syslog/my-devices \
  --region $REGION
```

Seu pipeline de ingestão de syslog agora está ativo. Todas as mensagens de syslog enviadas ao VPC endpoint serão entregues ao grupo de logs. `/syslog/my-devices`

Etapa 7: Verificar a entrega e a extração do campo

Envie uma mensagem de teste de qualquer host ou dispositivo EC2 que possa acessar o VPC endpoint e CloudWatch use o Log Analytics para verificar se a mensagem foi entregue e se os campos estruturados foram extraídos corretamente. As mensagens geralmente aparecem em 10 a 20 segundos.

Envie uma mensagem de teste (texto sem formatação TCP):

```
echo "<134>1 $(date -u +%Y-%m-%dT%H:%M:%SZ) myhost myapp 1234 - - Hello from syslog" |
 \
  nc $VPCE_DNS 1514
```

Verifique a entrega e os campos extraídos:

Console

Para verificar a entrega usando o Log Analytics (console)

1. Abra o console CloudWatch de registros em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, em Registros, escolha Log Analytics.
3. No seletor de grupos de registros, escolha seu grupo de registros (por exemplo,`/syslog/my-devices`).
4. Insira a consulta a seguir e escolha Executar consulta:

   ```
   fields @timestamp, facility, severity, hostname, appName, procId, message
   | sort @timestamp desc
   | limit 10
   ```

5. Verifique se sua mensagem de teste aparece e se os campos extraídos foram preenchidos corretamente. Para a mensagem de teste acima, você deve ver `facility` `severity` `=local0`, `hostname` = `infomyhost`, `=myapp`, `appName` = e `procId` `=1234`.

AWS CLI

Inicie uma consulta do Log Analytics para verificar a entrega e a extração do campo:

```
QUERY_ID=$(aws logs start-query \
```

```
  --log-group-name /syslog/my-devices \
  --start-time $(date -d '5 minutes ago' +%s 2>/dev/null || echo $(date -v-5M +%s))
 \
  --end-time $(date +%s) \
  --query-string 'fields @timestamp, facility, severity, hostname, appName, procId,
 message | sort @timestamp desc | limit 10' \
  --region $REGION \
  --query 'queryId' --output text)

# Wait a few seconds for the query to complete, then retrieve results
aws logs get-query-results \
  --query-id $QUERY_ID \
  --region $REGION
```

Verifique se sua mensagem de teste aparece e se os campos extraídos foram preenchidos corretamente. Para a mensagem de teste acima, você deve ver `facility` `severity` =`local0`, `hostname` = `infomyhost`, =`myapp`, `appName` = e `procId` =`1234`.

Configurando dispositivos syslog

Depois de configurar a ingestão de syslog, configure suas fontes de syslog para enviar mensagens para o nome DNS do VPC endpoint. Seus dispositivos alcançam o endpoint por meio de sua conexão VPN ou Direct Connect à VPC.

> **Note**
>
> As configurações a seguir são fornecidas como exemplos. Para obter as opções completas de configuração e as melhores práticas, consulte a documentação oficial da implementação do syslog (rsyslog, syslog-ng ou o fornecedor do seu dispositivo).

Nos exemplos a seguir, *`VPCE_DNS`* substitua pelo nome DNS do VPC endpoint do procedimento de configuração (por exemplo,). `vpce-0abc123def456.syslog-logs.us-east-1.vpce.amazonaws.com`

rsyslog

rsyslog é o daemon syslog padrão na maioria das distribuições Linux. Adicione a seguinte configuração ao`/etc/rsyslog.d/50-cwl-syslog.conf`.

TCP+TLS:

```
# Forward all logs to CloudWatch Logs via TCP+TLS
action(type="omfwd"
  target="VPCE_DNS"
  port="6514"
  protocol="tcp"
  StreamDriver="gtls"
  StreamDriverMode="1"
  StreamDriverAuthMode="x509/certvalid"
  template="RSYSLOG_SyslogProtocol23Format"
  queue.type="LinkedList"
  queue.filename="cwl_fwd"
  queue.saveOnShutdown="on"
  action.resumeRetryCount="-1")
```

Texto sem formatação TCP:

```
# Forward all logs to CloudWatch Logs via plaintext TCP
action(type="omfwd"
  target="VPCE_DNS"
  port="1514"
  protocol="tcp"
  template="RSYSLOG_SyslogProtocol23Format"
  queue.type="LinkedList"
  queue.filename="cwl_fwd"
  queue.saveOnShutdown="on"
  action.resumeRetryCount="-1")
```

UDP:

```
# Forward all logs to CloudWatch Logs via UDP
action(type="omfwd"
  target="VPCE_DNS"
  port="514"
  protocol="udp"
  template="RSYSLOG_SyslogProtocol23Format")
```

Depois de adicionar a configuração, reinicie o rsyslog:

```
sudo systemctl restart rsyslog
```

> **Note**
>
> As `queue.saveOnShutdown="on"` diretivas `queue.type="LinkedList"` e permitem o enfileiramento assistido por disco. Se a conexão com o VPC endpoint for temporariamente perdida, o rsyslog armazena as mensagens em buffer localmente e tentará a entrega novamente.

Filtragem por instalação ou gravidade:

Para enviar somente mensagens específicas, adicione um filtro antes da ação:

```
# Send only auth messages via TCP+TLS
auth,authpriv.* action(type="omfwd"
  target="VPCE_DNS"
  port="6514"
  protocol="tcp"
  StreamDriver="gtls"
  StreamDriverMode="1"
  StreamDriverAuthMode="x509/certvalid"
  template="RSYSLOG_SyslogProtocol23Format")
```

syslog-ng

O syslog-ng é comumente usado em dispositivos de rede e sistemas Linux corporativos. Adicione a seguinte configuração ao`/etc/syslog-ng/conf.d/cwl-syslog.conf`.

TCP+TLS:

```
destination d_cwl_syslog {
  network("VPCE_DNS"
    port(6514)
    transport("tls")
    tls(
      peer-verify(required-trusted)
      ca-dir("/etc/ssl/certs")
    )
  );
};

log {
```

```
  source(s_sys);
  destination(d_cwl_syslog);
};
```

Texto sem formatação TCP:

```
destination d_cwl_syslog {
  network("VPCE_DNS"
    port(1514)
    transport("tcp")
  );
};

log {
  source(s_sys);
  destination(d_cwl_syslog);
};
```

UDP:

```
destination d_cwl_syslog {
  network("VPCE_DNS"
    port(514)
    transport("udp")
  );
};

log {
  source(s_sys);
  destination(d_cwl_syslog);
};
```

Depois de adicionar a configuração, reinicie o syslog-ng:

```
sudo systemctl restart syslog-ng
```

Validação rápida

Use os comandos a seguir de qualquer host que possa acessar o VPC endpoint para verificar a conectividade.

Texto sem formatação TCP (porta 1514):

```
echo "<134>1 $(date -u +%Y-%m-%dT%H:%M:%SZ) myhost myapp 1234 - - Hello from syslog" |
 \
  nc VPCE_DNS 1514
```

TCP + TLS (porta 6514):

```
echo "<134>1 $(date -u +%Y-%m-%dT%H:%M:%SZ) myhost myapp 1234 - - Hello TLS syslog" | \
  openssl s_client -connect VPCE_DNS:6514 -quiet -no_ign_eof
```

UDP (porta 514):

```
echo "<134>1 $(date -u +%Y-%m-%dT%H:%M:%SZ) myhost myapp 1234 - - Hello UDP syslog" | \
  nc -u -w1 VPCE_DNS 514
```

Depois de enviar as mensagens de teste, verifique a entrega no CloudWatch Logs (geralmente em 10 a 20 segundos):

```
aws logs filter-log-events \
  --log-group-name /syslog/my-devices \
  --start-time $(date -d '5 minutes ago' +%s000 2>/dev/null || echo $(($(date -v-5M +
%s) * 1000))) \
  --region $REGION
```

Dispositivos de rede

A maioria dos firewalls, roteadores e switches suporta a configuração de um servidor syslog externo especificando um endereço IP e uma porta de destino. Use o endereço IP privado do VPC endpoint ENI (visível no console Amazon VPC em Endpoints) e uma das portas suportadas (6514, 1514 ou 514).

Consulte a documentação do seu dispositivo para obter a sintaxe de configuração específica. As principais configurações a serem definidas são:

- Endereço do servidor — O nome DNS do VPC endpoint ou o endereço IP privado da ENI.
- Porta — 6514 (TCP+TLS), 1514 (TCP) ou 514 (UDP).
- Protocolo — TCP (recomendado) ou UDP.
- Formato — RFC 5424 (preferencial) ou RFC 3164 (antigo). Ambos são compatíveis.

Confiança do certificado TLS

A conexão TLS na porta 6514 usa um certificado AWS gerenciado emitido pela Amazon Trust Services. A maioria dos sistemas operacionais e daemons syslog confiam automaticamente nesse certificado porque os certificados raiz do Amazon Trust Services estão incluídos nos repositórios confiáveis padrão da CA.

Se o seu dispositivo não confiar no certificado por padrão, baixe os certificados raiz do Amazon Trust Services https://www.amazontrust.com/repository/e adicione-os ao armazenamento confiável da CA do seu dispositivo.

Monitorando a ingestão de syslog

O serviço de ingestão de syslog publica métricas no namespace. CloudWatch `AWS/Logs` Essas métricas oferecem visibilidade do seu pipeline de ingestão de syslog — o que foi recebido, o que foi descartado e por quê.

Para visualizar essas métricas no CloudWatch console, navegue até Métricas > Todas as métricas > AWS/Logs e filtre pelos nomes das métricas listados abaixo.

Métricas do Syslog

Métrica	Dimensões	Description
`SyslogMessagesReceived`	`LogGroupName`	O número de mensagens de syslog ingeridas com sucesso em seu grupo de logs.
`SyslogMessagesDropped`	`LogGroupName` , `Reason`	O número de mensagens de syslog que não puderam ser entregues. Para mais detalhes, consulte the section called “Motivos da queda”.
`SyslogConnectionsRejected`	`Reason`	O número de conexões TCP que foram rejeitadas.
`SyslogConnectionsEstablished`	—	O número de conexões TCP aceitas com êxito.

Métrica	Dimensões	Description
`SyslogConnectionsClosed`	—	O número de conexões TCP fechadas.

Motivos da queda

A `SyslogMessagesDropped` métrica inclui uma `Reason` dimensão que indica por que as mensagens foram descartadas.

Motivo	Description
`MessageRateLimitExceeded`	A `PutLogEvents` cota da sua conta foi excedida. Considere solicitar um aumento de cota.
`MessageSizeExceeded`	Um datagrama UDP excedeu o tamanho máximo da mensagem.
`ServiceUnavailable`	A capacidade interna ou o limite de taxa foram excedidos. Isso geralmente é transitório.
`ResourceNotFound`	O grupo de registros de destino não existe. Verifique se o grupo de registros não foi excluído.
`AccessDenied`	A política de recursos no grupo de logs não concede acesso ao serviço syslog. Verifique se a política de recursos está correta.
`VpcePolicyDenied`	A política de endpoint da VPC negou a solicitação. Revise sua política de endpoint de VPC.
`InternalError`	Ocorreu um erro interno inesperado. Se isso persistir, entre em contato com o AWS Suporte.

Motivos de rejeição de conexão

A `SyslogConnectionsRejected` métrica inclui uma `Reason` dimensão.

Motivo	Description
`VpcePolicyDenied`	A política de endpoint da VPC negou a conexão.
`ServiceUnavailable`	A conexão caiu devido a um erro interno.

Alarmes recomendados

Recomendamos criar CloudWatch alarmes nas seguintes condições para detectar problemas precocemente:

Alarme	Condição	Ação sugerida
Mensagens sendo descartadas	`SyslogMessagesDropped` > 0 por 5 minutos	Investigue a `Reason` dimensão para determinar a causa.
Acesso negado	`SyslogMessagesDropped` com `Reason=AccessDenied`	Verifique se a política de recursos no grupo de registros está configurada corretamente.
Grupo de log ausente	`SyslogMessagesDropped` com `Reason=ResourceNotFound`	Verifique se o grupo de registros existe e não foi excluído.
Nenhuma mensagem recebida	`SyslogMessagesReceived` = 0 por 15 minutos (quando normalmente > 0)	Verifique se os dispositivos ainda estão enviando e se a conectividade de rede com o endpoint VPC está intacta.

Para obter informações sobre a criação de CloudWatch alarmes, consulte Criação de CloudWatch alarmes no Guia do CloudWatch usuário.

Visualizando métricas de syslog no painel automático

CloudWatch fornece um painel automático para CloudWatch registros que inclui uma seção de ingestão de syslog. Esse painel aparece automaticamente quando sua conta publica métricas de syslog — nenhuma configuração manual é necessária. O painel inclui um gráfico para cada métrica descrita emthe section called “Métricas do Syslog”. Cada gráfico exibe a taxa métrica dividida pelas dimensões relevantes, como grupo de registros ou motivo.

Acessar o painel

Para visualizar o painel automático do syslog:

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Painéis e, em seguida, selecione Painéis automáticos.
3. Escolha o painel CloudWatch de registros.
4. Role até a seção Syslog Ingestion.

Exemplos de consultas para dados de syslog

As seguintes consultas do CloudWatch Logs Insights demonstram padrões de análise comuns para dados de syslog. Eles usam os campos estruturados que o CloudWatch Logs extrai automaticamente das mensagens RFC 5424, RFC 3164 e Cisco. FTD/ASA Para ver a lista completa dos campos extraídos, consultethe section called “Formatos de syslog suportados”.

Essas consultas também estão disponíveis no CloudWatch console. No CloudWatch Logs Insights, escolha Consultas e expanda a categoria Syslog.

Monitoramento de segurança e acesso

Eventos de escalonamento de sudo e privilégios

A consulta a seguir lista a atividade recente de escalonamento de privilégios do sudo em hosts Linux. Ele filtra o nome do `sudo` aplicativo e funciona com as mensagens RFC 3164 e RFC 5424.

```
filter appName="sudo"
| fields @timestamp, hostname, message
| sort @timestamp desc
| limit 50
```

Os 50 principais hosts com falha na autenticação SSH

A consulta a seguir conta as tentativas fracassadas de autenticação SSH para cada host. Ele corresponde às mensagens sshd que contêm “Senha com falha” ou “Chave pública com falha”. Use-o para identificar hospedeiros que são alvos de ataques de força bruta.

```
filter appName="sshd" and (message like /Failed password/ or message like /Failed
 publickey/)
| stats count(*) as failedAttempts by hostname
| sort failedAttempts desc
| limit 50
```

Visibilidade operacional

Registre o volume por host ao longo do tempo

A consulta a seguir conta as mensagens para cada host em intervalos de 5 minutos. Use-o para medir a atividade da frota, identificar picos de volume ou detectar hosts que pararam de enviar. Essa consulta funciona com todos os tipos de formato syslog.

```
stats count(*) as messages by hostname, bin(5m)
| sort messages desc
```

Volume de mensagens por instalação e severidade

A consulta a seguir conta as mensagens agrupadas por instalação e nível de gravidade. Use-o para entender quais subsistemas geram mais tráfego de log e em qual nível de severidade.

```
stats count(*) as messageCount by facility, severity
| sort messageCount desc
```

High-severity mensagens por aplicativo

A consulta a seguir filtra mensagens de alta prioridade (SeverityCode 0—3, que corresponde a emerg, alert, crit e err) e as conta por aplicativo. Use-o para identificar quais aplicativos estão gerando mais erros.

```
filter severityCode<=3
| stats count(*) as errorCount by appName
```

```
| sort errorCount desc
```

Warning-and-above tendência ao longo do tempo

A consulta a seguir conta as mensagens com severidade de aviso ou superior (SeverityCode 0—4) em intervalos de 5 minutos. Use-o com o recurso de visualização no Logs Insights para mapear os picos de erro ao longo do tempo.

```
filter severityCode<=4
| stats count(*) as events by bin(5m)
| sort @timestamp
```

Consultas de firewall e dispositivos de rede

Conexões negadas pela Cisco (ACL 106023)

A consulta a seguir é para firewalls Cisco ASA e FTD. Ele conta ACL-denied as conexões (ID de mensagem Cisco 106023) agrupadas por dispositivo, o que ajuda a identificar dispositivos com o tráfego mais negado.

```
filter messageId="106023"
| stats count(*) as denies by deviceId
| sort denies desc
| limit 20
```

ASA/FTD Eventos da Cisco por ID de mensagem

A consulta a seguir é para firewalls Cisco FTD e ASA. Ele conta os eventos agrupados por ID de mensagem e gravidade da Cisco para ajudá-lo a identificar os tipos de eventos mais frequentes em sua frota de firewall.

```
filter ispresent(messageId)
| stats count(*) as eventCount by messageId, severity
| sort eventCount desc
```

O firewall descartou pacotes por IP de origem

A consulta a seguir analisa mensagens DROP do firewall (por exemplo, de iptables ou nftables) para contar pacotes descartados por endereço IP de origem. Use-o para identificar as principais fontes de tráfego bloqueado.

```
filter message like /DROP/
| parse message /SRC=(?<srcAddr>\S+)/
| stats count(*) as drops by srcAddr
| sort drops desc
| limit 20
```

Políticas de VPC endpoint para syslog

Você pode anexar uma política de VPC endpoint ao seu endpoint de VPC syslog para controlar qual tráfego é permitido por meio dele. Isso fornece uma camada adicional de controle de acesso além dos grupos de segurança e das políticas de recursos.

Por padrão, um VPC endpoint tem uma política de acesso total que permite todo o tráfego. Você pode restringir isso anexando uma política personalizada. Para obter mais informações sobre as políticas de VPC endpoint, consulte Controlar o acesso usando políticas de endpoint no Guia.AWS PrivateLink

> **Important**
>
> O tráfego do syslog chega como um principal anônimo (sem identidade do IAM na rede). As políticas devem usar `"Principal": "*"` e definir o escopo do acesso usando chaves de ação, recurso ou condição da VPC. Principal-scoped políticas (por exemplo, políticas que permitem somente uma função específica do IAM) negarão todo o tráfego do syslog.

Chaves de condição compatíveis

As seguintes chaves de condição são suportadas nas políticas de VPC endpoint para syslog:

Chave de condição	Description	Valor de exemplo
`aws:SourceVpc`	A VPC da qual o tráfego se origina.	`vpc-111bbb22`
`aws:SourceVpce`	O VPC endpoint pelo qual o tráfego chega.	`vpce-0abc123def456`
`aws:VpcSourceIp`	O endereço IP privado do remetente na VPC.	`10.0.1.50`

Chave de condição	Description	Valor de exemplo
`aws:ResourceAccount`	A conta proprietária do grupo de registros de destino.	`123456789012`
`aws:ResourceOrgID`	O ID da organização da conta que possui o grupo de registros de destino.	`o-aa111bb222`
`aws:ResourceOrgPaths`	O caminho da unidade organizacional da conta que possui o grupo de registros de destino.	`o-aa111bb222/r-ab12/ou-ab12-11111111/*`

As seguintes chaves de condição não são suportadas porque o tráfego do syslog não tem identidade principal do IAM:

- `aws:PrincipalArn`
- `aws:PrincipalAccount`
- `aws:PrincipalOrgID`
- `aws:PrincipalOrgPaths`
- `aws:PrincipalTag/*`

Políticas que usam chaves de condição não suportadas resultarão em uma negação implícita, bloqueando todo o tráfego de syslog pelo endpoint.

Restringir a um grupo de registros específico

A política a seguir permite tráfego de syslog somente para um grupo de log específico:

```
aws ec2 modify-vpc-endpoint \
  --vpc-endpoint-id $VPCE_ID \
  --policy-document '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "logs:PutLogEvents",
```

```
      "Resource": "arn:aws:logs:Region:account-id:log-group:/syslog/my-devices"
    }
  ]
}' \
  --region $REGION
```

Negar todo o tráfego

A política a seguir bloqueia todo o tráfego de syslog pelo endpoint. Isso é útil para interromper temporariamente a ingestão sem excluir o endpoint:

```
aws ec2 modify-vpc-endpoint \
  --vpc-endpoint-id $VPCE_ID \
  --policy-document '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Principal": "*",
      "Action": "*",
      "Resource": "*"
    }
  ]
}' \
  --region $REGION
```

Redefinir para a política padrão

Para restaurar a política padrão de permitir tudo:

```
aws ec2 modify-vpc-endpoint \
  --vpc-endpoint-id $VPCE_ID \
  --reset-policy \
  --region $REGION
```

Propagação de políticas

Depois de criar ou modificar uma política de VPC endpoint, pode levar alguns minutos para que a alteração da política entre em vigor. Durante esse período de propagação, a política anterior (ou política padrão) continua sendo aplicada.

Se uma política de VPC endpoint negar tráfego de syslog, você verá os seguintes indicadores:

- As conexões TCP são fechadas imediatamente após a avaliação da política.
- Os datagramas UDP são descartados.
- A `SyslogMessagesDropped` métrica com `Reason=VpcePolicyDenied` é emitida.
- A `SyslogConnectionsRejected` métrica with `Reason=VpcePolicyDenied` é emitida para conexões TCP.

Solução de problemas de ingestão de syslog

Esta seção descreve problemas comuns com a ingestão de syslog e como resolvê-los.

Mensagens que não aparecem no grupo de registros

Se você enviou mensagens de syslog, mas elas não estão aparecendo no seu grupo de registros, verifique o seguinte na ordem:

1. Verifique se a configuração do syslog existe.

   Execute `list-syslog-configurations` para confirmar se seu VPC endpoint está registrado no grupo de registros:

   ```
   aws logs list-syslog-configurations \
     --log-group-identifier /syslog/my-devices \
     --region $REGION
   ```

   Se nenhuma configuração for retornada, crie uma com`put-syslog-configuration`.
2. Verifique a política de recursos.

   Garanta que a política de recursos no grupo de registros conceda `logs:PutLogEvents` e `logs:CreateLogStream` para `syslog.logs.amazonaws.com` e que a `aws:SourceArn` condição corresponda exatamente ao ARN do seu endpoint de VPC:

   ```
   aws logs describe-resource-policies --region $REGION
   ```

   Verifique a `SyslogMessagesDropped` métrica com `Reason=AccessDenied` para confirmar se o problema é a autorização.

3. Verifique o grupo de segurança.

   Certifique-se de que o grupo de segurança conectado ao VPC endpoint permita tráfego de entrada na porta que você está usando (6514, 1514 ou 514) do CIDR de origem:

   ```
   aws ec2 describe-security-groups \
     --group-ids $VPCE_SG_ID \
     --region $REGION \
     --query 'SecurityGroups[0].IpPermissions'
   ```

4. Verifique a conectividade da rede.

   No dispositivo ou em um host na mesma rede, teste a conectividade TCP com o endpoint:

   ```
   # TCP connectivity test
   nc -zv $VPCE_DNS 6514
   nc -zv $VPCE_DNS 1514
   ```

   Se a conexão for recusada ou atingir o tempo limite, verifique o roteamento do VPN/Direct Connect e se o dispositivo pode alcançar a sub-rede do VPC endpoint.

5. Verifique a política de VPC endpoint.

   Se você anexou uma política de VPC endpoint personalizada, verifique se ela permite as ações necessárias. Verifique a `SyslogMessagesDropped` métrica com`Reason=VpcePolicyDenied`.

Falhas do handshake TLS

Se seu cliente syslog não conseguir estabelecer uma conexão TLS na porta 6514:

- Confiança do certificado — Certifique-se de que o armazenamento confiável da CA do seu dispositivo inclua os certificados raiz do Amazon Trust Services. A maioria dos sistemas operacionais os inclui por padrão. Faça o download deles, https://www.amazontrust.com/repository/se necessário.
- Versão TLS — O serviço é compatível com TLS 1.2 e TLS 1.3. Certifique-se de que seu cliente syslog suporte pelo menos TLS 1.2.
- Verificação do nome do host — Se seu cliente verificar o nome do host do servidor, verifique se ele está configurado para se conectar usando o nome DNS do VPC endpoint (não um endereço IP).

Teste a conexão TLS manualmente com o OpenSSL:

```
openssl s_client -connect $VPCE_DNS:6514 -brief
```

Uma conexão bem-sucedida mostra a versão TLS e o conjunto de cifras. Se falhar, a mensagem de erro indica a causa.

Mensagens UDP perdidas

O UDP é o protocolo de melhor esforço. Alguma perda de mensagens é um comportamento esperado nas seguintes condições:

- Congestionamento de rede entre seu dispositivo e o VPC endpoint.
- As mensagens excedem o tamanho máximo do datagrama UDP.
- Restrições temporárias de capacidade de serviço (indicadas `SyslogMessagesDropped` com`Reason=ServiceUnavailable`).

Se for necessária uma entrega confiável, use TCP (porta 1514 ou 6514) em vez disso. O TCP fornece confirmação de entrega e retransmissão automática.

Conexão recusada

Se seu dispositivo relatar “conexão recusada” ao se conectar ao VPC endpoint:

- Verifique se o VPC endpoint está no estado: `available`

  ```
  aws ec2 describe-vpc-endpoints --vpc-endpoint-ids $VPCE_ID \
    --region $REGION --query 'VpcEndpoints[0].State'
  ```

- Verifique se você está se conectando em uma porta compatível (6514, 1514 ou 514).
- Verifique se o roteamento VPN ou Direct Connect permite o tráfego da rede do seu dispositivo para a sub-rede do VPC endpoint.
- Verifique se o grupo de segurança permite tráfego de entrada na porta que você está usando.

Mensagens interrompidas devido à limitação de taxa

Se a `SyslogMessagesDropped` métrica aparecer`Reason=MessageRateLimitExceeded`, o tráfego do syslog está excedendo a cota da sua conta. `PutLogEvents`

Para resolver esse problema:

1. Verifique sua cota atual no console Service Quotas em Amazon CloudWatch Logs.
2. Solicite um aumento de cota se a taxa de transferência do syslog exigir. Consulte Solicitando um aumento de cota no Guia do Usuário de Quotas de Serviço.

Usando o CloudWatch Logs com um AWS SDK

AWS kits de desenvolvimento de software (SDKs) estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que permitem que os desenvolvedores criem facilmente aplicações em seu idioma de preferência.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++ exemplos de código
AWS CLI	AWS CLI exemplos de código
AWS SDK para Go	AWS SDK para Go exemplos de código
AWS SDK para Java	AWS SDK para Java exemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript exemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin exemplos de código
AWS SDK para .NET	AWS SDK para .NET exemplos de código
AWS SDK para PHP	AWS SDK para PHP exemplos de código
Ferramentas da AWS para PowerShell	Ferramentas da AWS para PowerShell exemplos de código
AWS SDK para Python v4	AWS Exemplos de código do SDK para Python v4
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) exemplos de código
AWS SDK para Ruby	AWS SDK para Ruby exemplos de código
AWS SDK para Rust	AWS SDK para Rust exemplos de código
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP exemplos de código

Documentação do SDK	Exemplos de código
AWS SDK para Swift	AWS SDK para Swift exemplos de código

Para exemplos específicos de CloudWatch registros, consulteExemplos de código para CloudWatch registros usando AWS SDKs.

> **Exemplo de disponibilidade**
>
> Não consegue encontrar o que precisa? Solicite um exemplo de código usando o link Fornecer feedback na parte inferior desta página.

Gerenciamento de logs

CloudWatch O Logs fornece recursos avançados de gerenciamento de registros que ajudam você a organizar, transformar e analisar seus dados de registro com mais eficiência. Esses recursos incluem centralização de dados entre contas e regiões, descoberta automática de dados e gerenciamento de esquemas, transformação de registros durante a ingestão e análises aprimoradas com facetas para exploração interativa de registros.

Tópicos

- Descoberta e gerenciamento de fontes de dados
- Recursos habilitados por fontes de dados
- Fonte de dados compatíveis

Descoberta e gerenciamento de fontes de dados

CloudWatch O Logs descobre e categoriza automaticamente seus dados de registro por fonte e tipo de dados, facilitando a compreensão e o gerenciamento de seus registros em grande escala. Esse recurso fornece descoberta de esquemas para fontes AWS vendidas, como Amazon VPC Flow Logs e Route 53 CloudTrail, bem como ferramentas de segurança de terceiros.

O console de gerenciamento de registros fornece uma visão de alto nível dos seus registros organizados por fonte e tipo de dados, em vez de apenas grupos de registros. Essa organização ajuda você a:

- Visualize registros categorizados por AWS serviços, fontes de terceiros (como Okta ou CrowdStrike) e fontes personalizadas
- Entenda automaticamente o esquema e a estrutura dos seus dados de registro
- Crie políticas de índice de campo com base nos campos de esquema descobertos
- Gerencie registros com mais eficiência em diferentes fontes de dados
- Registros de consulta por diferentes fontes de dados

Quando você ativa o registro de CloudWatch registros para AWS serviços compatíveis, o CloudWatch Logs aplica automaticamente o esquema apropriado aos seus registros. Esse aplicativo automático de esquema ajuda a manter a consistência e fornece informações imediatas sobre sua estrutura de registros.

O que são fontes de dados de CloudWatch registros?

CloudWatch Fontes de dados de registros é um recurso que fornece uma nova maneira de organizar e categorizar seus dados de registros com base na fonte que gera os registros. Embora o CloudWatch Logs tradicionalmente use grupos de registros para organizar registros, as fontes de dados oferecem uma camada adicional de organização que agrupa os registros por serviço de origem e tipo de registro.

Como as fontes de dados funcionam

As fontes de dados fornecem organização de registros baseada em serviços e descoberta simplificada em toda a sua AWS infraestrutura. Você pode localizar facilmente registros de serviços específicos e filtrar por tipo de registro sem precisar conhecer nomes ou estruturas de grupos de registros individuais.

Para fontes de terceiros e, opcionalmente, para fontes de registros de aplicativos, as fontes de dados trabalham com CloudWatch pipelines para categorizar seus registros. Ao configurar um pipeline para ingerir e transformar registros, você especifica o nome e o tipo da fonte de dados. CloudWatch Em seguida, os registros categorizam automaticamente todos os registros que o pipeline processa. Para obter mais informações, consulte CloudWatch pipelines no Guia do CloudWatch usuário da Amazon.

As fontes de dados categorizam os registros usando dois identificadores principais:

- Nome da fonte de dados: o AWS serviço, a fonte terceirizada ou o aplicativo que gera os registros (por exemplo, Route 53, Amazon VPC CloudTrail, Okta SSO ou Falcon). CrowdStrike
- Tipo de fonte de dados: o tipo específico de registro gerado por esse serviço.

Um esquema define a estrutura dos dados de log, incluindo quais campos estão presentes e como as informações são organizadas. Uma única fonte de dados pode produzir vários tipos de registros com diferentes esquemas e propósitos. Por exemplo, a fonte de AWS CloudTrail dados tem dois tipos: eventos de gerenciamento (que rastreiam as operações do plano de controle, como criar ou excluir recursos) e eventos de dados (que rastreiam as operações do plano de dados, como o acesso a objetos do S3). Cada tipo tem um esquema diferente porque eles capturam diferentes tipos de informações.

Como começar

CloudWatch O Logs categoriza seus registros em fontes de dados com base em sua origem. O método depende do tipo de registros com os quais você está trabalhando:

AWS service (Serviço da AWS) logs

Os registros suportados Serviços da AWS são agrupados automaticamente por fonte de dados sem necessidade de configuração. CloudWatch O Logs reconhece esses registros e aplica o nome e o tipo de fonte de dados apropriados com base no serviço de origem.

Third-party troncos

Third-party os registros exigem pipelines para a categorização da fonte de dados. Ao configurar um pipeline para ingerir registros de fontes de terceiros compatíveis, como Microsoft Office 365, Okta ou Palo Alto Networks CrowdStrike, você especifica o nome e o tipo da fonte de dados na configuração do pipeline. CloudWatch Os registros categorizam automaticamente todos os registros que o pipeline processa usando esses identificadores.

Opcionalmente, os pipelines podem transformar registros de terceiros no formato Open Cybersecurity Schema Framework (OCSF) para análise padronizada de eventos de segurança. Quando a transformação do OCSF está ativada, o nome e o tipo da fonte de dados são determinados automaticamente com base no mapeamento do esquema do OCSF. Sem a transformação do OCSF, você especifica o nome e o tipo da fonte de dados na configuração do pipeline.

Logs de aplicações

Para registros de aplicativos personalizados, você pode categorizá-los por fonte de dados usando um destes métodos:

- Tags de grupo de registros - adicione tags aos seus grupos de registros usando as chaves `cw:datasource:name` e `cw:datasource:type` especifique o nome e o tipo da fonte de dados, respectivamente, para todos os registros ingeridos no grupo de registros. Os valores das tags podem ter até 64 caracteres e conter somente letras minúsculas, números e sublinhado. Eles devem começar com uma letra ou um número e não podem conter sublinhados duplos (__).
- Configuração do pipeline - configure as informações da fonte de dados por meio de pipelines de processamento de registros ao ingerir os registros do seu aplicativo.

> **Note**
>
> Os nomes das fontes de dados não podem começar com “aws” ou “amazon” para evitar conflitos com os registros AWS de serviços.

Campos do sistema

CloudWatch O Logs adiciona automaticamente três campos do sistema aos registros que são categorizados por fonte de dados. Esses campos servem como facetas padrão:

- `@data_source_name`- Contém o nome da fonte de dados ou “Desconhecido” se não for determinado
- `@data_source_type`- Contém o tipo da fonte de dados ou “Desconhecido” se não for determinado
- `@data_format`- Indica o formato dos dados de registro

Quando o nome ou o tipo da fonte de dados não podem ser determinados, esses campos são definidos como “Desconhecido”. As fontes de dados com valores “Desconhecidos” ainda estão visíveis nas facetas e na tabela de fontes de dados em “Gerenciamento de registros” no console, permitindo que você identifique registros não categorizados e de qual grupo de registros eles vêm.

O `@data_format` campo pode conter um dos seguintes valores:

- `Default`- Registros ingeridos sem modificação.
- `Custom`- Registros processados por meio de processadores de pipeline ou registros ingeridos em um grupo de registros com name/type tags de fonte de dados.
- `OCSF-<version>`- Registros processados com processadores OCSF (Open Cybersecurity Schema Framework) em pipelines.
- `AWS-OTEL-LOG-V<version>`- OpenTelemetry registros ingeridos por meio do endpoint CloudWatch OTLP.
- `AWS-OTEL-TRACE-V<version>`- OpenTelemetry traços ingeridos por meio do endpoint CloudWatch OTLP.

Esses campos do sistema permitem filtrar e consultar seus registros com base na origem e no formato, facilitando o trabalho com registros de diferentes origens e pipelines de processamento.

Acessar fontes de dados

Console

No console de CloudWatch registros, você usa a guia Gerenciamento de registros para acessar suas fontes de dados. CloudWatch O Logs consolida automaticamente seus dados de registro por fontes

e tipos de dados, descobrindo continuamente dados recém-ingeridos. Na lista de fontes de dados, você pode criar pipelines, definir índices de campo e facetas.

AWS CLI

Use o comando a seguir para listar fontes de dados e tipos de registros distintos em sua conta:

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

Relacionamento com grupos de registros

As fontes de dados complementam, em vez de substituir, os grupos de registros. Seus registros continuam sendo armazenados em grupos de registros como antes, mas agora eles também são automaticamente marcados com informações da fonte de dados. Essa organização dupla permite que você:

- Use grupos de registros para políticas refinadas de controle de acesso e retenção
- Use fontes de dados para descoberta e análise de registros baseadas em serviços
- Consulte registros usando qualquer um dos métodos organizacionais, dependendo de suas necessidades

As fontes de dados facilitam o trabalho com registros em grande escala, fornecendo uma visão centrada no serviço de seus dados de registro em toda a sua infraestrutura. AWS

Recursos habilitados por fontes de dados

As fontes de dados permitem recursos avançados de processamento e análise de registros por meio de descoberta em campo e estruturas de dados consistentes.

- Facetas: Facetas são campos de registro indexados que fornecem filtragem e análise interativas sem escrever consultas. CloudWatch O Logs cria automaticamente facetas para o nome e o tipo da fonte de dados, e você pode criar políticas de facetas nos campos descobertos para acelerar a solução de problemas. As facetas exibem distribuições e contagens de valores no CloudWatch Logs Insights, facilitando a identificação de padrões por meio da exploração de apontar e escolher.
- Pipelines: crie pipelines de transformação que se apliquem a todos os registros de um nome e tipo de fonte de dados específicos. Isso permite que você defina regras de processamento consistentes para registros da mesma fonte.

- Descoberta de campo: o CloudWatch Logs descobre automaticamente os campos e seus tipos de dados para cada combinação de nome e tipo de fonte de dados com base nos processadores de pipeline. Para registros AWS gerenciados, as estruturas de campo são predefinidas. Para registros de aplicativos, recomendamos manter formatos de log consistentes para maximizar a compatibilidade com ferramentas de análise, como tabelas do Amazon S3, que exigem estruturas de campo bem definidas.

Você pode ver a lista completa de campos e seus tipos para qualquer fonte de dados usando a `GetLogFields` API:

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

Essa descoberta e consistência de campo permitem análises e integrações avançadas, pois ferramentas externas podem trabalhar com estruturas de campo previsíveis ao processar seus dados de log.

Fonte de dados compatíveis

CloudWatch O Logs oferece suporte a uma ampla variedade de fontes de dados, incluindo AWS serviços, ferramentas de segurança e TI de terceiros e fontes de aplicativos personalizadas. Esta página fornece um catálogo abrangente de todas as fontes de dados compatíveis. Para obter informações sobre como as fontes de dados são descobertas e gerenciadas, consultethe section called “Descoberta e gerenciamento de fontes de dados”.

Tópicos

- Compatível Serviços da AWS para fontes de dados
- Fontes de terceiros compatíveis para fontes de dados
- Fontes personalizadas

Compatível Serviços da AWS para fontes de dados

A tabela a seguir lista as Serviços da AWS que são automaticamente categorizadas pelo CloudWatch Logs como fontes de dados:

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
amazon_api_gateway	access
amazon_bedrock_agentcore	browser_usage
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

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
amazon_eks	audit
amazon_eks	authenticator
amazon_eks	controller_manager
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

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
amazon_route53	resolver_query
amazon_sagemaker	workteam_activity
amazon_ses	ingress_endpoints
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

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
aws_cloudtrail	data
aws_cloudtrail	management
aws_elemental_mediapackage	egress_access
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

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
aws_security_hub	compliance_finding
aws_security_hub	data_security_finding
aws_security_hub	detection_finding
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

Fontes de terceiros compatíveis para fontes de dados

A tabela a seguir lista as fontes de terceiros que são automaticamente categorizadas pelo CloudWatch Logs como fontes de dados quando ingeridas por meio de pipelines:

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
akamai_datastream_2	base_event
akamai_datastream_2	dns_activity
akamai_datastream_2	http_activity
box	account_change
box	authentication

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
box	detection_finding
box	entity_management
box	file_hosting_activity
box	group_management
box	user_inventory_info
broadcom_carbonblack	authentication
broadcom_carbonblack	detection_finding
broadcom_carbonblack	file_system_activity
broadcom_carbonblack	module_activity
broadcom_carbonblack	network_activity
broadcom_carbonblack	process_activity
broadcom_carbonblack	registry_key_activity
broadcom_carbonblack	registry_value_activity
broadcom_carbonblack	script_activity
checkpoint_nextgenerationfirewall	data_security_finding
checkpoint_nextgenerationfirewall	detection_finding
checkpoint_nextgenerationfirewall	entity_management

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
checkpoint_nextgenerationfirewall	network_activity
cisco_duo	authentication
cisco_duo	base_event
cisco_duo	device_inventory_info
cisco_duo	entity_management
cisco_duo	user_inventory_info
cisco_ftd	authentication
cisco_ftd	authorize_session
cisco_ftd	detection_finding
cisco_ftd	network_activity
cisco_ftd	tunnel_activity
cisco_ise	account_change
cisco_ise	api_activity
cisco_ise	application_error
cisco_ise	application_lifecycle
cisco_ise	authentication
cisco_ise	entity_management
cisco_ise	network_activity
cisco_ise	tunnel_activity

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
cisco_ise	web_resources_activity
cisco_meraki	api_activity
cisco_meraki	detection_finding
cisco_meraki	network_activity
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
drupal_core	application_lifecycle
drupal_core	authentication
drupal_core	entity_management
drupal_core	http_activity
entrust_idaas	authentication
entrust_idaas	entity_management
f5_bigip	http_activity
f5_bigip	network_activity
github_auditlogs	account_change
github_auditlogs	api_activity

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
github_auditlogs	entity_management
gitlab	group_management
gitlab	software_inventory_info
gitlab	user_inventory_info
gitlab	vulnerability_finding
gitlab	web_resources_activity
google_workspace	account_change
google_workspace	api_activity
google_workspace	authentication
google_workspace	base_event
google_workspace	data_security_finding
google_workspace	detection_finding
google_workspace	email_activity
google_workspace	entity_management
google_workspace	file_hosting_activity
google_workspace	user_access_management
google_workspace	web_resources_activity
hashicorp_vault	api_activity
jamf_pro	authentication
jamf_pro	device_inventory_info

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
jamf_pro	entity_management
jamf_protect	detection_finding
jamf_protect	device_inventory_info
jamf_protect	dns_activity
jamf_protect	software_inventory_info
jamf_protect	vulnerability_finding
jumpcloud	account_change
jumpcloud	authentication
jumpcloud	datastore_activity
jumpcloud	detection_finding
jumpcloud	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

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
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
microsoft_windows	event_log_activity
microsoft_windows	file_system_activity
microsoft_windows	group_management
microsoft_windows	kernel_activity
netskope_cloudexchange	account_change
netskope_cloudexchange	authentication
netskope_cloudexchange	data_security_finding
netskope_cloudexchange	detection_finding
netskope_cloudexchange	device_inventory_info
netskope_cloudexchange	entity_management

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
netskope_cloudexchange	file_hosting_activity
netskope_cloudexchange	network_activity
okta_auth0	api_activity
okta_auth0	authentication
okta_sso	api_activity
okta_sso	authentication
okta_sso	detection_finding
okta_sso	entity_management
onelogin_identity	account_change
onelogin_identity	authentication
onelogin_identity	entity_management
paloaltonetworks_nextgenerationfirewall	authentication
paloaltonetworks_nextgenerationfirewall	detection_finding
paloaltonetworks_nextgenerationfirewall	network_activity
paloaltonetworks_nextgenerationfirewall	process_activity
paloaltonetworks_prismacloud	base_event
paloaltonetworks_prismacloud	cloud_resources_inventory_info

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
paloaltonetworks_prismacloud	compliance_finding
paloaltonetworks_prismacloud	detection_finding
paloaltonetworks_prismacloud	device_inventory_info
paloaltonetworks_prismacloud	entity_management
paloaltonetworks_prismacloud	scan_activity
paloaltonetworks_prismacloud	vulnerability_finding
pingidentity_pingaccess	api_activity
pingidentity_pingaccess	http_activity
pingidentity_pingfederate	authentication
pingidentity_pingfederate	entity_management
pingidentity_pingone	account_change
pingidentity_pingone	authentication
pingidentity_pingone	entity_management
proofpoint_tap	detection_finding
proofpoint_tap	email_activity
proofpoint_tap	osint_inventory_info
qualys_csam	device_inventory_info
qualys_csam	osint_inventory_info
qualys_csam	software_inventory_info
qualys_csam	vulnerability_finding

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
qualys_policyaudit	compliance_finding
qualys_vmdr	account_change
qualys_vmdr	api_activity
qualys_vmdr	authentication
qualys_vmdr	device_inventory_info
qualys_vmdr	entity_management
qualys_vmdr	vulnerability_finding
sailpoint_isc	account_change
sailpoint_isc	authentication
sailpoint_isc	entity_management
sailpoint_isc	user_access_management
salesforce_elf	account_change
salesforce_elf	api_activity
salesforce_elf	authentication
salesforce_elf	datastore_activity
salesforce_elf	detection_finding
salesforce_elf	entity_management
salesforce_elf	file_hosting_activity
salesforce_elf	http_activity
salesforce_rtem	api_activity

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
salesforce_rtem	authentication
salesforce_rtem	datastore_activity
salesforce_rtem	file_hosting_activity
sentinelone_endpointsecurity	dns_activity
sentinelone_endpointsecurity	file_system_activity
sentinelone_endpointsecurity	http_activity
sentinelone_endpointsecurity	process_activity
servicenow_cmdb	api_activity
servicenow_cmdb	datastore_activity
servicenow_cmdb	entity_management
slack_auditlogs	account_change
slack_auditlogs	authentication
slack_auditlogs	detection_finding
slack_auditlogs	entity_management
slack_auditlogs	file_hosting_activity
slack_auditlogs	user_access_management
slack_auditlogs	web_resources_activity
snowflake_platform	authentication
snowflake_platform	datastore_activity
snowflake_platform	entity_management

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
snowflake_platform	group_management
snowflake_platform	user_access_management
snowflake_platform	user_inventory_info
tanium_endpointmanagement	account_change
tanium_endpointmanagement	authentication
tanium_endpointmanagement	compliance_finding
tanium_endpointmanagement	detection_finding
tanium_endpointmanagement	device_inventory_info
tanium_endpointmanagement	entity_management
tanium_endpointmanagement	group_management
tanium_endpointmanagement	vulnerability_finding
tenable_vulnerabilitymanagement	account_change
tenable_vulnerabilitymanagement	api_activity
tenable_vulnerabilitymanagement	authentication
tenable_vulnerabilitymanagement	compliance_finding
tenable_vulnerabilitymanagement	device_inventory_info
tenable_vulnerabilitymanagement	scan_activity
tenable_vulnerabilitymanagement	vulnerability_finding
wazuh_platform	api_activity
wazuh_platform	application_lifecycle

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
wazuh_platform	authentication
wazuh_platform	compliance_finding
wazuh_platform	detection_finding
wazuh_platform	device_inventory_info
wazuh_platform	file_system_activity
wazuh_platform	remediation_activity
wazuh_platform	vulnerability_finding
wiz_cnapp	api_activity
wiz_cnapp	authentication
wiz_cnapp	compliance_finding
wiz_cnapp	detection_finding
wiz_cnapp	vulnerability_finding
zeek	authentication
zeek	base_event
zeek	detection_finding
zeek	dhcp_activity
zeek	dns_activity
zeek	email_activity
zeek	ftp_activity
zeek	http_activity

Nome da fonte de dados (campo @data_source_name)	Tipo de fonte de dados (campo @data_source_type)
zeek	network_activity
zeek	rdp_activity
zeek	smb_activity
zeek	software_inventory_info
zeek	ssh_activity
zeek	tunnel_activity
zscaler_internetaccess	authentication
zscaler_internetaccess	dns_activity
zscaler_internetaccess	http_activity
zscaler_internetaccess	network_activity

Fontes adicionais de terceiros via AWS CSPM do Security Hub

Outras descobertas de segurança de terceiros estão disponíveis por meio da integração com o CSPM do AWS Security Hub. Os parceiros a seguir enviam as descobertas para o CSPM do Security Hub, que são então disponibilizadas como fontes de dados no CloudWatch Logs. Para obter detalhes abrangentes sobre essas integrações, consulte integrações de Third-party produtos com o CSPM do Security Hub no Guia do Usuário do AWS Security Hub.

Parceiro	Integração
3 núcleos SEC — NTA	Envia descobertas por meio do Security Hub CSPM
Lógica de alertas — gerenciamento de ameaças sem fio	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Aqua Security — Plataforma de segurança nativa em nuvem	Envia descobertas por meio do Security Hub CSPM
Aqua Security — Kube-bench	Envia descobertas por meio do Security Hub CSPM
Armadura — Armadura em qualquer lugar	Envia descobertas por meio do Security Hub CSPM
AttackIQ	Envia descobertas por meio do Security Hub CSPM
Barracuda Networks — Guardião da segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
BigID — BigID Enterprise	Envia descobertas por meio do Security Hub CSPM
Hexágono azul	Envia descobertas por meio do Security Hub CSPM
Check Point — CloudGuard IaaS	Envia descobertas por meio do Security Hub CSPM
Check Point — CloudGuard Gestão da postura	Envia descobertas por meio do Security Hub CSPM
Clarity — Dome	Envia descobertas por meio do Security Hub CSPM
Segurança de armazenamento em nuvem — Antivírus para Amazon S3	Envia descobertas por meio do Security Hub CSPM
Segurança de contraste — avaliação de contraste	Envia descobertas por meio do Security Hub CSPM
CrowdStrike — CrowdStrike Falcão	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
CyberArk — Análise privilegiada de ameaças	Envia descobertas por meio do Security Hub CSPM
Teorema dos dados	Envia descobertas por meio do Security Hub CSPM
Drata	Envia descobertas por meio do Security Hub CSPM
Forcepoint — CASB	Envia descobertas por meio do Security Hub CSPM
Forcepoint — Gateway de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
Forcepoint — DLP	Envia descobertas por meio do Security Hub CSPM
Forcepoint — NGFW	Envia descobertas por meio do Security Hub CSPM
Fuga	Envia descobertas por meio do Security Hub CSPM
Guardicore — Central	Envia descobertas por meio do Security Hub CSPM
HackerOne — Inteligência de vulnerabilidade	Envia descobertas por meio do Security Hub CSPM
jFrog — Raio X	Envia descobertas por meio do Security Hub CSPM
Juniper Networks — Firewall vSRX de próxima geração	Envia descobertas por meio do Security Hub CSPM
k9 Security — Analisador de acesso	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Rendas	Envia descobertas por meio do Security Hub CSPM
McAfee — MVISION CNAPP	Envia descobertas por meio do Security Hub CSPM
NETSCOUT — Investigador cibernético	Envia descobertas por meio do Security Hub CSPM
Orca — Plataforma de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
Palo Alto Networks — Computação em nuvem Prisma	Envia descobertas por meio do Security Hub CSPM
Palo Alto Networks — Prisma Cloud Enterprise	Envia descobertas por meio do Security Hub CSPM
Plerion — Plataforma de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
Gatuno	Envia descobertas por meio do Security Hub CSPM
Qualys — Gerenciamento de vulnerabilidades	Envia descobertas por meio do Security Hub CSPM
Rapid7 — InsightVM	Envia descobertas por meio do Security Hub CSPM
SentinelOne	Envia descobertas por meio do Security Hub CSPM
Snyk	Envia descobertas por meio do Security Hub CSPM
Segurança Sonrai — Sonrai Dig	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Sophos — Proteção de servidor	Envia descobertas por meio do Security Hub CSPM
StackRox — Segurança do Kubernetes	Envia descobertas por meio do Security Hub CSPM
Sumo Logic — Análise de dados de máquinas	Envia descobertas por meio do Security Hub CSPM
Symantec — Proteção da carga de trabalho na nuvem	Envia descobertas por meio do Security Hub CSPM
Tenable.io	Envia descobertas por meio do Security Hub CSPM
Trend Micro: Cloud One	Envia descobertas por meio do Security Hub CSPM
Vectra — Cognito Detect	Envia descobertas por meio do Security Hub CSPM
Wiz	Envia descobertas por meio do Security Hub CSPM
Caveonix — Caveonix Cloud	Envia e recebe descobertas por meio do Security Hub CSPM
Cloud Custodian	Envia e recebe descobertas por meio do Security Hub CSPM
DisruptOps	Envia e recebe descobertas por meio do Security Hub CSPM

Parceiro	Integração
Leão	Envia e recebe descobert as por meio do Security Hub CSPM
Turbot	Envia e recebe descobert as por meio do Security Hub CSPM

> **Note**
>
> Essa lista reflete as integrações de parceiros do Security Hub que enviam descobertas no momento da redação. Como o AWS Security Hub adiciona regularmente novas integrações de parceiros, consulte as integrações de Third-party produtos com o CSPM do Security Hub no Guia do Usuário do AWS Security Hub para obter a lista mais atualizada dos parceiros disponíveis.

Fontes personalizadas

Para registros de aplicativos personalizados, você pode definir sua própria categorização da fonte de dados usando um dos seguintes métodos:

- Tags de grupo de registros — adicione tags aos seus grupos de registros usando as chaves `cw:datasource:name` e `cw:datasource:type` especifique o nome e o tipo da fonte de dados para todos os registros ingeridos no grupo de registros.
- Configuração do pipeline — configure as informações da fonte de dados por meio de pipelines de processamento de registros ao ingerir os registros do seu aplicativo.

> **Note**
>
> Os nomes de fontes de dados personalizados não podem começar com “aws” ou “amazon” para evitar conflitos com os registros AWS de serviços.

Análise de dados de registro com o CloudWatch Log Analytics

O Log Analytics é uma experiência de console unificada no Amazon CloudWatch Logs que reúne recursos de análise de registros em um só lugar. Você pode consultar e analisar dados de registro com o CloudWatch Logs Insights, transmitir registros em tempo real com o Live Tail e identificar seus principais CloudWatch colaboradores com o Contributor Insights. O Log Analytics é a experiência padrão para analisar registros no console de CloudWatch registros.

O que você pode fazer com o Log Analytics

O Log Analytics reúne os seguintes recursos:

- Consulte e analise dados de registro — execute consultas em seus grupos de CloudWatch registros com o Logs Insights. Você pode executar várias consultas em guias diferentes, salvar e reutilizar consultas com parâmetros, explorar seus dados com facetas, gerar consultas usando linguagem natural e visualizar resultados. Para obter mais informações, consulte Análise de dados de registro com o CloudWatch Logs Insights.
- Transmita registros quase em tempo real — Use o Live Tail para ver uma lista de streaming dos eventos de registro à medida que são ingeridos e filtrar ou destacar os eventos que são importantes para você. Para obter mais informações, consulte Solução de problemas com o CloudWatch Logs Live Tail.
- Identifique os principais contribuidores — Use o CloudWatch Contributor Insights para analisar dados de registro de alta cardinalidade e ver os colaboradores que têm o maior impacto em seu sistema. Para obter mais informações, consulte Usar o Contributor Insights para analisar dados de alta cardinalidade.

Comece a usar o Log Analytics

Para usar o Log Analytics, faça login no CloudWatch console e escolha Log Analytics em Logs no painel de navegação.

O Log Analytics é a experiência padrão. Se você optar por não participar, poderá continuar usando o CloudWatch Logs Insights, o Live Tail e o Contributor Insights como experiências distintas junto com o Log Analytics e poderá voltar a qualquer momento para o Log Analytics.

O Log Analytics usa o mesmo preço de seus recursos subjacentes: consultas do CloudWatch Logs Insights, Live Tail e Contributor Insights. Para obter detalhes de preço, consulte Definição de preço do CloudWatch .

Análise de dados de registro com o CloudWatch Logs Insights

Com o CloudWatch Logs Insights, você pode pesquisar e analisar interativamente seus dados de log no Amazon CloudWatch Logs. Realize consultas para ajudar a responder de maneira mais rápida e eficiente a problemas operacionais. Além de consultar usando grupos de registros, você pode consultar usando facetas, fonte de dados e tipo de dados. Se ocorrer um problema, você pode usar o CloudWatch Logs Insights para identificar possíveis causas e validar as correções implantadas. Você está limitado a 100 CloudWatch Logs Insights QL simultâneos por conta, incluindo consultas adicionadas aos painéis. Além disso, você pode executar 15 consultas simultâneas para OpenSearch Service PPL ou OpenSearch Service SQL.

CloudWatch O Logs Insights oferece suporte a três linguagens de consulta que você pode usar para suas consultas:

- Uma linguagem de consulta do Logs Insights (Logs Insights QL) criada especificamente com alguns comandos simples, mas poderosos.
- OpenSearch Linguagem de processamento de serviços canalizados (PPL). OpenSearch O PPL permite que você analise seus registros usando um conjunto de comandos delimitados por canais (|).

  Com o OpenSearch PPL, você pode recuperar, consultar e analisar dados usando comandos que são agrupados, facilitando a compreensão e a composição de consultas complexas. A sintaxe permite o encadeamento de comandos para transformar e processar dados. Com a PPL, você pode filtrar e agregar dados e usar um amplo conjunto de funções matemáticas, de sequências de caracteres, de data, condicionais e outras para análise.
- OpenSearch Linguagem de consulta estruturada de serviços (SQL). Com as consultas OpenSearch SQL, você pode analisar seus registros de forma declarativa. Você pode usar comandos como SELECT, FROM, WHERE, GROUP BY, HAVING e vários outros comandos e funções disponíveis no SQL. Você pode executar JOINs em grupos de logs, correlacionar dados entre logs usando sub-consultas e usar o amplo conjunto de funções JSON, de matemáticas, de sequência de caracteres, condicionais e outras funções SQL para realizar análises poderosas em logs.

  Ao usar comandos SQL ou PPL, certifique-se de colocar entre crases os campos com caracteres especiais (não alfabéticos e não numéricos) para que a consulta seja bem-sucedida. Por exemplo,

coloque `@message`, `Operation.Export` e `Test::Field` entre crases. Você não precisa colocar entre crases os campos com nomes puramente alfabéticos.

CloudWatch O Logs Insights oferece os seguintes recursos que estão disponíveis para uso com qualquer uma das linguagens de consulta.

- Descoberta automática de campos de log em registros de AWS serviços como Amazon Route 53,, AWS Lambda AWS CloudTrail, e Amazon VPC, e qualquer aplicativo ou log personalizado que emita eventos de log como JSON.
- Criação de índices de campo para reduzir custos e acelerar os resultados, especialmente para consultas de um grande número de grupos de logs ou eventos de logs. Depois de criar índices de campos que são comuns em seus eventos de log, você pode usá-los em uma consulta. A consulta ignora o processamento de eventos de logs conhecidos por não incluir o campo indexado e processar menos dados.

  > **Note**
  >
  > O comando `filterIndex` está disponível somente no Logs Insights QL.

- Detecção e análise de padrões em seus eventos de logs. Um padrão é uma estrutura de texto compartilhada que se repete nos campos do log. Ao visualizar os resultados de uma consulta, você pode escolher a guia Padrões para ver os padrões que o CloudWatch Logs encontrou com base em uma amostra dos seus resultados.
- Salvando consultas, vendo seu histórico de consultas, reexecutando consultas salvas e usando consultas salvas com parâmetros.
- Adicionar consultas aos painéis.
- Criptografando os resultados da consulta com AWS Key Management Service.
- A geração de consultas usando linguagem natural permite que você use linguagem natural para criar consultas do CloudWatch Logs Insights. Você pode fazer perguntas ou descrever os dados que está procurando e, em seguida, a IA gera uma consulta com base no seu prompt e fornece uma explicação detalhada de como a consulta funciona.
- Use facetas para agrupar, filtrar e explorar interativamente seus registros.
- Os registros circundantes permitem que você visualize as linhas de registro antes e depois de qualquer registro de registro específico nos resultados da consulta para obter contexto instantâneo sobre eventos de registro críticos. Você pode configurar o intervalo para visualizar 5, 10, 20, 50

ou 100 linhas de registro antes e depois de um registro selecionado e pesquisar palavras-chave específicas nos registros ao redor.

Os seguintes recursos do CloudWatch Logs Insights são compatíveis somente quando você usa o Logs Insights QL.

- Consultas de comparação que comparam eventos de logs em um grupo de logs com eventos de logs de um período anterior.

> **Important**
>
> CloudWatch O Logs Insights não pode acessar eventos de registro com carimbos de data e hora anteriores à hora de criação do grupo de registros.

Se você estiver conectado a uma conta configurada como uma conta de monitoramento na observabilidade CloudWatch entre contas, poderá executar consultas do CloudWatch Logs Insights em grupos de registros nas contas de origem vinculadas a essa conta de monitoramento. Você pode executar uma consulta que pesquisa vários grupos de logs localizados em contas diferentes. Para obter mais informações, consulte Observabilidade entre contas do CloudWatch .

Ao criar consultas usando o Logs Insights QL, você também pode usar linguagem natural para criar consultas do CloudWatch Logs Insights. Para fazer isso, faça perguntas ou descreva os dados que você está procurando. Esse AI-assisted recurso gera uma consulta com base em sua solicitação e fornece uma explicação linha por linha de como a consulta funciona. Para obter mais informações, consulte Usar linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights.

O tempo limite das consultas que usam qualquer linguagem de consulta compatível é de 60 minutos caso não sejam concluídas. Os resultados da consulta ficam disponíveis por 7 dias.

CloudWatch As consultas do Logs Insights são cobradas com base na quantidade de dados de registro não compactados digitalizados, independentemente do idioma da consulta. Para obter mais informações, consulte Amazon CloudWatch Pricing.

Você pode usar o CloudWatch Logs Insights para pesquisar dados de registro enviados ao CloudWatch Logs em 5 de novembro de 2018 ou mais tarde.

> **Important**
>
> Se sua equipe de segurança de rede não permite o uso de soquetes da web, atualmente você não pode acessar a parte do CloudWatch Logs Insights do CloudWatch console. Você pode usar os recursos de consulta do CloudWatch Logs Insights usando APIs. Para obter mais informações, consulte StartQuerya Referência da API Amazon CloudWatch Logs.

Conteúdo

- Linguagens de consulta compatíveis
- Use linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights
- Logs compatíveis e campos descobertos
- Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento
- Use facetas para agrupar e explorar registros
- Visualizando registros ao redor no CloudWatch Logs Insights
- Análise de padrões
- Salve e execute novamente as consultas do CloudWatch Logs Insights
- Adicionar consulta ao painel ou exportar os resultados da consulta
- Exibir as consultas em execução ou o histórico de consultas
- Criptografe os resultados da consulta com AWS Key Management Service
- Gere um resumo em linguagem natural a partir dos resultados da consulta do CloudWatch Logs Insights

Linguagens de consulta compatíveis

As seções a seguir listam os comandos compatíveis com cada linguagem de consulta. Elas também descrevem o formato da sintaxe e fornecem exemplos de consultas.

Tópicos

- CloudWatch Linguagem de consulta do Logs Insights (Logs Insights QL)
- OpenSearch Linguagem de processamento canalizado (PPL)
- OpenSearch Linguagem de consulta estruturada (SQL)

CloudWatch Linguagem de consulta do Logs Insights (Logs Insights QL)

Esta seção inclui a documentação completa dos comandos e funções do Logs Insights QL. Também inclui exemplos de consultas para essa linguagem.

Para obter informações sobre outras linguagens de consulta que você pode usar, consulte OpenSearch Service PPL, OpenSearch Service SQL e. CloudWatch Metrics Insights

Tópicos

- CloudWatch Sintaxe de consulta de linguagem do Logs Insights
- Comece a usar o Logs Insights QL: tutoriais de consulta
- Exemplos de consultas
- Compare (diff) com intervalos de tempo anteriores
- Visualize dados de log em grafos

CloudWatch Sintaxe de consulta de linguagem do Logs Insights

Esta seção fornece detalhes sobre o Logs Insights QL. A sintaxe de consulta é compatível com diferentes funções e operações que incluem, mas não se limitam a funções gerais, operações aritméticas e de comparação e expressões regulares.

> **Important**
>
> Para evitar custos excessivos ao executar consultas grandes, lembre-se das seguintes práticas recomendadas:
>
> - Selecione apenas os grupos de logs necessários para cada consulta.
> - Sempre especifique o menor intervalo de tempo possível para suas consultas.
> - Ao usar o console para executar consultas, cancele todas as consultas antes de fechar a página do console do CloudWatch Logs Insights. Caso contrário, as consultas continuarão sendo executadas até serrem concluídas.
> - Ao adicionar um widget do CloudWatch Logs Insights a um painel, certifique-se de que o painel não seja atualizado com alta frequência, pois cada atualização inicia uma nova consulta.

Para criar consultas contendo vários comandos, separe-os com o caractere de barra vertical (|).

Para criar consultas contendo comentários, marque-os com o caractere de grade (#).

> **Note**
>
> CloudWatch O Logs Insights descobre automaticamente campos para diferentes tipos de registro e gera campos que começam com o caractere @. Para obter mais informações sobre esses campos, consulte Registros suportados e campos descobertos no Guia CloudWatch do usuário da Amazon.

A tabela a seguir descreve resumidamente cada comando. Em seguida, há uma descrição mais abrangente de cada comando com exemplos.

> **Note**
>
> Todos os comandos de consulta do Logs Insights QL são compatíveis com os grupos de logs da classe de logs Padrão. Os grupos de logs da classe de logs de Acesso infrequente são compatíveis com todos os comandos de consulta do Logs Insights QL, exceto `pattern`, `diff` e `unmask`.

addtotals	Calcula totais de linhas e totais de colunas para campos numéricos.
anomaly	Identifica padrões incomuns em dados de log usando machine learning.
display	Exibe um ou mais campos específicos em resultados de consultas.
fields	Exibe campos específicos em resultados de consultas e oferece suporte a funções e operações que podem ser usadas para modificar valores de campo e criar novos campos para uso em consultas.
filter	Filtra a consulta para retornar apenas os eventos de log que correspon dem a uma ou mais condições.
filterIndex	Força uma consulta a tentar verificar somente os grupos de registros que estão indexados no campo mencionado em um índice de campo e que também contêm um valor para esse índice de campo. Isso reduz o volume escaneado ao tentar verificar somente eventos de logs desses

	grupos que contêm o valor especificado na consulta para esse índice de campo. Esse comando não é compatível com os grupos de logs da classe de logs de Acesso infrequente.
pattern	Agrupa automaticamente seus dados de log em padrões. Um padrão é uma estrutura de texto compartilhada que se repete entre seus campos de registro. CloudWatch O Logs Insights fornece maneiras de analisar os padrões encontrados em seus eventos de registro. Para obter mais informações, consulte Análise de padrões.
diff	Compara os eventos de log encontrados no período solicitado com os eventos de log de um período anterior de igual duração, para que você possa procurar tendências e descobrir se determinados eventos de log são novos.
parse	Extrai dados de um campo de log para criar um campo extraído que pode ser processado na consulta. **parse** é compatível tanto com o modo glob usando curingas quanto com as expressões regulares.
sort	Exibe os eventos de log retornados em ordem crescente (`asc`) ou decrescente (`desc`).
SOURCE	Incluir SOURCE em uma consulta é uma forma útil de especificar uma grande quantidade de grupos de registros a serem incluídos em uma consulta com base no prefixo do nome do grupo de registros, nos identificadores de conta, na classe do grupo de registros, nas fontes de dados ou nas tags do grupo de registros. Esse comando é suportado somente quando você cria uma consulta no AWS CLI ou programat icamente, não no CloudWatch console.
stats	Calcule estatísticas agregadas usando valores nos campos de log.
limit	Especifica um número máximo de eventos de log que você deseja que sua consulta retorne. Útil com **sort** para retornar resultados “20 principais” ou “20 mais recentes”. Use `limit any` para interromper o escaneamento mais cedo quando forem encontrados resultados suficient es.

dedup	Remove resultados duplicados com base em valores específicos em campos que você especifica.
unmask	Exibe todo o conteúdo de um evento de log que tenha algum conteúdo mascarado devido à política de proteção de dados. Para mais informaçõ es sobre a proteção de dados nos grupos de logs, acesse Ajude a proteger dados de log confidenciais com mascaramento.
unnest	Nivela uma lista usada como entrada para gerar vários registros com um único registro para cada elemento na lista.
lookup	Enriquece os eventos de log com dados de uma tabela de pesquisa combinando os valores dos campos. Use tabelas de pesquisa para adicionar dados de referência, como detalhes do usuário, nomes de aplicativos ou informações do produto, aos resultados da consulta.
join	Combina eventos de log de um grupo de log de origem com eventos de outro grupo de log ou resultado de consulta com base em um campo correspondente. Use o comando join para correlacionar eventos de log relacionados em diferentes fontes usando chaves comuns entre elas, como identificadores de solicitação ou IDs de transação correspon dentes.
subqueries	Uma subconsulta é uma consulta aninhada do Logs Insights que pode ser usada como entrada para outra consulta. As subconsultas podem ser usadas para derivar conjuntos de resultados intermediários que são então consumidos pelos comandos subsequentes.
autoregress	Cria cópias com defasagem (linha anterior) dos valores de um campo.
logcompare	Compara a janela de tempo atual com uma janela de linha de base alterada por uma duração.
filldown	Transmite o último valor não nulo para preencher as lacunas.
fillmissing	Insere linhas para compartimentos de tempo vazios após stats by bin (), preenchendo opcionalmente os campos com uma constante.

cidrlookup	Enriquece os eventos combinando um campo IP com intervalos CIDR em uma tabela de pesquisa.
outlier	Detecta valores discrepantes estatísticos com base no intervalo interquartil (IQR) e pode remover ou transformar linhas discrepantes.
accum	Calcula uma soma cumulativa contínua de um campo numérico.
appendcols	Acrescenta colunas de uma subconsulta aos resultados atuais por correspondência de linhas posicionais.
sessionize	Agrupa eventos em sessões por campos de identidade e lacuna de inatividade.
countFrequent	Retorna uma contagem aproximada de cada combinação exclusiva de valor de campo, classificada em ordem decrescente.
where	Atua como um alias gramatical para o `filter` comando e aceita sintaxe idêntica.
estimativa	Retorna os bytes estimados que a consulta examinaria nos grupos de registros e no intervalo de tempo selecionados, sem executar a consulta.
Outras operações e funções	CloudWatch O Logs Insights também oferece suporte a muitas funções e operações de comparação, aritmética, data e hora, numéricas, seqüências de caracteres, endereços IP e funções e operações gerais.

As seções a seguir fornecem mais detalhes sobre os comandos de consulta do CloudWatch Logs Insights.

Tópicos

- Comandos do Logs Insights QL compatíveis com as classes de log
- adicionar totais
- anomalia
- display
- fields
- filtrar

- filterIndex
- SOURCE
- pattern
- diferença
- parse
- campos relevantes
- expandir
- sort
- stats
- limite
- dedup
- unmask
- unnest
- lookup
- ingressar
- subconsultas
- regressão automática
- logcompare
- preenchimento
- preenchimento ausente
- consulta de cidra
- outlier
- accum
- apêndices
- sessionizar
- Contagem frequente
- para onde
- estimativa
- Funções booleanas, de comparação, numéricas e de data e hora, entre outras
- Funções de hashing

- Time-series funções
- Campos contendo caracteres especiais
- Usar aliases e comentários em consultas

Comandos do Logs Insights QL compatíveis com as classes de log

Todos os comandos de consulta do Logs Insights QL são compatíveis com os grupos de logs da classe de logs Padrão. Os grupos de logs da classe de logs de Acesso infrequente são compatíveis com todos os comandos de consulta, exceto `pattern`, `diff`, `filterIndex` e `unmask`.

adicionar totais

Use `addtotals` para calcular totais de linhas e totais de colunas para campos numéricos. Os totais das linhas são calculados por bloco. Os totais das colunas (`col=true`) são calculados após a mesclagem para verificar a exatidão em partes distribuídas. Você pode usar a seleção de campo personalizada, um nome de campo personalizado e `row=false` desativar os totais das linhas.

Sintaxe

```
| addtotals [fieldname=Name] [row=bool] [col=bool] [field1, field2, ...]
```

- `fieldname=`*`Name`*— Define o nome da coluna total da linha. O padrão é `Total`.
- `row=`*`bool`*— Ativar ou desativar os totais das linhas. O padrão é `true`.
- `col=`*`bool`*— Ative ou desative os totais das colunas, o que adiciona uma linha de resumo na parte inferior. O padrão é `false`.
- *`field1, field2, ...`*— Lista opcional de campos específicos a serem somados. Se omitido, todos os campos numéricos são somados.

Exemplos

```
# Row totals (default) - adds a Total column summing all numeric fields
fields a, b
| addtotals
```

```
# Sum only specific fields
fields price, tax, qty
| addtotals price, tax
```

```
# Custom total column name
fields val
| addtotals fieldname=RowSum
```

```
# Column totals - adds a Summary row at the bottom
fields x
| addtotals col=true x
```

anomalia

Use `anomaly` para identificar automaticamente padrões incomuns e possíveis problemas em seus dados de log usando machine learning.

O `anomaly` comando amplia a `pattern` funcionalidade existente e usa análises avançadas para ajudar a identificar possíveis anomalias nos dados de registro. Você pode usar `anomaly` para reduzir o tempo necessário para identificar e resolver problemas operacionais, revelando automaticamente padrões ou comportamentos incomuns em seus logs.

O comando `anomaly` trabalha com o comando `pattern` para, primeiramente, identificar padrões de log e, em seguida, detectar anomalias nesses padrões. Você também pode combinar `anomaly` com os comandos `filter` ou `sort` e concentrar a detecção de anomalias em subconjuntos específicos de seus dados.

Entrada de comando de anomalia

Normalmente, o comando `anomaly` é usado após o comando `pattern` para analisar os padrões identificados em seus dados de log. O comando não exige parâmetros adicionais e analisa a saída dos comandos anteriores em sua consulta.

Tipos de anomalias identificadas

O comando `anomaly` identifica cinco tipos distintos de anomalias:

- Anomalias de frequência de padrões: frequências incomuns de padrões de log específicos, como quando um aplicativo começa a gerar mais mensagens de erro do que o normal.
- Anomalias de padrões novos: padrões de log inéditos que podem indicar novos tipos de erros ou mensagens aparecendo em seus logs.
- Anomalias de variação de token: mudanças inesperadas no conteúdo das mensagens de log que podem indicar variações incomuns nos formatos de log esperados.

- Anomalias de tokens numéricos: mudanças incomuns nos valores numéricos nos logs que podem ajudar a detectar possíveis problemas de desempenho ou variações métricas inesperadas.
- Anomalias do código de erro HTTP: padrões relacionados às respostas de erro HTTP, particularmente úteis ao monitorar aplicativos web e APIs.

Saída do comando de anomalia

O comando `anomaly` preserva todos os campos dos dados de entrada e adiciona resultados de detecção de anomalias para ajudar a identificar padrões incomuns em seus dados de log.

Exemplos

O seguinte comando identifica padrões em seus dados de log e, em seguida, detecta anomalias nesses padrões:

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

O comando `anomaly` pode ser usado com filtragem para se concentrar em tipos de log específicos:

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

O comando `anomaly` pode ser combinado com a classificação para organizar os resultados:

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

display

Use `display` para mostrar um ou mais campos específicos nos resultados da consulta.

O comando `display` mostra apenas os campos especificados. Se sua consulta contiver vários comandos `display`, os resultados da consulta mostram apenas o campo ou os campos que você especificou no comando `display` final.

Exemplo: Exibir um campo

O trecho de código mostra um exemplo de uma consulta que usa o comando parse para extrair dados de `@message` para criar os campos extraídos `loggingType` e `loggingMessage`. A consulta retorna eventos de log em que os valores para `loggingType` são ERROR. `display` mostra somente os valores para `loggingMessage` nos resultados da consulta.

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

> **Tip**
>
> Use `display` somente uma vez em uma consulta. Se você usar `display` mais de uma vez em uma consulta, os resultados da consulta mostram apenas os campos especificados na última ocorrência em que o comando `display` for usado.

fields

Use `fields` para mostrar campos específicos nos resultados da consulta.

Se sua consulta contiver vários comandos `fields` e não incluir um comando `display`, os resultados exibirão todos os campos especificados nos comandos `fields`.

Exemplo: Exibir campos específicos

O exemplo a seguir mostra uma consulta que retorna 20 eventos de logs e os exibe em ordem decrescente. Os valores para `@timestamp` e `@message` são mostrados nos resultados da consulta.

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

Use `fields` em vez de `display` quando não quiser usar as diferentes funções e operações compatíveis com o `fields` para modificar valores de campo e criar novos campos que podem ser usados em consultas.

É possível usar o comando `fields` com a palavra-chave as para criar campos extraídos que usam os campos e funções de seus eventos de log. Por exemplo, `fields ispresent as isRes`

cria um campo extraído chamado `isRes`, e o campo extraído pode ser usado no restante de sua consulta.

filtrar

Use `filter` para obter eventos de log que correspondam a uma ou mais condições.

Exemplo: Filtrar eventos de log usando uma condição

O trecho de código mostra um exemplo de uma consulta que retorna todos os eventos de log em que o valor de `range` é maior do que 3000. A consulta limita os resultados a 20 eventos de log e classifica os eventos de registros por `@timestamp` e em ordem decrescente.

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

Exemplo: Filtrar eventos de log usando mais de uma condição

Você pode usar as palavras-chave `and` e `or` para combinar mais de uma condição.

O trecho de código mostra um exemplo de consulta que retorna todos os eventos de log em que o valor de `range` é maior do que 3.000 e o valor de `accountId` é igual a 123.456.789.012. A consulta limita os resultados a 20 eventos de log e classifica os eventos de registros por `@timestamp` e em ordem decrescente.

```
fields @timestamp, @message
| filter (range>3000 and accountId=123456789012)
| sort @timestamp desc
| limit 20
```

Campos indexados e o comando de filtro

Se você criou índices de campo para um grupo de registros, pode usar esses índices de campo para tornar suas `filter` consultas mais eficientes e reduzir o volume digitalizado. Por exemplo, suponha que você tenha criado um índice de campo para `requestId`. Em seguida, qualquer consulta do CloudWatch Logs Insights nesse grupo de registros que `filter requestId IN [`*`value`*`, `*`value`*`, ...]` inclua `filter requestId =` *`value`* ou tente ignorar o processamento de eventos

de log que se sabe que não incluem o campo indexado. Ao tentar verificar somente os eventos de log conhecidos por conter esse campo indexado, o volume de escaneamento pode ser reduzido, e a consulta é mais rápida.

Para obter mais informações sobre os índices de campo e como criá-los, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.

> **Important**
>
> Somente consultas com `filter` *`fieldName`* `=...` e `filter` *`fieldName`* `IN...` se beneficiarão das melhorias no índice de campo. As consultas com `filter` *`fieldName`* `like` não usam índices e sempre examinam todos os eventos de logs nos grupos de logs selecionados.

Exemplo: encontre eventos de logs relacionados a um determinado ID de solicitação usando índices

Este exemplo pressupõe que você criou um índice de campo no `requestId`. Para grupos de registros que usam esse índice de campo, a consulta usará índices de campo para tentar verificar a menor quantidade de eventos de registro para encontrar eventos `requestId` com um valor de `123456`

```
fields @timestamp, @message
| filter requestId = "1234656"
| limit 20
```

Correspondências e expressões regulares no comando de filtro

O comando filtrar é compatível com o uso de expressões regulares. Você pode usar os seguintes operadores de comparação (=, `!=`, <, <=, >, >=) e operadores boolianos (`and`, `or` e `not`).

Você pode usar a palavra-chave `in` para testar a associação do conjunto e verificar se há elementos em uma matriz. Para verificar elementos em uma matriz, coloque a matriz após `in`. É possível usar o operador booliano `not` com `in`. Você pode criar consultas que usam `in` para retornar eventos de log nos quais os campos são correspondências de string. Os campos devem ser strings completas. Por exemplo, o trecho de código a seguir mostra uma consulta que usa `in` para retornar eventos de log nos quais o campo `logGroup` é a string `example_group` completa.

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

```
| filter logGroup in ["example_group"]
```

Você pode usar as frases de palavras-chave `like` e `not like` para combinar substrings. Você pode usar o operador de expressão regular =~ para combinar substrings. Para fazer a correspondência de uma substring com `like` e `not like`, envolva entre aspas duplas ou simples a substring que deseja corresponder. Você pode usar padrões de expressão regular com `like` e `not like`. Para fazer a correspondência de uma substritng com o operador de expressão regular, envolva entre barras a substring a ser correspondida. Os exemplos a seguir contêm trechos de código que mostram como você pode fazer a correspondências de substrings usando o comando `filter`.

Exemplos: correspondência de substrings

Os exemplos a seguir retornam eventos de log quando `f1` contém a palavra Exception. Os três exemplos fazem distinção entre maiúsculas e minúsculas.

O primeiro exemplo faz a correspondência de uma substring com `like`.

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

O segundo exemplo faz a correspondência de uma substring com `like` e um padrão de expressão regular.

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

O terceiro exemplo faz a correspondência de uma substring com uma expressão regular.

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

Exemplo: fazer a correspondência de substrings com curingas

Você pode usar o símbolo de ponto (`.`) como um curinga em expressões regulares para fazer a correspondência com substrings. No exemplo a seguir, a consulta retorna correspondências em que o valor de `f1` começa com a string `ServiceLog`.

```
fields f1, f2, f3
```

```
| filter f1 like /ServiceLog./
```

Você pode colocar um símbolo de asterisco depois do ponto (`.*`) para criar um quantificador ganancioso que retorna o maior número possível de correspondências. No exemplo a seguir, a consulta retorna correspondências em que o valor de `f1` não somente começa com a string `ServiceLog`, mas também inclui a string `ServiceLog`.

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

As correspondências possíveis podem ser formatadas da seguinte forma:

- `ServiceLogSampleApiLogGroup`
- `SampleApiLogGroupServiceLog`

Exemplo: excluir substrings das correspondências

O exemplo a seguir mostra uma consulta que retorna eventos de log quando `f1` não contém a palavra Exception. O exemplo diferencia minúsculas e maiúsculas.

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

Exemplo: fazer a correspondência de subcadeias de caracteres com padrões que não diferenciam maiúsculas e minúsculas

Você pode fazer a correspondência de subcadeias de caracteres que não diferenciam minúsculas e maiúsculas com `like` e expressões regulares. Coloque o seguinte parâmetro (?i) antes da substring a ser correspondida. O exemplo a seguir mostra uma consulta que retorna eventos de log quando `f1` contém a palavra Exception ou exception.

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

filterIndex

Use `filterIndex` para retornar somente dados indexados, forçando uma consulta a verificar somente grupos de logs indexados em um campo especificado na consulta. Para esses grupos

de logs indexados nesse campo, ele otimiza ainda mais a consulta ignorando os grupos que não têm nenhum evento de logs contendo o campo especificado na consulta do campo indexado. Isso reduz ainda mais o volume escaneado ao tentar verificar somente eventos de logs desses grupos que correspondam ao valor especificado na consulta para esse índice de campo. Para obter mais informações sobre os índices de campo e como criá-los, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.

O uso de `filterIndex` com campos indexados pode ajudar você a consultar grupos de logs que incluem petabytes de dados de log de forma eficiente, limitando o espaço de pesquisa real a grupos e eventos de logs que têm índices de campo.

Por exemplo, suponha que você criou um índice de campos para `IPaddress` em alguns dos grupos de logs da sua conta. Em seguida, você pode criar a consulta a seguir e optar por consultar todos os grupos de logs na conta para encontrar eventos de logs que incluam o valor `198.51.100.0` no campo `IPaddress`.

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

O comando `filterIndex` faz com que essa consulta tente ignorar todos os grupos de logs que não estão indexados para `IPaddress`. Além disso, nos grupos de logs indexados, a consulta ignora eventos de logs que têm um campo `IPaddress`, mas não observaram `198.51.100.0` como o valor desse campo.

Use o operador `IN` para expandir os resultados para qualquer um dos vários valores dos campos indexados. O exemplo a seguir encontra eventos de logs que incluem o valor `198.51.100.0` ou `198.51.100.1` no campo `IPaddress`.

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

CloudWatch Os registros fornecem índices de campo padrão para todos os grupos de registros na classe de registros padrão. Os índices de campo padrão ficam disponíveis automaticamente para os seguintes campos:

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

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

CloudWatch O Logs também fornece índices de campo padrão para determinadas combinações de nome e tipo de fonte de dados. Os índices de campo padrão estão disponíveis automaticamente para as seguintes combinações de nome e tipo de fonte de dados:

Nome e tipo da fonte de dados	Índices de campo padrão

`amazon_vpc.flow`	`action`
	`logStatus`
	`region`
	`flowDirection`
	`type`

`amazon_route53.resolver_query`	`query_type`
	`transport`
	`rcode`

`aws_waf.access`	`action` `httpRequest.country`
`aws_cloudtrail.data` `aws_cloudtrail.management`	`eventSource` `eventName`

Nome e tipo da fonte de dados	Índices de campo padrão

	`awsRegion`
	`userAgent`
	`errorCode`
	`eventType`
	`managementEvent`
	`readOnly`
	`eventCategory`
	`requestId`

Os índices de campo padrão são adicionais aos índices de campo personalizados definidos em sua política. Os índices de campo padrão não são contabilizados em sua cota de índice de campo.

filterIndex comparado ao filtro

Para ilustrar a diferença entre `filterIndex` e `filter`, considere os exemplos de consultas a seguir. Suponha que você criou um índice de campo para `IPaddress` e para quatro de seus grupos de logs, mas não para um quinto grupo. A consulta a seguir usando `filterIndex` ignorará a verificação do grupo de logs que não tem o campo indexado. Para cada grupo de logs indexados, ele tenta verificar somente os eventos de logs que tenham o campo indexado e também retorna somente os resultados após a criação do índice do campo.

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

Por outro lado, se você usar em `filter` vez de `filterIndex` para uma consulta dos mesmos cinco grupos de logs, a consulta tentará verificar não apenas os eventos de logs que contêm o valor nos grupos de logs indexados, mas também examinará o quinto grupo que não está indexado e examinará todos os eventos de logs desse quinto grupo de logs.

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

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

SOURCE

Incluir SOURCE em uma consulta é uma forma útil de especificar as fontes de and/or dados dos grupos de registros a serem incluídas em uma consulta quando você estiver usando a API AWS CLI ou para criar uma consulta. O SOURCE comando é suportado somente na API AWS CLI and, não no CloudWatch console. Ao usar o CloudWatch console para iniciar uma consulta, você usa a interface do console para especificar os grupos de registros.

Grupos de registros de consulta

Para usar SOURCE para especificar os grupos de logs a consultar, é possível usar as seguintes palavras-chave:

- `namePrefix` executa a consulta em grupos de logs que têm nomes que começam com a string que você especifica. Se você a omitir, todos os grupos de logs serão consultados.

  Você pode incluir até 5 prefixos na lista.
- `accountIdentifier`executa a consulta em grupos de registros na AWS conta especificada. Isso funciona somente quando você executa a consulta em uma conta de monitoramento. Se você a omitir, o padrão é consultar todas as contas de origem vinculadas e a conta de monitoramento atual. Para obter mais informações sobre observabilidade entre contas, consulte observabilidade CloudWatch entre contas.

  É possível incluir até 20 identificadores de conta na lista.
- `logGroupClass` executa a consulta em grupos de logs que estão na classe de log especificada, seja Padrão ou Acesso Infrequente. Se você a omitir, será usada a classe de log Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

Como você pode especificar um grande número de grupos de registros para consulta dessa forma, recomendamos que você use SOURCE somente em consultas que usem índices de campo que você criou. Para obter mais informações sobre campos de indexação em grupos de logs, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento

O exemplo a seguir seleciona todos os grupos de logs da conta. Se for uma conta de monitoramento, os grupos de logs no monitoramento e em todas as contas de origem serão selecionados. Se o

número total de grupos de logs exceder 10.000, você verá um erro solicitando reduzir o número de grupos usando um método diferente de seleção de grupos de logs.

```
SOURCE logGroups()
```

O exemplo a seguir seleciona os grupos de logs na conta 111122223333 de origem. Se você iniciar uma consulta em uma conta de monitoramento na observabilidade CloudWatch entre contas, os grupos de registros em todas as contas de origem e na conta de monitoramento serão selecionados por padrão.

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

O próximo exemplo seleciona grupos de logs com base em prefixos de nome.

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

O exemplo a seguir seleciona todos os grupos de logs da classe de logs de Acesso infrequente. Se você não incluir o identificador `class`, a consulta só se aplicará aos grupos de logs da classe de logs Padrão.

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

O próximo exemplo seleciona grupos de logs na conta 111122223333 que começam com prefixos de nome específicos e estão na classe de registro Padrão. A classe não é mencionada no comando porque Padrão é o valor padrão da classe de log.

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

O exemplo final mostra como usar o SOURCE comando com o `start-query` AWS CLI comando.

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

Consultar fontes de dados

SOURCEPara especificar as fontes de dados a serem consultadas, você pode usar a `dataSource` palavra-chave. Você pode incluir até dez fontes de dados na lista.

O exemplo a seguir seleciona a fonte `amazon_vpc.flow` de dados.

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

O exemplo a seguir seleciona a fonte de `amazon_vpc.flow` dados e limita os grupos de registros com base em um prefixo de nome de grupo de registros.

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

Consultar grupos de registros por tags

Para usar SOURCE para filtrar grupos de registros por suas tags, use a `logGroupTags` função. Especifique as tags como uma lista de filtros de tags, cada um com uma `key` `values` matriz opcional.

- Vários filtros de tags com teclas diferentes são combinados com a lógica AND.
- Vários valores dentro do mesmo filtro de tag são combinados com a lógica OR.
- Use * para correspondência de curingas. Por exemplo, `payment*` corresponde a valores que começam com`payment`.
- Use ! como prefixo para negação. Por exemplo, `!production` corresponde a valores que não são`production`.
- Você pode incluir até 5 filtros de tag, cada um com até 5 valores.

O exemplo a seguir seleciona todos os grupos de registros marcados com`team=team1 OR team=team2`.

```
SOURCE logGroupTags([{"key":"team", "values":["team1", "team2"]}])
| fields @message, @timestamp
```

O próximo exemplo seleciona grupos de registros com `payment` os quais a `service` tag começa e a `environment` tag não`production`.

```
SOURCE logGroupTags([{"key":"service", "values":["payment*"]}, {"key":"environment",
 "values":["!production"]}])
```

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

O exemplo a seguir combina a filtragem de tags com um filtro de prefixo de nome.

```
SOURCE logGroups(namePrefix: ['/aws/lambda']) logGroupTags([{"key":"environment",
 "values":["production"]}])
| fields @message, @timestamp
```

pattern

Use `pattern` para agrupar automaticamente seus dados de log em padrões.

Um padrão é uma estrutura de texto compartilhada que se repete nos campos do log. Você pode usar `pattern` para revelar tendências emergentes, monitorar erros conhecidos e identificar linhas de registro que ocorrem com frequência ou são de alto custo. CloudWatch O Logs Insights também fornece uma experiência de console que você pode usar para encontrar e analisar melhor os padrões em seus eventos de log. Para obter mais informações, consulte Análise de padrões.

Como o comando `pattern` identifica padrões comuns automaticamente, você pode usá-lo como ponto de partida para pesquisar e analisar os logs. Você também pode combinar `pattern` com os comandos `filter`, `parse`, ou `sort` para identificar padrões em consultas mais ajustadas.

Entrada do comando pattern

O comando `pattern` espera uma das seguintes entradas: o campo `@message`, um campo extraído criado usando o comando `parse` ou uma string manipulada usando uma ou mais Funções string.

Se o CloudWatch Logs não conseguir inferir o tipo de dados que um token dinâmico representa, exibi-o como <Token- *`number`* > e *`number`* indica onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

Exemplos comuns de tokens dinâmicos incluem códigos de erro, endereços IP, timestamps e IDs de solicitação.

Saída do comando pattern

O comando `pattern` produzirá a seguinte saída:

- `@pattern`: uma estrutura de texto compartilhada que se repete entre seus campos de eventos de log. Os campos que variam dentro de um padrão, como um ID da solicitação ou um timestamp, são representados por tokens. Se o CloudWatch Logs puder determinar o tipo de dados que um

token dinâmico representa, ele exibirá o token como<*string*-*number*>. *string*É uma descrição do tipo de dados que o token representa. *number*Mostra onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

CloudWatch O Logs atribui a parte da string do nome com base na análise do conteúdo dos eventos de log que a contêm.

Se o CloudWatch Logs não conseguir inferir o tipo de dados que um token dinâmico representa, exibi-o como <Token- *number* > e *number* indica onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

Por exemplo, `[INFO] Request time: <Time-1> ms` é uma saída potencial para a mensagem de log `[INFO] Request time: 327 ms`.

- `@ratio`: a proporção de eventos de log de um período selecionado e grupos de registros especificados que correspondem a um padrão identificado. Por exemplo, se metade dos eventos de log nos grupos de logs e no período de tempo selecionados corresponderem ao padrão, `@ratio` retornará `0.50`
- `@sampleCount`: o número de eventos de log de um período selecionado e grupos de logs especificados que correspondem a um padrão identificado.
- `@severityLabel`: a severidade ou o nível do log que indica o tipo de informação contida em um log. Por exemplo, `Error`, `Warning`, `Info` ou `Debug`.

Exemplos

O comando a seguir identifica logs com estruturas semelhantes em grupos de logs especificados no intervalo de tempo selecionado, agrupando-os por padrão e contagem

```
pattern @message
```

O comando `pattern` pode ser usado em combinação com o comando `filter`

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

O comando `pattern` pode ser usado com os comandos `parse` e `sort`

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

```
| pattern cause
| sort @sampleCount asc
```

diferença

Ele compara os eventos de log encontrados no período solicitado com os eventos de log de um período anterior de igual duração. Assim, você pode procurar tendências e descobrir se determinados eventos de log são novos.

Adicione um modificador ao comando `diff` para especificar o período com o qual deseja comparar:

- `diff` compara os eventos de log no intervalo de tempo atualmente selecionado com os eventos de log do intervalo de tempo imediatamente anterior.
- `diff previousDay` compara os eventos de log no intervalo de tempo atualmente selecionado com os eventos de log do mesmo intervalo de tempo no dia anterior.
- `diff previousWeek` compara os eventos de log no intervalo de tempo atualmente selecionado com os eventos de log do mesmo intervalo de tempo na semana anterior.
- `diff previousMonth` compara os eventos de log no intervalo de tempo atualmente selecionado com os eventos de log do mesmo intervalo de tempo no mês anterior.

Para obter mais informações, consulte Compare (diff) com intervalos de tempo anteriores.

parse

Use `parse` para extrair dados de um campo de registro e criar campos extraídos que você possa processar em sua consulta. O `parse` comando oferece suporte a quatro modos: expressões globais, expressões regulares, logfmt e CSV.

Se `fieldName` for omitido, `@message` será usado por padrão. Você pode analisar a partir de qualquer campo nomeado especificando o nome do campo como o primeiro argumento.

Se um evento de log não corresponder ao padrão especificado, você ainda o verá nos resultados, mas sem os campos extraídos.

Modo global

Use curingas (*) como espaços reservados para valores que você deseja extrair e atribua-os aos campos nomeados com. `as`

Sintaxe

```
parse fieldName "pattern" as alias1, alias2
```

O número de * curingas deve ser igual ao número de aliases.

Exemplos

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method, @user
```

```
parse @logStream "*/*/*/*" as env, service, instance, shard
| stats count(*) by env, service
```

Análise encadeada

Extraia um campo e, em seguida, analise ainda mais o campo extraído.

```
parse @message "url=*" as url
| parse url "/api/*/users/*" as apiVersion, userId
| display apiVersion, userId
```

Modo Regex

Use uma expressão regular com grupos de captura nomeados para extrair campos. Para obter informações sobre a sintaxe das expressões regulares, consulte Sintaxe de expressões regulares (regex) compatíveis.

Sintaxe

```
parse fieldName /regex/
```

Use grupos de captura nomeados (`?<name>...`) para definir os campos extraídos.

Exemplos

Use grupos de captura nomeados para extrair campos

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

Use um grupo de captura nomeado para extrair o ENI de um log de fluxo de VPC

```
parse @message /(?<NetworkInterface>eni-.*?) /
| display NetworkInterface, @message
```

Multi-match modo

Use o modo de várias correspondências para extrair todas as correspondências de uma expressão regular de um campo, produzindo várias linhas por evento de log. Adicione a palavra-chave `multi` após o padrão regex.

Sintaxe

```
parse fieldName /regex/ multi
```

Exemplos

Extraia todos os endereços IP de uma linha de registro (correspondência múltipla)

```
parse @message /(\d+\.\d+\.\d+\.\d+)/ as ip_addr multi
| stats count(*) by ip_addr
```

Modo Logfmt

Use `parse logfmt` para analisar linhas de log formatadas em logfmd em pares de valores-chave. Logfmt é um formato de registro estruturado em que cada linha contém pares separados por espaço`key=value`.

Sintaxe

```
parse fieldName logfmt as alias
```

O resultado é um mapa que você acessa com notação de pontos (por exemplo,`lf.level`,`lf.msg`).

Exemplos

```
parse @message logfmt as lf
| filter lf.level = "error"
| display lf.msg, lf.duration
```

```
parse @message logfmt as lf
| stats count(*) by lf.host
```

Modo CSV

Use `parse csv` para analisar linhas de CSV-formatted registro em campos estruturados. Cada valor separado por vírgula é atribuído ao alias correspondente.

Sintaxe

```
parse fieldName csv as alias1, alias2, alias3
```

Exemplos

```
parse @message csv as timestamp, level, message
| filter level = "ERROR"
| display timestamp, message
```

```
parse @message csv as host, method, path, status, duration
| stats avg(duration) by method
```

Extração de campo JSON

Use `json field=`*`fieldName`* para extração explícita de JSON em cadeia de um campo de objeto analisado anteriormente. Isso permite que você extraia chaves aninhadas de um campo estruturado sem analisar novamente a mensagem bruta.

Sintaxe

```
json field=fieldName "key.subkey" as alias
```

Exemplos

```
parse @message /(?<payload>\{.*\})/ as payload
| json field=payload "user.name" as username
| display username
```

```
json field=requestContext "identity.sourceIp" as caller_ip
```

```
| stats count(*) by caller_ip
```

campos relevantes

Use `relevantfields` para identificar automaticamente quais campos em seus dados de registro são mais relevantes para uma determinada condição. O comando compara os registros da linha de base com o subconjunto correspondente à sua condição e retorna os campos classificados por sua pontuação de relevância.

Sintaxe

```
relevantfields [field1, field2, ...] where condition
```

A lista de campos é opcional. Se omitidos, todos os campos são analisados.

O comando retorna os seguintes campos de saída:

- `@fieldName`— Nome do campo
- `@relevanceScore`— Uma pontuação indicando relevância em uma escala de 0 a 1
- `@topRelevanceContributors`— Para campos categóricos, mostra as entradas com as maiores mudanças na frequência.
- `@conditionalMedian`— Para campos numéricos, a mediana dos valores nos registros que atendem à condição
- `@baselineMedian`— Para campos numéricos, a mediana dos valores nos registros que não atendem à condição

Exemplos

Analise todos os campos para obter respostas lentas da API

```
relevantfields where Time > 400
```

Focalize a análise em campos específicos

```
relevantfields Controller, Path, Service where Time > 400
```

Identifique os fatores de tempo limite do Lambda

```
relevantfields where Duration > 5000
```

expandir

Use `expand` para pegar um campo contendo uma matriz JSON e criar um evento de log separado para cada elemento na matriz. Todos os outros campos do evento de registro original são duplicados em cada novo evento.

Sintaxe

```
expand fieldName
```

Exemplo

Se um evento de log contiver `items = ["apple","banana","cherry"]` e`host = "web-01"`, em seguida, `expand items` produzirá três eventos de log: `{items: "apple", host: "web-01"}``{items: "banana", host: "web-01"}`, `{items: "cherry", host: "web-01"}` e.

```
expand items
| stats count(*) by items, host
```

sort

Use `sort` para exibir eventos de log em ordem crescente (`asc`) ou decrescente (`desc`) por um campo especificado. Use-o com o comando `limit` para criar consultas de “N principais” ou “N inferiores”.

O algoritmo de classificação é uma versão atualizada da classificação natural. Se você classificar em ordem crescente, a lógica a seguir será usada.

- Todos os valores não numéricos vêm antes de todos os valores numéricos. Valores numéricos são valores que só incluem números, não uma mistura de números e outros caracteres.
- Para valores não numéricos, o algoritmo agrupa caracteres numéricos consecutivos e caracteres alfabéticos consecutivos em blocos separados para comparação. Ele ordena as partes não numéricas pelos valores Unicode e as partes numéricas primeiro pelo tamanho e depois pelo valor numérico.

Para obter mais informações sobre a ordem Unicode, consulte Lista de caracteres Unicode.

Por exemplo, o resultado a seguir é classificado na ordem crescente.

```
!: >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> sorted by unicode order
#
*%04
0# >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Alphanumeric starting with numbers
5A
111A   >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>  Starts with more digits than 5A, so it sorted
 to be later than 5A
2345_
@ >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> 2345 is compared with @ in the unicode order,
@_
A >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Values starting with letters
A9876fghj
a12345hfh
0 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> Number values
01
1
2
3
```

Se você classificar na ordem decrescente, os resultados da classificação serão invertidos.

Por exemplo, a consulta a seguir de logs de fluxo da Amazon VPC localiza as 15 principais transferências de pacotes entre hosts.

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

stats

Use `stats` para criar visualizações dos seus dados de log, como gráficos de barras, gráficos de linhas e gráficos de áreas empilhadas. Isso ajuda você a identificar padrões em seus dados de registro com mais eficiência. CloudWatch O Logs Insights gera visualizações para consultas que usam a `stats` função e uma ou mais funções de agregação.

Por exemplo, a consulta a seguir em um grupo de logs do Route 53 retorna visualizações que mostram a distribuição dos registros do Route 53 por hora, por tipo de consulta.

```
stats count(*) by queryType, bin(1h)
```

A consulta a seguir coleta os valores de status distintos para cada serviço.

```
stats values(status) as statuses by service
```

Todas essas consultas podem produzir gráficos de barras. Se sua consulta usar a função `bin()` para agrupar os dados por um campo ao longo do tempo, você também poderá ver gráficos de linhas e gráficos de áreas empilhadas.

As unidades e abreviações de tempo a seguir são compatíveis com a função `bin`. Para todas as unidades e abreviações que incluem mais de um caractere, é permitido adicionar "s" para formar o plural. Assim, ambos `hr` e `hrs` funcionam para especificar horas.

- `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`

Tópicos

- Visualizar dados de séries temporais
- Visualizar dados de log agrupados por campos
- Use vários comandos de estatísticas em uma única consulta
- Funções para uso com estatísticas

Visualizar dados de séries temporais

As visualizações de séries temporais funcionam para consultas com as seguintes características:

- A consulta contém uma ou mais funções de agregação. Para obter mais informações, consulte Aggregation Functions in the Stats Command.

- A consulta usa a função `bin()` para agrupar os dados por um campo.

Essas consultas podem produzir gráficos de linha, gráficos de áreas empilhadas, gráficos de barras e gráficos de pizza.

Exemplos

Para obter um tutorial completo, consulte the section called “Tutorial: Executar uma consulta que produz uma visualização de séries temporais”.

Aqui estão mais exemplos de consultas que funcionam para visualização de séries temporais.

A consulta a seguir gera uma visualização dos valores médios do campo `myfield1`, com um ponto de dados criado a cada cinco minutos. Cada ponto de dados é a agregação das médias dos valores `myfield1` dos logs dos últimos cinco minutos.

```
stats avg(myfield1) by bin(5m)
```

A consulta a seguir gera uma visualização dos três valores com base em campos diferentes, com um ponto de dados criado a cada cinco minutos. A visualização é gerada porque a consulta contém funções de agregação e usa `bin()` como o campo de agrupamento.

```
stats avg(myfield1), min(myfield2), max(myfield3) by bin(5m)
```

Restrições do gráfico de linhas e do gráfico de áreas empilhadas

Consultas que agregam informações de entrada de log, mas que não usam a função `bin()`, podem gerar gráficos de barras. No entanto, as consultas não podem gerar gráficos de linha ou gráficos de áreas empilhadas. Para obter mais informações sobre esses tipos de políticas, consulte the section called “Visualizar dados de log agrupados por campos”.

Visualizar dados de log agrupados por campos

É possível produzir gráficos de barras para consultas que usam a função `stats` e uma ou mais funções de agregação. Para obter mais informações, consulte Aggregation Functions in the Stats Command.

Para ver a visualização, execute sua consulta. Depois, escolha a guia Visualization(Visualização), selecione a seta ao lado de Line(Linha) e escolha Bar(Barra). As visualizações estão limitadas a até 100 barras no gráfico de barras.

Exemplos

Para obter um tutorial completo, consulte the section called “Tutorial: Executar uma consulta que produz uma visualização agrupada por campos de log”. Os parágrafos a seguir incluem mais consultas de exemplo para a visualização por campos.

A consulta de log de fluxo da VPC a seguir localiza o número médio de bytes transferidos por sessão para cada endereço de destino.

```
stats avg(bytes) by dstAddr
```

Também é possível produzir um gráfico que inclua mais de uma barra para cada valor resultante. Por exemplo, a consulta de log de fluxo da VPC a seguir localiza o número médio e máximo de bytes transferidos por sessão para cada endereço de destino.

```
stats avg(bytes), max(bytes) by dstAddr
```

A consulta a seguir localiza o número de logs de consulta do Amazon Route 53 para cada tipo de consulta.

```
stats count(*) by queryType
```

Use vários comandos de estatísticas em uma única consulta

Você pode usar vários `stats` comandos em uma única consulta. Isso permite que você realize agregações adicionais na saída da primeira agregação. O número máximo de `stats` comandos permitidos em uma consulta depende da classe de log do grupo de logs.

Exemplo: consulta com dois comandos **`stats`**

Por exemplo, a consulta a seguir encontra primeiro o volume total de tráfego em compartimentos de 5 minutos e, em seguida, calcula o volume de tráfego mais alto, mais baixo e médio entre esses compartimentos de 5 minutos.

```
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
```

Exemplo: combine vários comandos de estatísticas com outras funções, como **`filter`**, **`fields`** e **`bin`**

Você pode combinar vários `stats` comandos com outros comandos, como `filter` e `fields` em uma única consulta. Por exemplo, a consulta a seguir localiza o número de endereços IP distintos nas sessões e localiza o número de sessões por plataforma do cliente, filtra esses endereços IP e, por fim, localiza a média de solicitações de sessão por plataforma do cliente.

```
STATS countDistinct(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
```

Você pode usar as funções `bin` e `dateceil` em consultas com vários comandos `stats`. Por exemplo, a consulta a seguir primeiro combina mensagens em blocos de cinco minutos, depois agrega esses blocos de cinco minutos em blocos de dez minutos e calcula os volumes de tráfego mais altos, mais baixos e médios dentro de cada bloco de dez minutos.

```
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)
```

Observações e limitações

Uma consulta pode ter no máximo 10 `stats` comandos. Não é possível alterar esta cota.

Se você usar um `limit` comando `sort` or, ele deverá aparecer após o último `stats` comando. Se for anterior ao último `stats` comando, a consulta não é válida.

Quando uma consulta tem vários `stats` comandos, os resultados parciais da consulta não começam a ser exibidos até que a primeira `stats` agregação seja concluída.

Nos `stats` comandos subsequentes em uma única consulta, você pode se referir somente aos campos definidos no `stats` comando anterior. Por exemplo, a consulta a seguir não é válida porque o campo `@message` não estará disponível após a primeira agregação `stats`.

```
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
```

Todos os campos que você referenciar após um `stats` comando devem ser definidos nesse `stats` comando.

```
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

A função `bin` sempre usa implicitamente o campo `@timestamp`. Isso significa que você não pode usar `bin` em um `stats` comando subsequente sem usar o `stats` comando anterior para propagar o `timestamp` campo. Por exemplo, a consulta a seguir não é válida.

```
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
```

Em vez disso, defina o `@timestamp` campo no `stats` comando anterior e, em seguida, você poderá usá-lo `dateceil` em um `stats` comando subsequente, como no exemplo a seguir.

```
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)
```

Funções para uso com estatísticas

CloudWatch O Logs Insights suporta funções de agregação de estatísticas e funções de não agregação de estatísticas.

Use funções de agregação de estatísticas no comando `stats` e como argumentos de outras funções.

Função	Tipo de resultado	Description
`avg(fieldName: NumericLogField)`	número	A média dos valores no campo especificado.

`count()`
 	
`count(fieldName: LogField)`
	número	Faz a contagem dos eventos de log. `count()` (ou `count(*)`) conta todos os eventos retornados pela consulta, e o `count(fieldName)` conta todos os registros que incluam o nome do campo especificado.

`countDistinct(fieldName: LogField)`	número	Retorna o número de valores exclusivos do campo. Se o campo tiver cardinalidade muito alta (muitos valores exclusivos), o valor retornado por `countDistinct` será apenas uma aproximação.
`max(fieldName: LogField)`	LogFieldValue	O máximo dos valores desse campo de log nos logs consultados.
`min(fieldName: LogField)`	LogFieldValue	O mínimo dos valores desse campo de log nos logs consultados.
`pct(fieldName: LogFieldValue, percent: number)`	LogFieldValue	Um percentil indica a posição relativa de um valor no conjunto de dados. Por exemplo, `pct(@duration, 95)` retorna o valor `@duration` em que 95% dos valores de `@duration` são menores que esse valor e 5% são maiores que esse valor.
`stddev(fieldName: NumericLogField)`	número	O desvio padrão dos valores no campo especificado.
`sum(fieldName: NumericLogField)`	número	A soma dos valores no campo especificado.

Função	Tipo de resultado	Description
`values(fieldName: LogField)`	array	Coleta os valores distintos do campo especific ado para cada grupo. Também conhecido como`COLLECT_VALUES` .
`variance(fieldName: NumericLogField)`	número	Variância populacional dos valores no campo especificado. Exemplo: `stats variance(responseT ime) as rt_var by service`
`topk(k: number, fieldName: LogField)`	array	Retorna os k valores mais frequentes do campo especificado. Os valores válidos para k variam de 1 a 10000. Essa função não pode ser combinada com by ou outras funções de agregação. Exemplo: `stats topk(3, status)`

Funções de estatísticas de não agregação

Use funções de não agregação no comando `stats` e como argumentos de outras funções.

Função	Tipo de resultado	Description
`earliest(fieldName: LogField)`	LogField	Retorna o valor de `fieldName` do evento do log que tem o primeiro time stamp nos logs consultados.
`latest(fieldName: LogField)`	LogField	Retorna o valor de `fieldName` do evento do log que tem o último time stamp nos logs consultados.
`sortsFirst(fieldNa me: LogField)`	LogField	Retorna o valor do `fieldName` que aparece em primeiro lugar nos logs consultados.

Função	Tipo de resultado	Description
`sortsLast(fieldName: LogField)`	LogField	Retorna o valor do `fieldName` que aparece em último lugar nos logs consultados.

limite

Use `limit` para especificar o número de eventos de log que você deseja que sua consulta retorne. Se você omitir`limit`, o padrão da consulta é retornar até 10.000 eventos de log. Você pode especificar um `limit` valor de até 100.000.

Por exemplo, o exemplo a seguir retorna apenas os 25 eventos de log mais recentes

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

limitar qualquer

Use `limit any` *N* para interromper a consulta assim que os eventos de log *N* correspondentes forem encontrados, sem verificar os dados restantes. Isso pode reduzir significativamente a quantidade de dados digitalizados e acelerar as consultas quando você precisa apenas de uma amostra de eventos correspondentes, em vez de um conjunto ordenado específico.

Com uma consulta normal`limit`, a consulta verifica todos os dados no intervalo de tempo e retorna os principais *N* resultados. Com`limit any`, a consulta é concluída mais cedo quando resultados suficientes são encontrados, o que significa que os eventos retornados podem vir de qualquer parte do intervalo de tempo.

Por exemplo, a consulta a seguir retorna 1 evento de log e interrompe a varredura assim que encontra uma correspondência:

```
fields @message | limit any 1
```

> **Note**
>
> Como `limit any` interrompe a digitalização mais cedo, não é garantido que os resultados estejam em uma ordem específica. Se você precisar de resultados ordenados, use `limit` sem `any` junto com`sort`.

dedup

Use `dedup` para remover resultados duplicados com base em valores específicos em campos que você especifica. É possível usar `dedup` com um ou mais campos. Se você especificar um campo com `dedup`, apenas um evento de log será retornado para cada valor exclusivo desse campo. Se especificar vários campos, um evento de log será retornado para cada combinação exclusiva de valores desses campos.

As duplicatas serão descartadas em ordem de classificação, e apenas o primeiro resultado dessa ordem será mantido. Convém classificar os resultados antes de os submeter ao comando `dedup`. Se os resultados não forem classificados antes de serem executados por `dedup`, a ordem de classificação decrescente padrão `@timestamp` será usada.

Valores nulos não são considerados duplicatas para avaliação. Eventos de log com valores nulos para qualquer um dos campos especificados são mantidos. Para eliminar campos com valores nulos, use **`filter`** com a função `isPresent(field)`.

O único comando de consulta que você pode usar em uma consulta após o comando `dedup` é `limit`.

Quando você usa `dedup` em uma consulta, o console exibe uma mensagem como Mostrar registros X de Y, em que X é o número de resultados desduplicados e Y é o número total de registros correspondidos antes da desduplicação. Isso indica que os registros duplicados foram removidos e não significa que faltam dados.

Exemplo: visualizar somente o evento de log mais recente de cada valor exclusivo do campo chamado **`server`**

O exemplo a seguir mostra os campos `timestamp`, `server`, `severity` e `message` somente para o evento mais recente de cada valor exclusivo de `server`.

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server
```

Para ver mais exemplos de consultas do CloudWatch Logs Insights, consulteConsultas gerais.

unmask

Use `unmask` para exibir todo o conteúdo de um evento de log que tenha algum conteúdo mascarado devido à política de proteção de dados. Para usar este comando, você deve ter a permissão `logs:Unmask`.

Para mais informações sobre a proteção de dados nos grupos de logs, acesse Ajude a proteger dados de log confidenciais com mascaramento.

unnest

Use `unnest` para nivelar uma lista usada como entrada para gerar vários registros com um único registro para cada elemento na lista. Com base no número de itens que um campo contém, esse comando descarta o registro atual e gera novos registros. Cada registro inclui o `unnested_field`, que representa um item. Todos os outros campos vêm do registro original.

A entrada para `unnest` é `LIST`, que vem da função `jsonParse`. Para obter mais informações, consulte Tipos de estrutura. Quaisquer outros tipos, como `MAP`, `String` e `numbers`, são tratados como uma lista com um item no `unnest`.

Estrutura do comando

O exemplo a seguir descreve o formato desse comando.

```
unnest field into unnested_field
```

Consulta de exemplo

O exemplo a seguir analisa uma string de objeto JSON e expande uma lista de eventos de campo.

```
fields jsonParse(@message) as json_message
| unnest json_message.events into event
| display event.name
```

O evento de logs para essa consulta de exemplo pode ser uma string JSON da seguinte forma:

```
{
   "events": [
        {
            "name": "exception"
```

```
        },
        {
            "name": "user action"
        }
    ]
}
```

Nesse caso, a consulta de amostra produz dois registros no resultado da consulta, um com `event.name` como `exception` e outro com `event.name` como ação do usuário.

Consulta de exemplo

O exemplo a seguir nivela uma lista e depois filtra os itens.

```
fields jsonParse(@message) as js
| unnest js.accounts into account
| filter account.type = "internal"
```

Consulta de exemplo

O exemplo a seguir nivela uma lista para agregação.

```
fields jsonParse(trimmedData) as accounts
| unnest accounts into account
| stats sum(account.droppedSpans) as n by account.accountId
| sort n desc
| limit 10
```

lookup

Use `lookup` para enriquecer os resultados da consulta com dados de referência de uma tabela de pesquisa. Você pode preencher uma tabela de pesquisa com dados CSV que você carrega no Amazon CloudWatch Logs ou com os resultados de uma consulta de log concluída ou cancelada. Quando uma consulta é executada, o `lookup` comando combina um campo em seus eventos de log com um campo na tabela de pesquisa e acrescenta os campos de saída especificados aos resultados.

Use tabelas de pesquisa para cenários de enriquecimento de dados, como mapear IDs de usuário para detalhes do usuário, códigos de produto para informações do produto ou códigos de erro para descrições de erros.

Criação e gerenciamento de tabelas de pesquisa

Antes de usar o `lookup` comando em uma consulta, você deve criar uma tabela de pesquisa. Você pode criar e gerenciar tabelas de pesquisa a partir do CloudWatch console ou usando a API Amazon CloudWatch Logs.

Para criar uma tabela de pesquisa (console)

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Configurações e, em seguida, escolha a guia Registros.
3. Role até Pesquisar tabelas e escolha Gerenciar.
4. Escolha Criar tabela de pesquisa.
5. Insira um nome para a tabela de pesquisa. O nome só pode conter caracteres alfanuméricos e sublinhados.
6. (Opcional) Insira uma descrição.
7. Faça upload de um arquivo CSV. O arquivo deve incluir uma linha de cabeçalho com nomes de colunas, usar UTF-8 codificação e não exceder 10 MB.
8. (Opcional) Especifique uma AWS KMS chave para criptografar os dados da tabela.
9. Escolha Criar.

Depois de criar uma tabela de pesquisa, você pode visualizá-la no editor de consultas do CloudWatch Logs Insights. Escolha a guia Pesquisar tabelas para procurar tabelas disponíveis e seus campos.

Para atualizar uma tabela de pesquisa, selecione a tabela e escolha Ações, Atualizar. Faça upload de um novo arquivo CSV para substituir todo o conteúdo existente. Para excluir uma tabela de pesquisa, escolha Ações, Excluir.

Em vez de carregar dados CSV, você também pode criar ou atualizar uma tabela de pesquisa a partir dos resultados de uma consulta de log concluída ou cancelada que você executa no console do Log Analytics ou com a API Amazon CloudWatch Logs. Especifique o ID da consulta com o `queryId` parâmetro da `UpdateLookupTable` API `CreateLookupTable` ou. Uma consulta cancelada preenche a tabela com os resultados parciais que estavam disponíveis quando a consulta foi interrompida. Você deve especificar o conteúdo CSV ou o ID da consulta, mas não ambos. A atualização de uma tabela é uma operação de substituição completa — todo o conteúdo existente é substituído independentemente da fonte. Para atualizar uma tabela de pesquisa com resultados de

consulta automaticamente em uma agenda, use uma consulta agendada com um destino de tabela de pesquisa. Para obter mais informações, consulte Configurando destinos da tabela de pesquisa para consultas agendadas.

> **Note**
>
> Você pode criar até 100 tabelas de pesquisa por conta por Região da AWS. Os arquivos CSV podem ter até 10 MB. Você também pode gerenciar tabelas de pesquisa usando a API Amazon CloudWatch Logs. Para obter mais informações, consulte CreateLookupTablea Referência da API Amazon CloudWatch Logs.

> **Note**
>
> Se a tabela de pesquisa for criptografada com uma chave KMS, o chamador deverá ter a `kms:Decrypt` permissão na chave (a chave KMS usada para criptografar a tabela de pesquisa) para usar a `StartQuery` API com uma consulta que faça referência a essa tabela de pesquisa. Para obter mais informações, consulte Criptografe tabelas de pesquisa em CloudWatch registros usando AWS Key Management Service.

Sintaxe de consulta para pesquisa

Estrutura do comando

O seguinte mostra o formato desse comando.

```
lookup table match-fields output-mode output-field[,...]
```

O comando usa os seguintes argumentos:

- *`table`*— O nome da tabela de pesquisa a ser usada.
- *`match-fields`*— Especifique um ou mais campos para comparar os eventos de log com a tabela de pesquisa. Você pode usar qualquer um dos seguintes formulários:
  - *`lookup-field`* `as` *`log-field`* `[,...]`— Use `as` quando o nome da coluna da tabela de pesquisa for diferente do nome do campo do evento de registro. Por exemplo, `ip_address as srcAddr` compara a `ip_address` coluna na tabela de pesquisa com o `srcAddr` campo em seus eventos de registro.

  - *lookup-field* `[,...]`— Quando o nome do campo do evento de registro é igual ao nome da coluna da tabela de pesquisa, você pode omitir `as` e especificar o nome do campo diretamente. Por exemplo, `department, role` compara as duas colunas aos campos de eventos de log com os mesmos nomes.

  Quando vários campos de correspondência são especificados, uma linha na tabela de pesquisa deve corresponder a todos os campos para produzir um resultado (lógica AND).

- *output-mode*— Especifica como os campos de saída são adicionados aos resultados. Use uma das seguintes opções:
  - `OUTPUT`— Adiciona os campos de saída aos resultados. Se um campo com o mesmo nome já existir no evento de log, ele será substituído pelo valor da tabela de pesquisa. Se nenhuma correspondência for encontrada, o campo será definido como nulo.
  - `OUTPUTNEW`— Adiciona os campos de saída aos resultados somente se o campo ainda não existir no evento de log. Se o campo já tiver um valor, o valor original será mantido. Se nenhuma correspondência for encontrada, o campo permanecerá inalterado.
- *output-field*— Um ou mais campos da tabela de pesquisa para adicionar aos resultados.

Exemplo: enriqueça eventos de registro com detalhes do usuário

Suponha que você tenha um grupo de registros com eventos que contêm um `id` campo e uma tabela de pesquisa chamada `user_data` com colunas `id` `nameemail`,, `department` e. A consulta a seguir enriquece cada evento de log com o nome do usuário, e-mail e departamento da tabela de pesquisa.

```
fields action, status, name, email, department
| lookup user_data id OUTPUT name, email, department
```

Exemplo: Use a pesquisa com agregação

Você pode usar campos de saída de pesquisa com funções de agregação. A consulta a seguir enriquece os eventos de log com detalhes do usuário e, em seguida, conta os eventos agrupados por endereço de e-mail.

```
fields user_id, action, username, email, department
| lookup user_data user_id OUTPUT username, email, department
| stats count(*) by email
```

Exemplo: Use a pesquisa com filtro

Você pode filtrar os resultados com base nos campos retornados pela pesquisa. A consulta a seguir enriquece os eventos de registro e, em seguida, filtra para mostrar somente os eventos de um departamento específico.

```
fields user_id, action
| lookup user_data user_id OUTPUT username, email, department
| filter department = "Engineering"
```

Exemplo: use OUTPUTNEW para enriquecer sem sobrescrever

Se seus eventos de registro já contiverem um `hostname` campo, mas às vezes ele estiver vazio, use `OUTPUTNEW` para preencher os valores ausentes sem sobrescrever os existentes.

```
fields srcAddr, hostname
| lookup known_hosts ip_address as srcAddr OUTPUTNEW hostname, region
```

Exemplo: use a pesquisa com vários campos de correspondência

Você pode combinar em mais de um campo. A consulta a seguir corresponde à tabela de pesquisa `srcAddr` e `dstPort` à tabela de pesquisa para identificar serviços de rede conhecidos.

```
fields @timestamp, srcAddr, dstAddr, dstPort
| lookup network_services ip_address as srcAddr, port as dstPort OUTPUT service_name,
 owner
| filter ispresent(service_name)
```

Exemplo: Use a pesquisa com nomes de campo correspondentes

Quando os nomes dos campos do evento de log correspondem exatamente aos nomes das colunas da tabela de pesquisa, você pode omitir a `as` palavra-chave. A consulta a seguir combina os `role` campos `department` e os campos diretamente com a tabela de pesquisa.

```
fields @timestamp, department, role
| lookup employees department, role OUTPUT office, manager
```

ingressar

Combina eventos de log de um grupo de log de origem com eventos de outro grupo de log ou resultado de consulta com base em um campo correspondente.

Use o `join` comando para correlacionar eventos de log relacionados em diferentes fontes, como grupos de registros, usando chaves comuns entre eles, como identificadores de solicitação ou IDs de transação correspondentes.

Sintaxe

```
join type=<join_type> left=<left_alias> right=<right_alias>
    where <left_alias>.<field>=<right_alias>.<field>
    (SOURCE <right_log_group>)
```

Parâmetros

- `<right_log_group>`— A fonte de dados secundária à qual se juntar.
- `<left_alias>`e `<right_alias>` — Aliases para distinguir campos das fontes de dados esquerda (primária) e direita (secundária).
- `where <field>`— Especifica o campo usado como chave de junção. O campo deve existir nas duas fontes de dados.
- `type=<join_type>`(opcional) — Especifica o tipo de junção. Os valores válidos são:
  - `inner`(padrão) — Retorna somente registros correspondentes
  - `left`— Retorna todos os registros da fonte de dados primária e os registros correspondentes da fonte de dados secundária

Exemplos

Example Exemplo 1: correlacionar solicitações do API Gateway com registros de execução do Lambda

Este exemplo mostra como unir os registros de acesso do API Gateway aos registros de funções do Lambda para correlacionar as solicitações recebidas com o processamento de back-end. Isso é útil para solucionar problemas de fluxos de solicitações de ponta a ponta e identificar quais invocações do Lambda correspondem a solicitações de API específicas.

```
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
```

Essa consulta:

1. Consulta registros de acesso e filtros do API Gateway em busca de erros do servidor (status >= 500)
2. Une-se aos registros da função Lambda usando `requestId` o campo que aparece nas duas fontes de registro
3. Usa aliases (`apielambda`) para distinguir campos de cada fonte
4. Retorna informações combinadas que mostram a latência da API junto com a duração da execução do Lambda e o uso da memória
5. Classifica os resultados por latência da API para identificar as solicitações mais lentas

Example Exemplo 2: Rastreie transações distribuídas em microsserviços

Ao depurar problemas em uma arquitetura de microsserviços, você geralmente precisa rastrear uma transação em vários serviços. Este exemplo mostra como unir registros de dois serviços diferentes usando um ID de transação comum.

```
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)
```

Essa consulta:

1. Começa com eventos de criação de pedidos do serviço de pedidos
2. Usa a `left join` para incluir todos os pedidos, mesmo aqueles sem registros de pagamento correspondentes
3. Associa-se a eventos de processamento de pagamentos usando o campo compartilhado `transactionId`

4. Filtra os resultados finais para mostrar somente pedidos com pagamentos falhados ou registros de pagamento ausentes

A junção à esquerda é importante aqui porque garante que você veja pedidos que foram criados, mas nunca tiveram um evento de pagamento correspondente, o que pode indicar uma falha no sistema.

Comportamento

- A fonte de dados primária (lado esquerdo) é processada primeiro.
- A fonte de dados secundária é avaliada e combinada usando a chave de junção especificada.
- A correspondência é realizada usando a comparação de igualdade no campo de junção.
- Para junções à esquerda, registros incomparáveis da fonte de dados primária são mantidos com valores nulos para campos secundários.

Observações e limitações

- Somente condições de igualdade (=) são suportadas.
- Somente um comando de junção é suportado por consulta.
- As chaves de junção devem existir em ambas as fontes de dados e ser de tipos compatíveis.
- As consultas usando join podem digitalizar mais dados e gerar custos mais altos.
- O número de valores-chave exclusivos na fonte de dados secundária é limitado a 50.000 para garantir o desempenho da consulta.
- Não há suporte para subconsultas no lado direito da junção.

Comandos relacionados

- campos
- filtro
- analisar
- stats
- classificar

- limit

subconsultas

Uma subconsulta é uma consulta aninhada do Logs Insights que pode ser usada como entrada para outra consulta. As subconsultas podem ser usadas para derivar conjuntos de resultados intermediários que são então consumidos pelos comandos subsequentes.

Sintaxe

Subconsulta no filtro

```
filter <field> in (
    <subquery>
)
```

Parâmetros

- `<subquery>`— Uma consulta válida do Logs Insights que retorna um conjunto de resultados. A subconsulta deve produzir campos referenciados pela consulta externa.

Exemplos

Example Exemplo 1: encontre solicitações que encontraram erros em serviços downstream

Este exemplo mostra como usar uma subconsulta para identificar solicitações em seu serviço principal que resultaram em erros em um serviço downstream. Isso é útil para solucionar falhas em cascata em sistemas distribuídos.

```
filter requestId in (
    SOURCE '/aws/lambda/database-service'
    | filter errorType = "DatabaseConnectionTimeout"
    | fields requestId
)
| fields @timestamp, requestId, endpoint, userId, responseTime
| sort @timestamp desc
```

Essa consulta:

1. A subconsulta encontra todos os `requestId` valores do serviço de banco de dados que tiveram tempos limite de conexão.
2. A consulta externa filtra os registros do seu serviço principal para mostrar somente solicitações que correspondam a esses IDs de solicitação propensos a erros.
3. Os resultados mostram o contexto completo das solicitações que falharam posteriormente, incluindo quais endpoints e usuários foram afetados

Esse padrão ajuda você a entender o impacto inicial das falhas posteriores.

Example Exemplo 2: Identifique solicitações frequentes de investigação direcionada

Este exemplo demonstra o uso de uma subconsulta com agregação para encontrar solicitações que falham repetidamente, o que geralmente indica problemas sistemáticos em vez de erros transitórios.

```
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
```

Essa consulta:

1. A subconsulta agrega tentativas de pagamento malsucedidas e identifica os IDs de solicitação que falharam mais de três vezes
2. A consulta externa recupera todos os eventos de registro desses IDs de solicitação problemáticos.
3. Os resultados são classificados cronologicamente para mostrar a progressão das tentativas de repetição

Isso ajuda a distinguir entre falhas transitórias (ocorrência única) e problemas persistentes (várias falhas) que exigem uma investigação mais profunda.

Comportamento

- As subconsultas são executadas independentemente da consulta externa.

- Os resultados são materializados antes de serem consumidos pela consulta externa.
- Somente os campos selecionados explicitamente na subconsulta estão disponíveis para a consulta externa.

Observações e limitações

- As subconsultas devem retornar campos referenciados pela consulta externa.
- Não há suporte para subconsultas aninhadas.
- As subconsultas podem aumentar o tempo e o custo de execução da consulta.
- Não há suporte para subconsultas correlacionadas.
- A execução da consulta interna é limitada a 30 segundos.

Comandos relacionados

- campos
- filtro
- analisar
- stats
- classificar
- limit

regressão automática

Use o `autoregress` comando para criar cópias com retardamento (linha anterior) dos valores de um campo. Isso é lag/lead equivalente ao CloudWatch Logs Insights.

Sintaxe

```
| autoregress field [AS alias] [p=start[-end]]
```

O comando usa os seguintes argumentos:

- *`field`*— O campo para o qual criar valores de defasagem.
- `AS` *`alias`*(Opcional) — Um alias para o campo de saída.

- p=*N*(Opcional) — Profundidade de defasagem única (padrão 1). Use p=*N*-*M* para uma variedade de defasagens.

Exemplo

A consulta a seguir cria quatro campos de atraso (`bytes_p1`até`bytes_p4`).

```
fields @timestamp, bytes
| autoregress bytes p=1-4
```

logcompare

Use o `logcompare` comando para comparar a janela de tempo atual com uma janela de linha de base deslocada para trás por uma duração especificada.

Sintaxe

```
| logcompare timeshift duration
```

O comando usa os seguintes argumentos:

- `timeshift` *`duration`*— A duração da mudança para a janela da linha de base (por exemplo,7d).

Exemplo

A consulta a seguir compara os registros do dia anterior com o mesmo período da semana anterior.

```
SOURCE lg start=-1d end=now
| logcompare timeshift 7d
```

preenchimento

Use o `filldown` comando para levar o último valor não nulo para preencher lacunas. Você pode especificar nomes de campo ou usar curingas. Se você não especificar campos, todos os campos serão preenchidos.

Sintaxe

```
| filldown [field1 [field2 ...]]
```

O comando usa os seguintes argumentos:

- *field*(Opcional) — Um ou mais nomes de campo ou padrões curinga. Se omitido, todos os campos serão preenchidos.

Exemplo

A consulta a seguir transmite valores não nulos no `host` campo e em todos os campos correspondentes. `cpu*`

```
fields @timestamp, host, cpu_user, cpu_system
| filldown host cpu*
```

preenchimento ausente

Use o `fillmissing` comando para inserir linhas para compartimentos de tempo vazios depois`stats ... by bin()`. Opcionalmente, você pode preencher campos com um valor constante.

Sintaxe

```
| fillmissing [with value for field [, value for field ...]]
```

O comando usa os seguintes argumentos:

- `with` *value* `for` *field*(Opcional) — O valor constante a ser atribuído ao campo especificado nas linhas inseridas.

Exemplo

A consulta a seguir preenche compartimentos vazios de 1 minuto com 0 para o campo. `avg_latency`

```
fields @timestamp, latency
| stats avg(latency) as avg_latency by bin(1m)
| fillmissing with 0 for avg_latency
```

consulta de cidra

Use o `cidrlookup` comando para enriquecer eventos combinando um campo IP com intervalos CIDR em uma tabela de pesquisa.

Sintaxe

```
| cidrlookup table ipField [AS cidrColumn]
    OUTPUT|OUTPUTNEW lookupField [AS eventField] [, ...]
```

O comando usa os seguintes argumentos:

- *`table`*— O nome da tabela de pesquisa contendo intervalos de CIDR.
- *`ipField`*— O campo do evento contendo o endereço IP correspondente.
- `AS` *`cidrColumn`*(Opcional) — A coluna na tabela de pesquisa que contém os intervalos de CIDR.
- `OUTPUT|OUTPUTNEW`— Use `OUTPUT` para sobrescrever campos existentes ou `OUTPUTNEW` para preencher somente campos vazios.
- *`lookupField`*— Um ou mais campos da tabela de pesquisa para adicionar ao evento.

Exemplo

A consulta a seguir enriquece os eventos com região, proprietário e datacenter a partir de uma tabela de pesquisa de rede.

```
fields @timestamp, srcAddr
| cidrlookup net_table srcAddr OUTPUT region, owner, datacenter
```

outlier

Use o `outlier` comando para detectar discrepâncias estatísticas em campos numéricos com base no intervalo interquartil (IQR). Você pode remover ou fixar linhas discrepantes.

Sintaxe

```
| outlier [action=remove|transform] [param=n] [uselower=true|false] [mark=true|
false] field [field ...]
```

O comando usa os seguintes argumentos:

- `action`(Opcional) — Ou `remove` (elimina linhas discrepantes) ou `transform` (fixa-se até o limite). Padrão: `transform`.
- `param`(Opcional) — O multiplicador IQR. Padrão: `2.5`.
- `uselower`(Opcional) — Se devem ser detectados valores discrepantes de limite inferior. Padrão: `false`.
- `mark`(Opcional) — Se deve adicionar um campo de marcador booleano. Padrão: `false`.
- *`field`*— Um ou mais campos numéricos para verificar se há valores discrepantes.

Exemplo

A consulta a seguir remove as linhas em que a CPU ou a memória são discrepantes, usando um multiplicador IQR 3.0 com detecção e marcação de limite inferior.

```
fields @timestamp, cpu, memory
| outlier action=remove param=3.0 uselower=true mark=true cpu memory
```

accum

Use o `accum` comando para calcular uma soma cumulativa em execução de um campo numérico.

Sintaxe

```
| accum field [AS out]
```

O comando usa os seguintes argumentos:

- *`field`*— O campo numérico a ser acumulado.
- `AS` *`out`*(Opcional) — Um alias para o campo de saída.

Exemplo

A consulta a seguir calcula um total de bytes em execução.

```
fields @timestamp, bytes
| accum bytes AS totalBytes
```

apêndices

Use o `appendcols` comando para acrescentar as colunas de uma subconsulta aos resultados atuais por meio da correspondência de linhas posicionais.

Sintaxe

```
| appendcols [override=true|false] [max=n] ( subquery )
```

O comando usa os seguintes argumentos:

- `override`(Opcional) — Se os campos existentes devem ser substituídos. Padrão: `false`.
- `max`(Opcional) — Máximo de linhas a serem processadas (1—100000, padrão 10000).
- *`subquery`*— Uma consulta completa do CloudWatch Logs Insights entre parênteses.

Exemplo

A consulta a seguir acrescenta a latência média por serviço de outro grupo de registros.

```
fields svc
| stats count(*) as cnt by svc
| appendcols override=true ( SOURCE lg | stats avg(latency) as avgLat by svc )
```

sessionizar

Use o `sessionize` comando para agrupar eventos em sessões por campos de identidade e uma lacuna de inatividade. O comando emite um campo identificador de sessão.

Sintaxe

```
| sessionize field[, field ...] [maxspan duration] [AS out]
```

O comando usa os seguintes argumentos:

- *`field`*— Um ou mais campos de identidade para agrupar.
- `maxspan` *`duration`*(Opcional) — A lacuna máxima de inatividade antes de iniciar uma nova sessão. Padrão: `30m`.
- `AS` *`out`*(Opcional) — O nome do campo de saída. Padrão: `session_id`.

Exemplo

A consulta a seguir agrupa eventos em sessões por usuário e dispositivo com um intervalo de inatividade de 1 hora.

```
fields @timestamp, userId, deviceId
| sessionize userId, deviceId maxspan 1h as my_session
```

Contagem frequente

Use o `countFrequent` comando (também disponível como`count_frequent`) para calcular uma contagem aproximada de cada combinação exclusiva de valor de campo, classificada em ordem decrescente. O comando emite um `_approxcount` campo.

Sintaxe

```
| countFrequent field[, field ...]
```

O comando usa os seguintes argumentos:

- *`field`*— Um ou mais campos para contar as frequências.

Exemplo

A consulta a seguir conta a frequência das combinações de método e status.

```
fields method, status
| countFrequent method, status
```

para onde

Use o `where` comando como um alias para o `filter` comando. Ele aceita sintaxe e comportamento idênticos.

Sintaxe

```
| where condition
```

O comando usa os seguintes argumentos:

- *condition*— Uma expressão booleana idêntica à que o `filter` comando aceita.

Exemplo

A consulta a seguir filtra eventos de log contendo “erro”.

```
fields @timestamp, @message
| where @message like /error/
```

estimativa

Use o `estimate` comando para retornar os bytes estimados que a consulta examinaria nos grupos de registros e no intervalo de tempo selecionados, sem executar a consulta. Usando os bytes estimados retornados por esse comando, você pode refinar a seleção do grupo de registros, o intervalo de tempo e os filtros antes de executar a consulta. O `estimate` comando não incorre em cobranças de consulta do CloudWatch Logs Insights.

O `estimate` comando deve ser o último comando na consulta.

> **Note**
>
> O valor que `estimate` retorna é aproximado e pode diferir do volume real de dados digitalizados quando você executa a consulta.

Sintaxe

```
| estimate
```

Exemplo

A consulta a seguir retorna uma estimativa do volume de dados, em bytes, que a consulta verificaria sem executá-la.

```
fields @timestamp, @message
| filter @message like /error/
| estimate
```

Funções booleanas, de comparação, numéricas e de data e hora, entre outras

CloudWatch O Logs Insights oferece suporte a muitas outras operações e funções em consultas, conforme explicado nas seções a seguir.

Tópicos

- Operadores aritméticos
- Operadores booleanos
- Operadores de comparação
- Operadores numéricos
- Tipos de estrutura
- Funções de data e hora
- Funções gerais
- Funções JSON
- Funções de string de endereço IP
- Funções de string

Operadores aritméticos

Os operadores aritméticos aceitam tipos de dados numéricos como argumentos e retornam resultados numéricos. Use operadores aritméticos nos comandos `filter` e `fields` e como argumentos de outras funções.

Operation	Description
`a + b`	Adição
`a - b`	Subtração
`a * b`	Multiplicação
`a / b`	Divisão
`a ^ b`	Exponenciação (`2 ^ 3` retorna 8)
`a % b`	Resto ou módulo (`10 % 3` retorna 1)

Operadores booleanos

Use os operadores booleanos **`and`**, **`or`** e **`not`**.

> **Note**
>
> Use operadores booleanos somente em funções que retornam um valor de TRUE ou FALSE.

Operadores de comparação

Os operadores de comparação aceitam todos os tipos de dados como argumentos e retornam um resultado booliano. Use operadores de comparação no comando `filter` e como argumentos de outras funções.

Operador	Description
=	Equal
!=	Not equal
<	Menor que
>	Maior que
<=	Menor ou igual a
>=	Maior ou igual a

Operadores numéricos

As operações numéricas aceitam tipos de dados numéricos como argumentos e retornam resultados numéricos. Use operadores numéricos nos comandos `filter` e `fields` e como argumentos de outras funções.

Operation	Tipo de resultado	Description
`abs(a: number)`	número	Valor absoluto

Operation	Tipo de resultado	Description
`ceil(a: number)`	número	Arredonde para o máximo (o menor inteiro maior que o valor de a)
`floor(a: number)`	número	Arredonde para o mínimo (o maior inteiro menor que o valor de a)
`greatest(a: number, ...numbers: number[])`	número	Retorna o maior valor
`least(a: number, ...numbers: number[])`	número	Retorna o menor valor
`log(a: number)`	número	Log natural
`round(a: number [, d: number])`	número	Arredonda o valor de. a Com um argumento, arredonda para o número inteiro mais próximo. Com dois argumento s, arredonda para d casas decimais.
`sqrt(a: number)`	número	Raiz quadrada
`haversine(lat1: number, lon1: number, lat2: number, lon2: number)`	número	Calcula a distância do grande círculo em quilômetros entre dois pontos geográficos especificados por latitude e longitude em graus.
`toNumber(fieldName: string)`	número	Converte um valor de campo de string em um valor numérico.

Operation	Tipo de resultado	Description
`toInt(fieldName: string)`	número	Converte um valor de campo em um inteiro (32 bits).
`toLong(fieldName: string)`	número	Converte um valor de campo em um inteiro longo (64 bits).
`toDouble(fieldName: string)`	número	Converte um valor de campo em um número de ponto flutuante de precisão dupla.

Tipos de estrutura

Um mapa ou lista é um tipo de estrutura no CloudWatch Logs Insights que permite acessar e usar atributos para consultas.

Exemplo: para obter um mapa ou uma lista

Use `jsonParse` para analisar um campo que é uma string json em um mapa ou uma lista.

```
fields jsonParse(@message) as json_message
```

Exemplo: para acessar atributos

Use o operador de acesso por ponto (map.attribute) para acessar itens em um mapa. Se um atributo em um mapa contiver caracteres especiais, use crases para delimitar o nome do atributo (map.attributes). `special.char`).

```
fields jsonParse(@message) as json_message
| stats count() by json_message.status_code
```

Use o operador de acesso por colchetes (list[index]) para recuperar um item em uma posição específica na lista.

```
fields jsonParse(@message) as json_message
| filter json_message.users[1].action = "PutData"
```

Coloque caracteres especiais entre crases (``) quando caracteres especiais estiverem presentes no nome da chave.

```
fields jsonParse(@message) as json_message
| filter json_message.`user.id` = "123"
```

Exemplo: resultados vazios

Mapas e listas são tratados como nulos para funções de string, número e data e hora.

```
fields jsonParse(@message) as json_message
| display toupper(json_message)
```

A comparação do mapa e da lista com quaisquer outros campos resulta em `false`.

> **Note**
>
> O uso de mapa e lista em `dedup`, `pattern`, `sort` e `stats` não é suportado.

Funções de data e hora

Funções de data e hora

Use funções de data e hora nos comandos `fields` e `filter` e como argumentos de outras funções. Use essas funções para criar buckets de tempo para consultas com funções de agregação. Use períodos que consistam em um número e uma das seguintes abreviaturas:

- `ms` para milissegundos
- `s` para segundos
- `m` para minutos
- `h` para horas

Por exemplo, `10m` é 10 minutos, e `1h` é uma hora.

> **Note**
>
> Use a unidade de tempo mais apropriada para a função de data e hora. CloudWatch Os registros limitam sua solicitação de acordo com a unidade de tempo que você escolher.

Por exemplo, o limite máximo será 60 para qualquer solicitação que use `s`. Então, se você especificar`bin(300s)`, o CloudWatch Logs realmente implementa isso como 60 segundos, porque 60 é o número de segundos em um minuto, então o CloudWatch Logs não usará um número maior que 60 com`s`. Para criar um bucket de 5 minutos, use `bin(5m)` em vez disso. O limite para `ms` é 1.000, o limite para `s` e `m` é 60 e o limite para `h` é 24.

A tabela a seguir contém uma lista das diferentes funções de data e hora que você pode usar nos comandos de consulta. A tabela lista o tipo de resultado de cada função e contém uma descrição de cada função.

Tip

Quando você cria um comando de consulta, é possível usar o seletor de intervalo de tempo para selecionar um período que deseja consultar. Por exemplo, você pode definir um período entre intervalos de 5 e 30 minutos; intervalos de 1, 3 e 12 horas; ou um período personalizado. Você também pode definir períodos entre datas específicas.

Função	Tipo de resultado	Description

`bin(period: Period)`	Timestamp	Arredonda o valor de `@timestamp` para o período indicado e trunca. Por exemplo, `bin(5m)` arredonda o valor de `@timestamp` para os 5 minutos mais próximos.
		
		Você pode usar isso para agrupar várias entradas de log em uma consulta. O seguinte exemplo conta a quantidade de exceções por hora:
		
		`filter @message like /Exception/`
		`    | stats count(*) as exceptionCount`
		`by bin(1h)`
		`    | sort exceptionCount desc`

Função	Tipo de resultado	Description
		As unidades e abreviações de tempo a seguir são compatíveis com a função `bin`. Para todas as unidades e abreviações que incluem mais de um caractere, é permitido adicionar "s" para formar o plural. Assim, ambos `hr` e `hrs` funcionam para especificar horas. • `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`
`datefloor(timestamp: Timestamp, period: Period)`	Timestamp	Trunca o time stamp para o período indicado. Por exemplo, `datefloor(@timestamp, 1h)` trunca todos os valores de `@timestamp` no final.
`dateceil(timestamp : Timestamp, period: Period)`	Timestamp	Arredonda o time stamp para o período indicado e trunca. Por exemplo, `dateceil( @timestamp, 1h)` trunca todos os valores de `@timestamp` no início.
`fromMillis(fieldNa me: number)`	Timestamp	Interpreta o campo de entrada como o número de milissegundos desde a epoch do Unix e o converte em um time stamp.

Função	Tipo de resultado	Description
`toMillis(fieldName: Timestamp)`	número	Converte o time stamp encontrado no campo nomeado em um número que representa os milissegundos desde a epoch do Unix. Por exemplo, `toMillis(@timestamp)` converte o carimbo de data/hora `2022-01-14T13:18:031.000-08:00` para `1642195111000` .

`now()`	número	Retorna a hora em que o processamento da consulta foi iniciado, em segundos de época. Essa função não usa argumentos.
		
		Você pode usá-la para filtrar os resultados da consulta de acordo com a hora atual.
		
		Por exemplo, a consulta a seguir gera todos os erros 4xx das últimas duas horas:
		
		`parse @message "Status Code: *;" as statusCode\n`
		`| filter statusCode >= 400 and statusCode <= 499 \n`
		`| filter toMillis(@timestamp) >= (now() * 1000 - 7200000)`
		
		O exemplo a seguir retorna todas as entradas de log das últimas cinco horas que contêm a palavra `error` ou `failure`
		
		`fields @timestamp, @message`
		`| filter @message like /(?i)(error|failure)/`
		`| filter toMillis(@timestamp) >= (now() * 1000 - 18000000)`

Função	Tipo de resultado	Description
`parseDate(fieldName: string, format: string [, timezone: string])`	número	Analisa uma string de data em milissegundos de época usando um padrão Java. DateTimeFormatter O `timezone` argumento opcional especifica o fuso horário a ser usado para análise. Exemplo: `fields parseDate(field, "yyyy-MM-dd", "UTC") as epoch`
`formatDate(timestamp: LogField, format: string [, timezone: string])`	string	Formata um timestamp usando uma string de formato no estilo strftime. Também disponível como alias`strftime`. O `timezone` argumento opcional especifica o fuso horário para formatação. Exemplo: `fields formatDate(@timestamp, "%Y-%m-%d", "UTC") as date`

> **Note**
>
> Atualmente, o CloudWatch Logs Insights não oferece suporte à filtragem de registros com carimbos de data/hora legíveis por humanos.

Funções gerais

Funções gerais

Use funções gerais nos comandos `fields` e `filter` e como argumentos de outras funções.

Função	Tipo de resultado	Description
`ispresent(fieldName: LogField)`	Booleano	Retorna `true` se o campo existir

Função	Tipo de resultado	Description
`coalesce(fieldName: LogField, ...fieldNames: LogField[])`	LogField	Retorna o primeiro valor não nulo da lista
`case(cond1, val1, cond2, val2, ..., [default])`	LogField	Avalia as condições em ordem e retorna o valor da primeira condição verdadeira. Se nenhuma condição for verdadeira e um padrão for fornecido, retornará o padrão. Suporta até 10 filiais.
`if(condition: Boolean, trueValue: LogField, falseValue: LogField)`	LogField	Avalia uma condição e retorna `trueValue` se a condição é verdadeira `falseValue` ou não.
`isNumeric(fieldName: LogField)`	Booleano	Retorna `true` se o valor do campo pode ser analisado como um número. Exemplo: `filter isNumeric(@duration)`
`messageSize(fieldName: LogField)`	número	Retorna o comprimento do byte de um campo de string. Exemplo: `fields messageSize(@message) as size`
`queryStartTime()`	número	Retorna a hora de início da janela de consulta em milissegundos de época.

Função	Tipo de resultado	Description
`queryEndTime()`	número	Retorna a hora de término da janela de consulta em milissegundos de época.
`queryTimeRange()`	número	Retorna a duração da janela de consulta em milissegundos.

Funções JSON

Funções JSON

Use funções JSON nos comandos `fields` e `filter` como argumentos de outras funções.

Função	Tipo de resultado	Description
`jsonParse(fieldName: string)`	Mapa | Lista | Vazio	Retorna um mapa ou uma lista quando a entrada é uma representação de string do objeto JSON ou de uma matriz JSON. Retorna um valor vazio, se a entrada não for uma das representações.
`jsonStringify(fieldName: Map | List)`	String	Retorna uma string JSON de um mapa ou dados da lista.
`jsonArraySize(fieldName: LogField)`	número	Retorna a contagem de elementos de um campo de string de matriz JSON. Exemplo: `fields jsonArraySize(Operation) as arrayLen`

Função	Tipo de resultado	Description
`jsonArrayContains(fieldName: LogField, value: string)`	Booleano	Retorna `true` (1) se a matriz JSON no campo contiver o valor especificado. Retorna `false` (0) para JSON inválido, sem matriz ou matriz vazia. Usa comparação com distinção entre maiúsculas e minúsculas Exemplo: `filter jsonArrayContains( @roles, "admin")`

Funções de string de endereço IP

Funções de string de endereço IP

Use funções de string no endereço IP dos comandos `filter` e `fields` e como argumentos de outras funções.

Função	Tipo de resultado	Description
`isValidIp(fieldName: string)`	booliano	Retorna `true` se o campo for um endereço IPv4 ou IPv6 válido.
`isValidIpV4(fieldName: string)`	booleano	Retorna `true` se o campo for um endereço IPv4 válido.
`isValidIpV6(fieldName: string)`	booleano	Retorna `true` se o campo for um endereço IPv6 válido.
`isIpInSubnet(fieldName: string, subnet: string)`	booleano	Retorna `true` se o campo for um endereço IPv4 ou IPv6 válido dentro da sub-rede v4 ou v6 especificada. Ao especificar a sub-rede,

Função	Tipo de resultado	Description
		use a notação CIDR, como `192.0.2.0/24` ou `2001:db8::/32` , onde `192.0.2.0` ou`2001:db8::` é o início do bloco CIDR.
`isIpv4InSubnet(fieldName: string, subnet: string)`	booleano	Retorna `true` se o campo for um endereço IPv4 válido dentro da sub-rede v4 especificada. Ao especificar a sub-rede, use a notação CIDR, como `192.0.2.0/24` , em que `192.0.2.0` é o início do bloco CIDR.
`isIpv6InSubnet(fieldName: string, subnet: string)`	booleano	Retorna `true` se o campo for um endereço IPv6 válido dentro da sub-rede v6 especificada. Ao especificar a sub-rede, use a notação CIDR, como `2001:db8::/32` , em que `2001:db8::` é o início do bloco CIDR.
`ipv4ToNumber(fieldName: string)`	número	Converte uma string de endereço IPv4 em sua representação numérica.
`isPrivateIP(fieldName: string)`	booleano	Retorna `true` se o endereço IP estiver em um intervalo privado (RFC 1918).
`isPublicIP(fieldName: string)`	booleano	Retorna `true` se o endereço IP for roteável publicamente.
`isReservedIP(fieldName: string)`	booleano	Retorna `true` se o endereço IP estiver em um intervalo reservado.

Funções de string

Funções de string

Use funções de string nos comandos `fields` e `filter` e como argumentos de outras funções.

Função	Tipo de resultado	Description
`isempty(fieldName: string)`	Número	Retornará 1 se o campo não for encontrado ou for uma string vazia.
`isblank(fieldName: string)`	Número	Retornará 1 se o campo não for encontrado, for uma string vazia ou só contiver espaço branco.
`concat(str: string, ...strings: string[])`	string	Concatena as strings.

`ltrim(str: string)`
 	
`ltrim(str: string, trimChars: string)`
	string	Se a função não tiver um segundo argumento, ela removerá os caracteres em branco da esquerda da string. Se a função tiver um segundo argumento de string, ela não removerá os caracteres em branco. Em vez disso, remove os caracteres em `trimChars` à esquerda de `str`. Por exemplo, `ltrim("xy ZxyfooxyZ","xyZ")` exibe `"fooxyZ"`.

`rtrim(str: string)`
 	
`rtrim(str: string, trimChars: string)`
	string	Se a função não tiver um segundo argumento, ela removerá os caracteres em branco da direita da string. Se a função tiver um segundo argumento de string, ela não removerá os caracteres em branco. Em vez disso, remove

Função	Tipo de resultado	Description
		os caracteres de `trimChars` à direita de `str`. Por exemplo, `rtrim("xyZfooxyxyZ","xyZ")` exibe `"xyZfoo"`.

`trim(str: string)`
 	
`trim(str: string, trimChars: string)`
	string	Se a função não tiver um segundo argumento, ela removerá os caracteres em branco nas duas extremidades da string. Se a função tiver um segundo argumento de string, ela não removerá os caracteres em branco. Em vez disso, remove os caracteres de `trimChars` de ambos os lados de `str`. Por exemplo, `trim("xyZxyfooxyxyZ","xyZ")` exibe `"foo"`.

`strlen(str: string)`	número	Retorna o tamanho da string em pontos de código Unicode.
`toupper(str: string)`	string	Converte a string em letras maiúsculas.
`tolower(str: string)`	string	Converte a string em letras minúsculas.

Função	Tipo de resultado	Description

`substr(str: string, startIndex: number)`
 	
`substr(str: string, startIndex: number, length: number)`
	string	Retorna uma substring do índice especificado pelo argumento de número ao final da string. Se tiver um segundo argumento de número, a função conterá o tamanho da substring a ser recuperada. Por exemplo, `substr("xyZfooxyZ",3, 3)` exibe `"foo"`.

`replace(fieldName: string, searchValue: string, replaceValue: string)`	string	Substitui todas as instâncias de `searchValue` em `fieldName: string` por `replaceValue` . Por exemplo, a função `replace(logGroup,"smoke_test","Smoke")` pesquisa eventos de log em que o campo `logGroup` contém o valor da string `smoke_test` e substitui o valor pela string `Smoke`.
`regexReplace(fieldName: string, pattern: string, replacement: string)`	string	Substitui todas as substrings que correspondem à expressão `pattern` regular com. `replacement` Usa a sintaxe regex RE2.

Função	Tipo de resultado	Description

`strcontains(str: string, searchValue: string)`
 	
`strcontains(str: string, searchValue: string, caseInsensitive: boolean)`
	número	Retornará 1 se `str` contiver `searchValue` e 0, do contrário. Se o terceiro parâmetro for definido como`true`, a correspondência não diferencia maiúsculas de minúsculas.

`startsWith(str: string, searchValue: string)`	número	Retorna 1 se `str` começar com `searchValue` e 0 caso contrário.
`endsWith(str: string, searchValue: string)`	número	Retorna 1 se `str` terminar com `searchValue` e 0 caso contrário.
`urlencode(str: string)`	string	URL-encodes a corda.
`urldecode(str: string)`	string	URL-decodes a corda.
`base64encode(str: string)`	string	Base64-encodes a corda.
`base64decode(str: string)`	string	Base64-decodes a corda.
`split(str: string, delimiter: string)`	array	Divide uma string pelo delimitador especificado e retorna uma matriz de substrings.
`hexToAscii(value: string)`	string	Converte uma string hexadecimal em texto ASCII. Exemplo: `fields hexToAscii("48656c6c6f") as text`

Função	Tipo de resultado	Description
`hexToDec(value: string)`	número	Converte uma string hexadecimal em um número decimal. Exemplo: `fields hexToDec("0xff") as dec`
`decToHex(value: number)`	string	Converte um número inteiro decimal em uma string hexadecimal minúscula com prefixo. `0x` Números negativos produzem `-0x` prefixo. Non-integers estão truncados. Exemplo: `fields decToHex(255) as hex`

Funções de hashing

Você pode usar funções de hash em `fields` e `filter` comandos.

Função	Tipo de resultado	Description
`md5(fieldName: string)`	string	Calcula o hash MD5 do valor da string.
`sha256(fieldName: string)`	string	Calcula o SHA-256 hash do valor da string.

Time-series funções

Use funções de série temporal com o `stats` comando para analisar métricas em janelas de tempo e calcular taxas de mudança.

Função	Tipo de resultado	Description
`rate(fieldName: NumericLogField, interval: Period)`	número	Calcula a taxa de alteração por intervalo para um campo numérico.
`countOverTime(fieldName: LogField)`	número	Conta os eventos de log por intervalo de tempo. Use com `by bin(interval)` para definir a janela.
`sumOverTime(fieldName: NumericLogField)`	número	Soma os valores do campo por intervalo de tempo. Use com `by bin(interval)` para definir a janela.
`histogram(fieldName: NumericLogField, buckets: number)`	mapear	Coloca os valores do campo numérico em um número especificado de intervalos de largura igual e retorna a distribuição.

deslocamento

Use `offset` no final de uma `stats ... by bin()` cláusula para deslocar compartimentos de séries temporais por uma duração especificada. Isso permite comparações com mudança de horário, como comparar as métricas atuais com o mesmo período na hora ou no dia anterior.

Sintaxe

```
stats <aggregation> by bin(<period>) offset <duration>
```

Exemplos

```
stats count(*) by bin(5m) offset 1h
```

```
stats avg(latency) by bin(1m) offset 1d
```

Campos contendo caracteres especiais

Se um campo contiver caracteres não alfanuméricos, exceto o símbolo `@` ou o ponto (`.`), você deverá delimitá-lo com caracteres de crase (`` ` ``). Por exemplo, o campo de log `foo-bar` deve estar entre acentos graves (`` `foo-bar` ``) porque contém um caractere não alfanumérico, o hífen (`-`).

Para campos do sistema que começam com `@` e contêm caracteres especiais no nome restante, coloque o `@` símbolo fora dos acentos cravos e inclua somente a parte do nome do campo. Por exemplo, os campos da tag de recursos usam o padrão`@aws.tag.<tagKey>`. Se a chave da tag contiver caracteres especiais, como dois pontos, você deverá usar a seguinte sintaxe:

```
@`aws.tag.aws:cloudformation:stack-name`
```

O exemplo a seguir mostra uma consulta de filtro que usa essa sintaxe:

```
filter @`aws.tag.aws:cloudformation:stack-name` = "my-stack"
```

Usar aliases e comentários em consultas

Crie consultas que contenham aliases. Use aliases para renomear campos de log ou ao extrair valores em campos. Use a palavra-chave `as` para dar um campo de log ou resultar em um alias. Você pode usar mais de um alias em uma consulta. Você pode usar aliases nos comandos a seguir:

- `fields`
- `parse`
- `sort`
- `stats`

Os exemplos a seguir mostram como criar consultas que contenham aliases.

Exemplo

A consulta contém um alias no comando `fields`.

```
fields @timestamp, @message, accountId as ID
| sort @timestamp desc
| limit 20
```

A consulta retorna os valores dos campos `@timestamp`, `@message` e `accountId`. Os resultados são classificados em ordem decrescente e limitados a 20. Os valores para `accountId` estão listados sob o alias `ID`.

Exemplo

A consulta contém um alias nos comandos `sort` e `stats`.

```
stats count(*) by duration as time
| sort time desc
```

A consulta conta o número de vezes que o campo `duration` ocorre no grupo de logs e classifica os resultados em ordem decrescente. Os valores para `duration` estão listados sob o alias `time`.

Usar comentários

CloudWatch O Logs Insights suporta comentários em consultas. Use o caractere jogo da velha (#) para separar comentários. Você pode usar comentários para ignorar linhas em consultas ou consultas de documentos.

Exemplo: consulta

Quando a consulta a seguir é executada, a segunda linha é ignorada.

```
fields @timestamp, @message, accountId
# | filter accountId not like "7983124201998"
| sort @timestamp desc
| limit 20
```

Comece a usar o Logs Insights QL: tutoriais de consulta

As seções a seguir incluem exemplos de tutoriais de consulta para ajudar você a começar a usar o Logs Insights QL.

Tópicos

- Tutorial: executar e modificar um exemplo de consulta
- Tutorial: Executar uma consulta com uma função de agregação
- Tutorial: Executar uma consulta que produz uma visualização agrupada por campos de log
- Tutorial: Executar uma consulta que produz uma visualização de séries temporais

Tutorial: executar e modificar um exemplo de consulta

O tutorial a seguir ajuda você a começar a usar o CloudWatch Logs Insights. Você executa uma consulta de amostra no Logs Insights QL e vê como modificá-la e voltar a executá-la.

Para executar uma consulta, você já deve ter registros armazenados em CloudWatch Registros. Se você já usa o CloudWatch Logs e tem grupos e fluxos de registros configurados, você está pronto para começar. Você também pode já ter registros se usar serviços como AWS CloudTrail o Amazon Route 53 ou o Amazon VPC e tiver configurado os registros desses serviços para CloudWatch acessar o Logs. Para obter mais informações sobre o envio de registros para o CloudWatch Logs, consulteIntrodução ao CloudWatch Logs.

As consultas no CloudWatch Logs Insights retornam um conjunto de campos de eventos de log ou o resultado de uma agregação matemática ou outra operação realizada em eventos de log. Este tutorial demonstra uma consulta que retorna uma lista de eventos de log.

Executar um exemplo de consulta

Para executar uma consulta de amostra do CloudWatch Logs Insights

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).

   Na página Logs Insights (Insights de log), o editor de consultas contém uma consulta padrão no Logs Insights QL que retorna os 20 eventos de logs mais recentes.
3. No menu suspenso Select log group(s) (Selecione grupo(s) de logs), escolha um ou mais grupos de logs a serem consultados.

   Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você poderá selecionar grupos de registros nas contas de origem e na conta de monitoramento. Uma única consulta pode consultar logs de diferentes contas ao mesmo tempo.

   Você pode filtrar os grupos de log por nome de grupo de log, ID de conta ou rótulo de conta.

Quando você seleciona um grupo de registros na classe de registros padrão, o CloudWatch Logs Insights detecta automaticamente os campos de dados no grupo. Para ver esses campos descobertos, selecione o menu Fields (Campos) próximo ao canto superior direito da página.

> **Note**
>
> A descoberta de campos só é compatível com os grupos de logs da classe de logs Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

4. (Opcional) Use o seletor de tempo para selecionar o período que você deseja consultar.

   Você pode escolher intervalos de 5 a 30 minutos, intervalos de 1, 3 e 12 horas ou um período personalizado.

5. Selecione Run (Executar) para visualizar os resultados.

   Para este tutorial, os resultados incluem os 20 eventos de log adicionados mais recentemente.

   CloudWatch Os registros exibem um gráfico de barras dos eventos de registro no grupo de registros ao longo do tempo. O gráfico de barras mostra não apenas os eventos na tabela. mas também a distribuição de eventos no grupo de logs correspondente à consulta e ao intervalo de tempo.

6. Para ver todos os campos de um evento de log retornado, escolha o ícone suspenso triangular à esquerda do evento numerado.

Modificar o exemplo de consulta

Neste tutorial, você modifica a consulta de amostra para mostrar os 50 eventos de log mais recentes.

Se você ainda não tiver executado o tutorial anterior, faça isso agora. Este tutorial começa onde esse tutorial anterior termina.

> **Note**
>
> Alguns exemplos de consultas fornecidos com o CloudWatch Logs Insights usam `tail` comandos `head` ou em vez `delimit`. Esses comandos estão defasados e foram substituídos por `limit`. Use `limit` em vez de `head` ou `tail` em todas as consultas que você gravar.

Para modificar a consulta de amostra do CloudWatch Logs Insights

1. No editor de consultas, altere 20 para 50, e escolha Executar.

   Os resultados da nova consulta são exibidos. Pressupondo-se que haja dados suficientes no grupo de logs no intervalo de tempo padrão, agora há 50 eventos de log listados.
2. (Opcional) Você pode salvar consultas que você criou. Para salvar essa consulta, escolha Salvar. Para obter mais informações, consulte Salve e execute novamente as consultas do CloudWatch Logs Insights.

Adicionar um comando de filtro ao exemplo de consulta

Este tutorial mostra como fazer uma alteração mais eficiente na consulta no editor. Neste tutorial, filtre os resultados da consulta anterior com base em um campo nos eventos de log recuperados.

Se você ainda não tiver executado os tutoriais anteriores, faça isso agora. Este tutorial começa onde esse tutorial anterior termina.

Para adicionar um comando de filtro à consulta anterior

1. Decida um campo a ser filtrado. Para ver os campos mais comuns que o CloudWatch Logs detectou nos eventos de log contidos nos grupos de log selecionados nos últimos 15 minutos e a porcentagem desses eventos de log nos quais cada campo aparece, selecione Campos no lado direito da página.

   Para ver os campos contidos em um determinado evento de log, escolha o ícone à esquerda dessa linha.

   O campo awsRegion pode ser exibido no evento de log, dependendo de quais eventos estão nos logs. No restante deste tutorial, usamos awsRegion como o campo de filtro, mas você pode usar um campo diferente caso esse campo não esteja disponível.
2. Na caixa do editor de consultas, coloque o cursor após 50 e pressione Enter.
3. Na nova linha, digite primeiro | (o caractere de barra vertical) e um espaço. Os comandos em uma consulta do CloudWatch Logs Insights devem ser separados pelo caractere de barra vertical.
4. Insira **`filter awsRegion="us-east-1"`**.
5. Escolha Executar.

A consulta é reexecutada e agora exibe os 50 resultados mais recentes correspondentes ao novo filtro.

Se tiver filtrado em um campo diferente e obtido um resultado de erro, você poderá precisar inserir caracteres de escape no nome do campo. Se o nome do campo incluir caracteres não alfanuméricos, você deverá inserir caracteres de apóstrofo (`) antes e depois do nome do campo (por exemplo, **`` `error-code`="102" ``**).

Você deve usar os caracteres de apóstrofo para nomes de campo que contenham caracteres não alfanuméricos, mas não para valores. Os valores estão sempre entre aspas (").

O Logs Insights QL contém recursos de consulta eficientes, inclusive vários comandos e suporte para expressões regulares, além de operações matemáticas e estatísticas. Para obter mais informações, consulte CloudWatch Sintaxe de consulta de linguagem do Logs Insights.

Tutorial: Executar uma consulta com uma função de agregação

Você pode usar funções de agregação no comando `stats` e como argumentos de outras funções. Neste tutorial, você executará um comando de consulta que conta o número de eventos de log contendo um campo especificado. O comando de consulta retorna uma contagem total agrupada pelo valor ou valores do campo especificado. Para obter mais informações sobre funções de agregação, consulte Operações e funções suportadas no Guia do usuário do Amazon CloudWatch Logs.

Executar uma consulta com uma função de agregação

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. Confirme se a guia Logs Insights QL está selecionada.
4. No menu suspenso Select log group(s) (Selecione grupo(s) de logs), escolha um ou mais grupos de logs a serem consultados.

   Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você poderá selecionar grupos de registros nas contas de origem e na conta de monitoramento. Uma única consulta pode consultar logs de diferentes contas ao mesmo tempo.

   Você pode filtrar os grupos de log por nome de grupo de log, ID de conta ou rótulo de conta.

Quando você seleciona um grupo de CloudWatch registros, o Logs Insights detecta automaticamente os campos de dados no grupo de registros se for um grupo de registros de classe padrão. Para ver esses campos descobertos, selecione o menu Fields (Campos) próximo ao canto superior direito da página.

5. Exclua a consulta padrão no editor de consultas e insira o seguinte comando:

```
stats count(*) by fieldName
```

6. *fieldName*Substitua por um campo descoberto no menu Campos.

   O menu Campos está localizado no canto superior direito da página e exibe todos os campos descobertos que o CloudWatch Logs Insights detecta em seu grupo de registros.

7. Selecione Run (Executar) para visualizar os resultados da consulta.

   Os resultados da consulta mostram o número de registros no grupo de logs que correspondem ao comando de consulta e a contagem total agrupada pelo valor ou valores do campo especificado.

Tutorial: Executar uma consulta que produz uma visualização agrupada por campos de log

Ao executar uma consulta que usa a função `stats` para agrupar os resultados retornados pelos valores de um ou mais campos nas entradas de log, é possível visualizar os resultados como gráfico de barras, gráfico de pizza, grafo de linhas ou grafo de área empilhada. Isso ajuda a visualizar as tendências em seus logs de forma mais eficiente.

Para executar uma consulta de visualização

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. No menu suspenso Select log group(s) (Selecione grupo(s) de logs), escolha um ou mais grupos de logs a serem consultados.

   Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você poderá selecionar grupos de registros nas contas de origem e na conta de monitoramento. Uma única consulta pode consultar logs de diferentes contas ao mesmo tempo.

   Você pode filtrar os grupos de log por nome de grupo de log, ID de conta ou rótulo de conta.

4. No editor de consultas, exclua o conteúdo atual, insira a função `stats` a seguir e escolha Run query(Executar consulta).

```
stats count(*) by @logStream
    | limit 100
```

   Os resultados mostram o número de eventos de log no grupo de logs para cada fluxo de log. Os resultados são limitados a somente 100 linhas.

5. Escolha a guia Visualization (Visualização).
6. Selecione a seta ao lado de Linha, e escolha Barra.

   O gráfico de barras será exibido, mostrando uma barra para cada fluxo de log no grupo de logs.

Tutorial: Executar uma consulta que produz uma visualização de séries temporais

Ao executar uma consulta que usa a função `bin()` para agrupar os resultados retornados por um período, é possível visualizar os resultados como um grafo de linhas, um grafo de áreas empilhadas, um gráfico de pizza ou gráfico de barras. Isso ajuda a visualizar as tendências em eventos de log ao longo do tempo de modo mais eficiente.

Para executar uma consulta de visualização

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. Confirme se a guia Logs Insights QL está selecionada.
4. No menu suspenso Select log group(s) (Selecione grupo(s) de logs), escolha um ou mais grupos de logs a serem consultados.

   Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você poderá selecionar grupos de registros nas contas de origem e na conta de monitoramento. Uma única consulta pode consultar logs de diferentes contas ao mesmo tempo.

   Você pode filtrar os grupos de log por nome de grupo de log, ID de conta ou rótulo de conta.

5. No editor de consultas, exclua o conteúdo atual, insira a função `stats` a seguir e escolha Run query(Executar consulta).

```
stats count(*) by bin(30s)
```

Os resultados mostram o número de eventos de registro no grupo de registros que foram recebidos pelo CloudWatch Logs em cada período de 30 segundos.

6. Escolha a guia Visualization (Visualização).

   Os resultados são mostrados como um gráfico de linhas. Para alternar para um gráfico de barras, de pizza ou de áreas empilhadas, escolha a seta ao lado de Line (Linha) no canto superior direito do grafo.

Exemplos de consultas

Esta seção contém uma lista de comandos de consulta gerais e úteis que você pode executar no CloudWatchconsole. Para obter informações sobre como executar um comando de consulta, consulte Tutorial: Executar e modificar uma consulta de amostra no Guia do usuário do Amazon CloudWatch Logs.

Para obter mais informações sobre a sintaxe das consultas, consulte CloudWatch Sintaxe de consulta de linguagem do Logs Insights.

Tópicos

- Consultas gerais
- Consultas de logs do Lambda
- Consultas para logs de fluxo da Amazon VPC
- Consultas de logs do Route 53
- Consultas para registros CloudTrail
- Consultas para Amazon API Gateway
- Consultas para gateway NAT
- Consultas para logs do servidor Apache
- Consultas para a Amazon EventBridge
- Consultas para tags de recursos
- Exemplos do comando de análise

Consultas gerais

Encontre os 25 eventos de log adicionados mais recentemente.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

Obtenha uma lista do número de exceções por hora.

```
filter @message like /Exception/
    | stats count(*) as exceptionCount by bin(1h)
    | sort exceptionCount desc
```

Obtenha uma lista de eventos de log que não sejam exceções.

```
fields @message | filter @message not like /Exception/
```

Obtenha o evento de log mais recente para cada valor exclusivo do campo **`server`**.

```
fields @timestamp, server, severity, message
| sort @timestamp asc
| dedup server
```

Obtenha o evento de log mais recente para cada valor exclusivo do campo **`server`** para cada tipo **`severity`**.

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server, severity
```

Consultas de logs do Lambda

Determine a quantidade de memória provisionada excessivamente.

```
filter @type = "REPORT"
    | stats max(@memorySize / 1000 / 1000) as provisonedMemoryMB,
        min(@maxMemoryUsed / 1000 / 1000) as smallestMemoryRequestMB,
        avg(@maxMemoryUsed / 1000 / 1000) as avgMemoryUsedMB,
        max(@maxMemoryUsed / 1000 / 1000) as maxMemoryUsedMB,
        provisonedMemoryMB - maxMemoryUsedMB as overProvisionedMB
```

Crie um relatório de latência.

```
filter @type = "REPORT" |
    stats avg(@duration), max(@duration), min(@duration) by bin(5m)
```

Procure invocações de funções lentas e elimine solicitações duplicadas que podem surgir de novas tentativas ou código no lado do cliente. Nessa consulta, **`@duration`** está em milissegundos.

```
fields @timestamp, @requestId, @message, @logStream
| filter @type = "REPORT" and @duration > 1000
| sort @timestamp desc
| dedup @requestId
| limit 20
```

Consultas para logs de fluxo da Amazon VPC

Encontre as 15 principais transferências de pacotes nos hosts:

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

Encontre as 15 principais transferências de bytes para hosts em uma determinada sub-rede.

```
filter isIpv4InSubnet(srcAddr, "192.0.2.0/24")
    | stats sum(bytes) as bytesTransferred by dstAddr
    | sort bytesTransferred desc
    | limit 15
```

Encontre os endereços IP que usam o UDP como um protocolo de transferência de dados.

```
filter protocol=17 | stats count(*) by srcAddr
```

Encontre os endereços IP nos quais os registros do fluxo foram ignorados durante a janela de captura.

```
filter logStatus="SKIPDATA"
    | stats count(*) by bin(1h) as t
    | sort t
```

Encontre um único registro para cada conexão para ajudar a solucionar problemas de conectividade com a rede.

```
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
```

Consultas de logs do Route 53

Encontre a distribuição de registros por hora por tipo de consulta.

```
stats count(*) by queryType, bin(1h)
```

Encontre os 10 resolvedores DNS com o maior número de solicitações.

```
stats count(*) as numRequests by resolverIp
    | sort numRequests desc
    | limit 10
```

Encontre o número de registros por domínio e subdomínio em que o servidor deixou de concluir a solicitação DNS.

```
filter responseCode="SERVFAIL" | stats count(*) by queryName
```

Consultas para registros CloudTrail

Encontre o número de entradas de registro para cada serviço, tipo de evento e AWS região.

```
stats count(*) by eventSource, eventName, awsRegion
```

Encontre os hosts do Amazon EC2 que foram iniciados ou parados em uma determinada AWS região.

```
filter (eventName="StartInstances" or eventName="StopInstances") and awsRegion="us-
east-2"
```

Encontre as AWS regiões, os nomes de usuário e os ARNs dos usuários recém-criados do IAM.

```
filter eventName="CreateUser"
    | fields awsRegion, requestParameters.userName, responseElements.user.arn
```

Encontre o número de registros em que ocorreu uma exceção durante a invocação da API **`UpdateTrail`**.

```
filter eventName="UpdateTrail" and ispresent(errorCode)
    | stats count(*) by errorCode, errorMessage
```

Encontre entradas de log em que TLS 1.0 ou 1.1 foi usado

```
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
```

Encontre o número de chamadas por serviço que usaram o TLS versões 1.0 ou 1.1

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
| stats count(*) as numOutdatedTlsCalls by eventSource
| sort numOutdatedTlsCalls desc
```

Consultas para Amazon API Gateway

Encontre os 10 últimos erros 4XX

```
fields @timestamp, status, ip, path, httpMethod
| filter status>=400 and status<=499
| sort @timestamp desc
| limit 10
```

Identifique as 10 Amazon API Gateway solicitações mais antigas em seu grupo de registros de Amazon API Gateway acesso

```
fields @timestamp, status, ip, path, httpMethod, responseLatency
| sort responseLatency desc
| limit 10
```

Retorne a lista dos caminhos de API mais populares no seu grupo de logs de acesso do Amazon API Gateway

```
stats count(*) as requestCount by path
| sort requestCount desc
| limit 10
```

Crie um relatório de latência de integração para seu grupo de logs de acesso do Amazon API Gateway

```
filter status=200
| stats avg(integrationLatency), max(integrationLatency),
min(integrationLatency) by bin(1m)
```

Consultas para gateway NAT

Se você notar custos mais altos do que o normal em sua AWS fatura, você pode usar o CloudWatch Logs Insights para encontrar os principais contribuidores. Para obter mais informações sobre os seguintes comandos de consulta, consulte Como posso encontrar os principais contribuidores para o tráfego por meio do gateway NAT na minha VPC? na página de suporte AWS premium.

> **Note**
>
> Nos comandos de consulta a seguir, substitua “x.x.x.x” pelo IP privado do gateway NAT e substitua “y.y” pelos dois primeiros octetos do intervalo CIDR da VPC.

Encontre as instâncias que estão enviando mais tráfego por meio de seu gateway NAT.

```
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
```

Determine o tráfego de/para as instâncias em seus gateways NAT.

```
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
```

Determine os destinos da Internet com os quais as instâncias em sua VPC se comunicam com mais frequência para uploads e downloads.

Para uploads

```
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
```

Para downloads

```
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
```

Consultas para logs do servidor Apache

Você pode usar o CloudWatch Logs Insights para consultar os registros do servidor Apache. Para obter mais informações sobre as consultas a seguir, consulte Simplificando os registros do servidor Apache com CloudWatch o Logs Insights no blog AWS Cloud Operations & Migrations.

Encontre os campos mais relevantes para que você possa revisar seus logs de acesso e verificar se há tráfego no caminho /admin da aplicação.

```
fields @timestamp, remoteIP, request, status, filename| sort @timestamp desc
| filter filename="/var/www/html/admin"
| limit 20
```

Encontre o número de solicitações GET exclusivas que acessaram sua página principal com o código de status “200” (sucesso).

```
fields @timestamp, remoteIP, method, status
| filter status="200" and referrer= http://34.250.27.141/ and method= "GET"
```

```
| stats countDistinct(remoteIP) as UniqueVisits
| limit 10
```

Encontre o número de vezes que o serviço Apache foi reiniciado.

```
fields @timestamp, function, process, message
| filter message like "resuming normal operations"
| sort @timestamp desc
| limit 20
```

Consultas para a Amazon EventBridge

Obtenha o número de EventBridge eventos agrupados por tipo de detalhe do evento

```
fields @timestamp, @message
| stats count(*) as numberOfEvents by `detail-type`
| sort numberOfEvents desc
```

Consultas para tags de recursos

Filtre os registros por um valor de tag de recurso.

```
filter @aws.tag.env = "production"
```

Filtre por uma chave de tag que contém caracteres especiais.

```
filter @`aws.tag.aws:cloudformation:stack-name` = "my-stack"
```

Obtenha uma contagem de eventos de log agrupados por um valor de tag.

```
stats count(*) by @aws.tag.team
```

Exemplos do comando de análise

Use uma expressão de glob para extrair os campos **`@user`**, **`@method`** e **`@latency`** do campo de log **`@message`** e retornar a latência média para cada combinação exclusiva de **`@method`** e **`@user`**.

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

Use uma expressão regular para extrair os campos **@user2**, **@method2** e **@latency2** do campo de log **@message** e retornar a latência média para cada combinação exclusiva de **@method2** e **@user2**.

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

Extrai os campos **loggingTime**, **loggingType** e **loggingMessage**, aplica o filtro para eventos de logs que contêm strings **ERROR** ou **INFO** e exibe apenas os campos **loggingMessage** e **loggingType** para eventos que contêm uma string **ERROR**.

```
FIELDS @message
    | PARSE @message "* [*] *" as loggingTime, loggingType, loggingMessage
    | FILTER loggingType IN ["ERROR", "INFO"]
    | DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

Compare (diff) com intervalos de tempo anteriores

Você pode usar o CloudWatch Logs Insights com o QL do Logs Insights para comparar as alterações em seus eventos de registro ao longo do tempo. Você pode comparar os eventos de log ingeridos durante um intervalo de tempo recente com os logs do período imediatamente anterior. Ou então, pode comparar com períodos passados similares. Isso pode ajudar a descobrir se um erro nos logs foi introduzido recentemente ou se já estava ocorrendo, além de ajudar a encontrar outras tendências.

As consultas de comparação retornam apenas padrões nos resultados, não eventos de log brutos. Os padrões retornados ajudarão você a ver rapidamente tendências e mudanças nos eventos de log ao longo do tempo. Depois que você executa uma consulta de comparação e obtém resultados de padrões, pode ver exemplos de eventos de log brutos para os padrões em que estiver interessado. Para obter mais informações sobre padrões de log, consulte Análise de padrões.

Quando você executa uma consulta de comparação, ela é analisada em relação a dois períodos diferentes: o período original de consulta que você seleciona e o período de comparação. O período de comparação é sempre igual ao período de consulta original. Os intervalos de tempo padrão para as comparações são que se seguem.

- Período anterior: compara com o período imediatamente anterior ao período da sua consulta.
- Dia anterior: compara com o período um dia antes do período da sua consulta.

- Semana anterior: compara com o período uma semana antes do período da sua consulta.
- Mês anterior: compara com o período um mês antes do período da sua consulta.

> **Note**
>
> As consultas que usam comparações geram cobranças semelhantes às da execução de uma única consulta do CloudWatch Logs Insights no intervalo de tempo combinado. Para obter mais informações, consulte Definição de preço do Amazon CloudWatch.

Para executar uma consulta de comparação

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Logs Insights.

   Uma consulta padrão aparece na caixa de consulta.
3. Confirme se a guia Logs Insights QL está selecionada.
4. Mantenha a consulta padrão ou insira outra consulta.
5. No menu suspenso Selecionar grupos de logs, escolha um ou mais grupos de logs para consultar.
6. (Opcional) Use o seletor de tempo para selecionar o período que você deseja consultar. A consulta padrão é para a hora anterior de dados de log.
7. No seletor de intervalos de tempo, escolha Comparar. Depois, escolha o período anterior com o qual você deseja comparar os logs originais e escolha Aplicar.
8. Selecione Executar consulta.

   Para que a consulta busque os dados no período de comparação, o comando `diff` é adicionado ao fim da consulta ela.
9. Escolha a guia Padrões para ver os resultados.

   A tabela exibe as seguintes informações:

   - Cada padrão, com as partes variáveis do padrão substituídas pelo símbolo de token dinâmico <*string*-*number*>. *string*É uma descrição do tipo de dados que o token representa. *number*Mostra onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos. Para obter mais informações, consulte Análise de padrões.

- Número de eventos é o número de eventos de log com esse padrão no período original mais atual.
- Diferença de número de eventos é a diferença entre o número de eventos de log correspondentes no período atual em relação ao período de comparação. Uma diferença positiva significa que há mais eventos desse tipo no período atual.
- Descrição da diferença resume brevemente a alteração no padrão entre o período atual e o período de comparação.
- Tipo de gravidade é a provável gravidade dos eventos de log com esse padrão, baseado nas palavras encontradas nos eventos de log, como `FATAL`, `ERROR`, e `WARN`.

10. Para inspecionar melhor um dos padrões na lista, escolha o ícone na coluna Inspecionar para um dos padrões.

    O painel Inspeção de padrões aparece e exibe o seguinte:

    - O padrão. Selecione um token no padrão para analisar os valores desse token.
    - Um histograma mostrando o número de ocorrências do padrão no intervalo consultado. Isso pode ajudar a identificar tendências interessantes, como um aumento repentino na ocorrência de um padrão.
    - A guia Exemplos de log exibe alguns eventos de log que correspondem ao padrão selecionado.
    - A guia Valores do token exibe os valores do token dinâmico selecionado, se você tiver selecionado um.

      > **Note**
      >
      > No máximo de 10 valores de token são capturados para cada token. A contagem de tokens pode não ser precisa. CloudWatch O Logs usa um contador probabilístico para gerar a contagem de tokens, não o valor absoluto.

    - A guia Padrões relacionados exibe outros padrões que ocorreram com frequência quase ao mesmo tempo que o padrão que você está inspecionando. Por exemplo, se um padrão para uma mensagem `ERROR` geralmente foi acompanhado por outro evento de log marcado como `INFO` com detalhes adicionais, esse padrão será exibido aqui.

Visualize dados de log em grafos

Você pode usar visualizações como gráficos de barras, gráficos de linhas e gráficos de áreas empilhadas para identificar com mais eficiência padrões em seus dados de registro. CloudWatch O Logs Insights gera visualizações para consultas que usam a `stats` função e uma ou mais funções de agregação. Para obter mais informações, consulte stats.

OpenSearch Linguagem de processamento canalizado (PPL)

Esta seção contém uma introdução básica à consulta de CloudWatch registros usando OpenSearch PPL. Com a PPL, você pode recuperar, consultar e analisar dados usando comandos agrupados, facilitando a compreensão e a composição de consultas complexas. Sua sintaxe é baseada em pipes Unix e permite o encadeamento de comandos para transformar e processar dados. Com a PPL, você pode filtrar e agregar dados e usar um amplo conjunto de funções matemáticas, de sequências de caracteres, de data, condicionais e outras para análise.

Incluir `SOURCE` em uma consulta PPL é uma forma útil de especificar os grupos de registros, os índices de campo e as fontes de dados a serem incluídos em uma consulta quando você estiver usando a API AWS CLI ou para criar uma consulta. O `SOURCE` comando é suportado somente na API AWS CLI and, não no CloudWatch console. Ao usar o CloudWatch console para iniciar uma consulta, você usa a interface do console para especificar os grupos de log e o nome e o tipo da fonte de dados.

Use `aws:fieldIndex` para retornar somente dados indexados, forçando uma consulta a verificar somente grupos de logs indexados em um campo especificado na consulta. Os grupos de registros relevantes são selecionados automaticamente, com base nos campos especificados no `filterIndex` comando. Isso reduz o volume escaneado ignorando grupos de registros que não têm nenhum evento de registro contendo o campo especificado na consulta e somente grupos de registros de varredura que correspondam ao valor especificado na consulta para esse índice de campo. Use `aws:fieldIndex` para especificar o nome do campo, junto com o nome e o valor do campo no comando de origem para consultar somente dados indexados contendo o campo e o valor especificados. Para obter mais informações, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.

Você pode usar o OpenSearch PPL para consultas de grupos de registros na Classe de Registro Padrão.

> **Note**
>
> Para obter informações sobre todos os comandos de consulta OpenSearch PPL compatíveis com CloudWatch os registros e informações detalhadas sobre sintaxe e restrições, consulte Comandos PPL compatíveis no OpenSearch Service Developer Guide.
> Para obter informações sobre outras linguagens de consulta que você pode usar, consulte CloudWatch Logs Insights, OpenSearch Service SQL e CloudWatch Metrics Insights

Comando ou função	Consulta de exemplo	Description
`fields`	`fields field1, field2`	Exibe um conjunto de campos que precisam de projeção.
`join`	`LEFT JOIN left=l, right=r on l.id = r.id` `` `join_right_lg` `` `| fields l.field_1, r.field_2`	Une dois conjuntos de dados.
`where`	`where field1="success" | where field2 != "i-023fe0a90929d8822" | fields field3, field4, field5,field6 | head 1000`	Filtra os dados com base nas condições que você especifica.
`aws:field Index`	`source = [` `` `aws:fieldIndex` `` `="region",` `` `region` `` `= "us-west-2"] | where status = 200 | head 10`	Retorna somente dados indexados, forçando uma consulta a examinar somente grupos de registros que estão indexados em um campo que

Comando ou função	Consulta de exemplo	Description
		você especifica na consulta.
stats	`stats count(), count(field1), min(field 1), max(field1), avg(field1) by field2 | head 1000`	Executa agregações e cálculos
parse	`parse field1 ".*/(?<field2>[^/]+$)" | where field2 = "requestId" | fields field1, field2 | head 1000`	Extrai um padrão de expressão regular (regex) de uma string e exibe o padrão extraído. O padrão extraído pode ser usado posteriormente para criar novos campos ou filtrar dados.
sort	`stats count(), count(field1), min(field 1) as field1Alias, max(`field1`), avg(`field1`) by field2 | sort -field1Al ias | head 1000`	Classifique os resultados exibidos por um nome de campo. Use classific ar - FieldName para classific ar em ordem decrescente.

Comando ou função	Consulta de exemplo	Description
`eval`	`eval field2 = field1 * 2 | fields field1, field2 | head 20`	Modifica ou processa o valor de um campo e o armazena em outro campo. Esse comando é útil para modificar matematic amente uma coluna, aplicar funções de string a uma coluna ou aplicar funções de data a uma coluna.
`rename`	`rename field2 as field1 | fields field1;`	Renomeia um ou mais campos no resultado da pesquisa.
`head`	`fields `@message` | head 20`	Limita os resultados da consulta exibidos às primeiras N linhas.
`top`	`top 2 field1 by field2`	Encontra os valores mais frequentes para um campo.

Comando ou função	Consulta de exemplo	Description
dedup	`dedup field1 | fields field1, field2, field3`	Remove entradas duplicadas com base nos campos que você especifica.
rare	`rare field1 by field2`	Encontra os valores menos frequentes de todos os campos na lista de campos.
subquery	`where field_1 IN [ search source= `subquery_lg` | fields field_2 ] | fields id, field_1`	Executa consultas complexas e aninhadas em suas instruções de PPL.
trendline	`trendline sma(2, field1) as field1Alias`	Calcula as médias móveis dos campos.

Comando ou função	Consulta de exemplo	Description
eventStats	`eventstats sum(field1) by field2`	Enriquece os dados de evento com estatísticas resumidas calculadas. Esse comando analisa campos especificados em seus eventos, calcula várias medidas estatísticas e, em seguida, anexa esses resultados a cada evento original como novos campos.
expand	`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`	Divide um campo contendo vários valores em linhas separadas, criando uma nova linha para cada valor no campo especificado.

Comando ou função	Consulta de exemplo	Description
fillnull	`fields `@timestamp`, error_code, status_code | fillnull using status_code = "UNKNOWN", error_code = "UNKNOWN"`	Preenche campos nulo com o valor que você fornece. Pode ser usado em um ou mais campos.
flatten	`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)`	Nivela um campo. O campo deve ser do tipo: `struct<?,?>` ou `array<struct<?,?>>` .
cidrmatch	`where cidrmatch(ip, '2003:db8::/32') | fields ip`	Verifica se o endereço IP especificado está dentro do intervalo CIDR especificado.
fieldsummary	`where field1 != 200 | fieldsummary includefields= field1 nulls=true`	Calcula estatísticas básicas para cada campo (contagem, contagem distinta, mín, máx, méd., stddev e mediana).

Comando ou função	Consulta de exemplo	Description
grok	`grok email '.+@%{HOSTNAME:host}' | fields email, host`	Analisa um campo de texto com um padrão Grok e anexa os resultados ao resultado da pesquisa.
Funções de string	`eval field1Len = LENGTH(field1) | fields field1Len`	Built-in funções no PPL que podem manipular e transformar dados de string e texto em consultas PPL. Por exemplo, converter maiúsculas e minúsculas, combinar strings, extrair partes e apagar texto.

Comando ou função	Consulta de exemplo	Description
Date-Time funções	`eval newDate = ADDDATE(DATE('2020-08-26'), 1) | fields newDate`	Built-in funções para lidar e transformar dados de data e carimbo de data/hora em consultas PPL. Por exemplo: date_add, date_format, datediff, date-sub, timestamp add, timestamp diff, current_timezone, utc_timestamp, e current_date.
Funções de condição	`eval field2 = isnull(field1) | fields field2, field1, field3`	Built-in funções que verificam condições de campo específicas e avaliam expressões condicionalmente. Por exemplo, se campo1 for nulo, retorne campo2.

Comando ou função	Consulta de exemplo	Description
Funções matemáticas	`eval field2 = ACOS(field1) | fields field1`	Built-in funções para realizar cálculos e transformações matemáticas em consultas PPL. Por exemplo: abs (valor absoluto), round (arredonda números), sqrt (raiz quadrada), pow (cálculo de potência) e ceil (arredonda até o número inteiro mais próximo).
CryptoGraphic funções	`eval crypto = MD5(field)| head 1000`	Para calcular o hash de um determinado campo

Comando ou função	Consulta de exemplo	Description
Funções JSON	`eval valid_json = json('[1,2,3,{"f1":1,"f2":[5,6]},4]') | fields valid_json`	Built-in funções para lidar com JSON, incluindo matrizes, extração e validação. Por exemplo: json_object, json_array, to_json_string, json_array_length, json_extract, json_keys e json_valid.

Escopo da consulta

Incluir SOURCE em uma consulta é uma forma útil de especificar os grupos de registros a serem incluídos em uma consulta quando você estiver usando a API AWS CLI ou para criar uma consulta. O comando SOURCE é suportado somente na API AWS CLI e, não no CloudWatch console. Ao usar o CloudWatch console para iniciar uma consulta, você usa a interface do console para especificar os grupos de log e o nome e o tipo da fonte de dados.

O comando source do PPL agora oferece suporte a várias maneiras de especificá-los:

1. Grupo de registros
2. Índices de campo - Novo
3. Fonte e tipo de dados - Novo

Grupo de logs

A seleção da fonte do grupo de registros pode ser usada quando os clientes sabem quais grupos de registros exatos precisam ser pesquisados

```
source = [lg:`/aws/lambda/my-function`] | where status = 200 | head 10
```

Índices de campo

A seleção da fonte com base no índice de campo reduz a quantidade de dados consultados ao limitar os resultados somente aos dados indexados quando seus filtros segmentam campos que foram indexados. Os grupos de registros relevantes são selecionados automaticamente, com base nos campos especificados no `filterIndex` comando. Para obter mais informações sobre índices de campo e como criá-los, consulte Criar índices de campo para melhorar o desempenho da consulta e reduzir o volume de varredura.

Use `aws:fieldIndex` para retornar somente dados indexados, forçando uma consulta a verificar somente grupos de logs indexados em um campo especificado na consulta. Para esses grupos de logs indexados nesse campo, ele otimiza ainda mais a consulta ignorando os grupos que não têm nenhum evento de logs contendo o campo especificado na consulta do campo indexado. Isso reduz ainda mais o volume escaneado ao tentar verificar somente eventos de logs desses grupos que correspondam ao valor especificado na consulta para esse índice de campo. Para obter mais informações sobre índices de campo e como criá-los, consulte Criar índices de campo para melhorar o desempenho da consulta e reduzir o volume de varredura.

Em PPL, `aws:fieldIndex` é usado para especificar quais pares de valores-chave devem ser tratados como índices. A sintaxe é a seguinte

```
source = [`aws:fieldIndex`="region", `region` = "us-west-2"] | where status = 200 |
 head 10
```

em que,

1. `` `aws:fieldIndex`="region" `` identifica a região como índice de campo.
   a. Observação: em vez de =, os clientes podem usar IN para especificar vários índices (exemplo abaixo)
2. `` `region`="us-west-2" `` identifica a condição do filtro a ser aplicada
   a. Observação: em vez de =, os clientes podem usar IN para especificar vários valores (exemplo abaixo)

Os clientes podem especificar vários índices de campo da seguinte forma

```
source = [`aws:fieldIndex` IN ("status", "region"), `status` = 200, `region` IN ("us-
west-2", "us-east-1")] | head 10
```

Fonte e tipo de dados

A seleção da fonte de dados e da fonte baseada no tipo pode ser usada quando os clientes sabem quais fontes de dados exatas precisam ser consultadas. Essa consulta é executada em um ou mais grupos de registros que contêm a fonte e o tipo de dados especificados.

```
source = [ds:`data_source.type`] | where status = 200 | head 10
```

PPL compatível para consultas de fontes de dados

Para oferecer suporte ao caso de uso da consulta de fontes de dados no PPL, você pode usar a cláusula seletora dinâmica de origem. Usando essa sintaxe, você pode consultar fontes de dados especificando-as no comando search. Você pode especificar até 10 fontes de dados.

Sintaxe

```
source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`, ...ds:`DataSourcen.Typen`]
```

Consulta de exemplo

```
search source=[ds:`DataSource1.Type1`, ds:`DataSource2.Type2`] | fields field1, field2
```

Exemplo de combinação

Os clientes podem especificar todos os operadores de seleção de origem em qualquer ordem e os resultados seriam a interseção de todas as condições aplicadas.

Por exemplo,/aws/lambda/my-function-1 pode conter várias fontes e tipos de dados, incluindo uma grande variedade de índices. Quando a consulta a seguir for executada, os resultados retornados terão apenas eventos de origem e tipo DataSource1.Type1 e corresponderão aos critérios de 'status' = 200.

```
search source=[
    ds:`DataSource1.Type1`,
    lg:`/aws/lambda/my-function-1`,
    `aws:fieldIndex` IN ("status"), `status` = 200
]
```

Restrições

As restrições a seguir se aplicam quando você usa o OpenSearch PPL para fazer consultas no CloudWatch Logs Insights.

- Você não pode usar comandos de junção ou subconsulta com consultas de fonte de dados.

OpenSearch Linguagem de consulta estruturada (SQL)

Esta seção contém uma introdução básica à consulta de CloudWatch registros usando OpenSearch SQL. Ela fornece uma opção mais conhecida se você estiver acostumado a trabalhar com bancos de dados relacionais. OpenSearch O SQL oferece um subconjunto da funcionalidade SQL, o que o torna uma boa opção para realizar consultas ad-hoc e tarefas de análise de dados. Com o OpenSearch SQL, você pode usar comandos como SELECT, FROM, WHERE, GROUP BY, HAVING e vários outros comandos e funções SQL. Você pode executar JOINs em grupos de logs, correlacionar dados entre grupos de logs usando sub-consultas e usar o amplo conjunto de funções JSON, de matemáticas, de strings, condicionais e outras funções SQL para realizar análises poderosas em dados de log e de segurança.

Use `filterIndex` para retornar somente dados indexados, forçando uma consulta a verificar somente grupos de logs indexados em um campo especificado na consulta. Reduza o volume escaneado ignorando grupos de registros que não têm nenhum evento de registro contendo o campo especificado na consulta e somente grupos de registros de varredura que correspondam ao valor especificado na consulta para esse índice de campo. Use `filterIndex` para especificar o nome do campo, junto com o nome e o valor do campo para consultar somente dados indexados contendo o campo e o valor especificados.

Você pode usar o OpenSearch SQL para consultas de grupos de registros na Classe de Registro Padrão. O SQL também oferece suporte à consulta usando o nome e o tipo da fonte de dados.

> **Note**
>
> A tabela a seguir lista os comandos e funções SQL compatíveis com os CloudWatch registros. Para obter informações sobre todos os comandos OpenSearch SQL, incluindo a sintaxe, consulte Comandos SQL compatíveis no OpenSearch Service Developer Guide. Para obter informações sobre outras linguagens de consulta que você pode usar, consulte CloudWatch Logs Insights, OpenSearch Service PPL e. CloudWatch Metrics Insights

Comandos SQL compatíveis

> **Note**
>
> Na coluna de consulta de exemplo, substitua *<logGroup>* conforme necessário, dependendo da fonte de dados que você está consultando.

Comando ou função	Consulta de exemplo	Description
SELECT	`SELECT \`@message\`, Operation FROM \`LogGroupA\``	Exibe os valores projetados.
FROM	`SELECT \`@message\`, Operation FROM \`LogGroupA\``	Built-in cláusula que especific a a (s) tabela (s) de origem ou visualiza ções das quais recuperar dados, suportando vários tipos de junções e subconsultas.
WHERE	`SELECT * FROM \`LogGroupA\` WHERE Operation = 'x'`	Filtra eventos de logs com base nos critérios de campo fornecido s.
filterIndex	`SELECT * FROM \`filterIndex('region' = 'us-east-1')\` WHERE status = 200 LIMIT 10;`	Retorna somente dados indexados , forçando uma consulta a examinar

Comando ou função	Consulta de exemplo	Description
		somente grupos de registros que estão indexados em um campo que você especifica na consulta.
GROUP BY	`SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream``	Agrupa eventos de logs com base na categoria e encontra a média com base nas estatísticas.
HAVING	`SELECT `@logStream`, COUNT(*) as log_count FROM `LogGroupA` GROUP BY `@logStream` HAVING log_count > 100`	Filtra os resultados com base nas condições de agrupamento.
ORDER BY	`SELECT * FROM `LogGroupA` ORDER BY `@timestamp` DESC`	Ordena os resultados com base nos campos da cláusula ORDER BY. É possível classificar em ordem crescente ou decrescente.

Comando ou função	Consulta de exemplo	Description
JOIN	`SELECT A.`@message`, B.`@timestamp`FROM `LogGroupA` as A INNER JOIN `LogGroupB` as B ON A.`requestId` = B.`requestId``	Une os resultados de duas tabelas com base em campos comuns. Deve-se especificar Inner JOIN ou Left Outer Join
LIMIT	`Select * from `LogGroupA` limit 10`	Limita os resultados da consulta exibidos às primeiras N linhas.
Funções de string	`SELECT upper(Operation) , lower(Operation), Operation FROM `LogGroupA``	Built-in funções em SQL que podem manipular e transformar dados de string e texto em consultas SQL. Por exemplo, converter maiúsculas e minúsculas, combinar strings, extrair partes e apagar texto.

Comando ou função	Consulta de exemplo	Description
Funções de data	`SELECT current_date() as today, date_add( current_date(), 30) as thirty_da ys_later, last_day(current_date()) as month_end FROM `LogGroupA``	Built-in funções para lidar e transformar dados de data e carimbo de data/hora em consultas SQL. Por exemplo: date_add, date_form at, datediff e current_date.
Funções condicionais	`SELECT Operation, IF(Error > 0, 'High', 'Low') as error_category FROM `LogGroup A`;`	Built-in funções que executam ações com base em condições especificadas ou que avaliam expressõe s condicion almente. Por exemplo: CASE e IF.

Comando ou função	Consulta de exemplo	Description
Funções agregadas	`SELECT AVG(bytes) as bytesWritten FROM` `` `LogGroupA` ``	Built-in funções que realizam cálculos em várias linhas para produzir um único valor resumido. Por exemplo: SUM, COUNT, AVG, MAX e MIN.
Funções JSON	`SELECT get_json_object(json_column, '$.name') as name FROM` `` `LogGroupA` ``	Built-in funções para analisar, extrair, modificar e consultar JSON-form atted dados em consultas SQL (por exemplo, from_json , to_json, get_json_object, json_tuple) permitindo a manipulação de estruturas JSON em conjuntos de dados.

Comando ou função	Consulta de exemplo	Description
Funções de array	`SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;`	Built-in funções para trabalhar com colunas do tipo array em consultas SQL, permitindo operações como acessar, modificar e analisar dados de matriz (por exemplo, size, explode, array_contains).
Funções de janela	`SELECT field1, field2, RANK() OVER (ORDER BY field2 DESC) as field2Rank FROM `LogGroupA`;`	Built-in funções que realizam cálculos em um conjunto específico de linhas relacionadas à linha (janela) atual, permitindo operações como classificação, totais acumulados e médias móveis. Por exemplo: ROW_NUMBER, RANK, LAG e LEAD

Comando ou função	Consulta de exemplo	Description
Funções de conversão	`SELECT CAST('123' AS INT) as converted_number, CAST(123 AS STRING) as converted_string FROM `LogGroupA``	Built-in funções para converter dados de um tipo para outro em consultas SQL, permitindo transformações de tipos de dados e conversões de formato. Por exemplo: CAST, TO_DATE, TO_TIMESTAMP e BINARY.
Funções de predicado	`SELECT scores, size(scores) as length, array_contains(scores, 90) as has_90 FROM `LogGroupA`;`	Built-in funções que avaliam condições e retornam valores booleanos (true/false) com base em critérios ou padrões especificados. Por exemplo: IN, LIKE, BETWEEN, IS NULL e EXISTS.

Comando ou função	Consulta de exemplo	Description
Selecione vários grupos de logs	`SELECT lg1.field1, lg1.field2 from` `logGroups( logGroupIdentifier: ['LogGroup1', 'LogGroup2'])` as lg1 where lg1.field3= "Success"	Permite que você especifiq ue vários grupos de logs em uma instrução SELECT
Selecione várias fontes de dados	`SELECT ds1.field1, ds1.field2 from` `dataSource(['DataSource1', 'DataSour ce2'])` as ds1 where ds1.field3= "Success"	Permite que você especifiq ue várias fontes de dados em uma instrução SELECT

SQL compatível para consultas de vários grupos de logs

Para oferecer suporte ao caso de uso para consultar vários grupos de logs no SQL, você pode usar o comando `logGroups`. Usando essa sintaxe, é possível consultar vários grupos de logs especificando-os no comando FROM.

Sintaxe:

```
`logGroups(
    logGroupIdentifier: ['LogGroup1','LogGroup2', ...'LogGroupn']
)`
```

Nessa sintaxe, é possível especificar até 50 grupos de logs no parâmetro `logGroupIdentifier`. Para referenciar grupos de logs em uma conta de monitoramento, use ARNs em vez de nomes de `LogGroup`.

Consulta de exemplo:

```
SELECT LG1.Column1, LG1.Column2 from `logGroups(
    logGroupIdentifier: ['LogGroup1', 'LogGroup2']
)` as LG1 WHERE LG1.Column1 = 'ABC'
```

A sintaxe a seguir, envolvendo vários grupos de registros após a FROM declaração, NÃO é compatível com a consulta CloudWatch de registros.

```
SELECT Column1, Column2 FROM 'LogGroup1', 'LogGroup2', ...'LogGroupn'
WHERE Column1 = 'ABC'
```

SQL compatível para consultas de fontes de dados

Para oferecer suporte ao caso de uso para consultar fontes de dados em SQL, você pode usar o comando DataSource. Usando essa sintaxe, você pode consultar fontes de dados especificando-as no FROM comando. Você pode especificar até 10 fontes de dados.

Sintaxe

```
`dataSource(
    ['DataSource1', 'DataSource2', ...'DataSourcen']
)`
```

Exemplo de consulta

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(
    ['DataSource1', 'DataSource2']
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

Escopo da consulta

Na API AWS CLI and, você pode especificar quais registros consultar usando o grupo de registros, a fonte e o tipo de dados e os índices de campo.

Grupo de logs

A seleção da fonte do grupo de registros pode ser usada quando os clientes sabem quais grupos de registros exatos precisam ser pesquisados

```
SELECT * FROM `logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])`;
```

Fonte e tipo de dados

Os clientes podem consultar seus registros usando o nome e o tipo da fonte de dados.

A seleção da fonte de dados e da fonte baseada no tipo pode ser usada quando os clientes sabem quais fontes de dados exatas precisam ser consultadas. Essa consulta é executada em um ou mais grupos de registros que contêm a fonte e o tipo de dados especificados.

Para oferecer suporte ao caso de uso para consultar fontes de dados em SQL, você pode usar o comando DataSource. Usando essa sintaxe, você pode consultar fontes de dados especificando-as no comando FROM. Você pode especificar até 10 fontes de dados.

Sintaxe:

```
`dataSource(
    ['DataSource1.Type1', 'DataSource2.Type2', ...'DataSourcen.Typen']
)`
```

Consulta de exemplo:

```
SELECT DS1.Column1, DS1.Column2 from `dataSource(
    ['DataSource1.Type1', 'DataSource2.Type2']
)` as DS1 WHERE DS1.Column1 = 'ABC'
```

Para obter mais informações sobre consultas por fontes de dados, consulteUse facetas para agrupar e explorar registros.

Exemplo de combinação

Os clientes podem especificar todos os operadores de seleção de origem dentro dos acentos em qualquer ordem e os resultados seriam baseados na interseção de todas as condições aplicadas.

Por exemplo,/aws/lambda/my-function-1 pode conter várias fontes e tipos de dados, incluindo uma grande variedade de índices. Quando a consulta a seguir for executada, os resultados retornados terão apenas eventos de origem e tipo DataSource1.Type1 e corresponderão aos critérios de 'status' = 200.

```
SELECT * FROM `
   logGroups(logGroupIdentifier: ['/aws/lambda/my-function'])
   filterIndex('status' = 200)
   dataSource(['DataSource1.Type1'])
`;
```

Índices de campo

A seleção da Index-based fonte de campo identifica automaticamente grupos de registros relevantes quando seus filtros têm como alvo campos indexados, reduzindo o volume de escaneamento e o tempo de execução da consulta.

Use `filterIndex` para retornar somente dados indexados, forçando uma consulta a verificar somente grupos de logs indexados em um campo especificado na consulta. Para esses grupos de logs indexados nesse campo, ele otimiza ainda mais a consulta ignorando os grupos que não têm nenhum evento de logs contendo o campo especificado na consulta do campo indexado. Isso reduz ainda mais o volume escaneado ao tentar verificar somente eventos de logs desses grupos que correspondam ao valor especificado na consulta para esse índice de campo. Para obter mais informações sobre índices de campo e como criá-los, consulte Criar índices de campo para melhorar o desempenho da consulta e reduzir o volume de varredura.

No SQL, filterIndex é usado para especificar quais pares de valores-chave devem ser tratados como índices. A sintaxe é a seguinte

```
SELECT * FROM `filterIndex('region' = 'us-east-1')`;
```

em que,

1. filterIndex (...) especifica, trata os valores-chave dentro deles como índices de campo. Cada par de valores-chave é separado por uma vírgula (exemplo abaixo)
2. 'region' = 'us-east-1' especifica a condição real a ser aplicada
   a. Observação: em vez de =, os clientes podem usar IN para especificar vários valores (exemplo abaixo)

Usar vários FilterIndex combinaria as condições usando “AND”. No exemplo, registros que correspondam a status=200 e region em us-east-1 ou us-west-2 seriam consultados.

```
SELECT * FROM `filterIndex('status' = 200, 'region' IN ['us-east-1', 'us-west-2'])`;
```

Restrições

As restrições a seguir se aplicam quando você usa OpenSearch SQL para fazer consultas no CloudWatch Logs Insights.

- É possível incluir somente um JOIN em uma instrução SELECT.
- Você não pode usar JOIN ou subconsultas com consultas de fonte de dados.
- Somente um nível de sub-consultas aninhadas é suportado.
- Não há suporte para várias consultas de instruções separadas por ponto e vírgula (;).
- Não há suporte para consultas contendo nomes de campo idênticos, mas que diferem somente em maiúsculas e minúsculas (como campo1 e CAMPO1).

  Por exemplo, a consulta a seguir não é suportada:

  ```
  Select AWSAccountId, AwsAccountId from LogGroup
  ```

  No entanto, a consulta a seguir é compatível porque o nome do campo (`@logStream`) é idêntico nos dois grupos de logs:

  ```
  Select a.`@logStream`, b.`@logStream` from Table A INNER Join Table B on a.id = b.id
  ```

- Funções e expressões devem operar em nomes de campo e fazer parte de uma instrução SELECT com um grupo de logs especificado na cláusula FROM.

  Por exemplo, esta consulta não é compatível:

  ```
  SELECT cos(10) FROM LogGroup
  ```

  Esta consulta é suportada:

  ```
  SELECT cos(field1) FROM LogGroup
  ```

- Ao usar comandos SQL ou PPL, coloque certos campos entre crases para consultá-los com êxito. As crases são necessárias para campos com caracteres especiais (não alfabéticos e não numéricos). Por exemplo, coloque `@message`, `Operation.Export` e `Test::Field` entre crases. Você não precisa colocar entre crases os campos com nomes puramente alfabéticos.

  Exemplo de consulta com campos simples:

  ```
  SELECT SessionToken, Operation, StartTime  FROM `LogGroup-A`
  LIMIT 1000;
  ```

  Consulta semelhante com acentos graves acrescentados:

```
SELECT `@SessionToken`, `@Operation`, `@StartTime`  FROM `LogGroup-A` LIMIT 1000;
```

Use linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights

CloudWatch O Logs oferece suporte a um recurso de consulta em linguagem natural para ajudar você a gerar e atualizar consultas para CloudWatch Logs Insights, OpenSearch Service PPL, OpenSearch Service SQL e. CloudWatch Metrics Insights

Com esse recurso, você pode fazer perguntas ou descrever os dados de CloudWatch registros que está procurando em inglês simples. O recurso de linguagem natural gera uma consulta com base em um prompt que você envia e fornece uma explicação detalhada de como a consulta funciona. Você também pode atualizar a consulta para investigar melhor seus dados.

Dependendo do ambiente, você pode inserir prompts, como "Quais os 100 principais endereços IP de origem por bytes transferidos?" e "Encontre as dez solicitações de função do Lambda mais lentas".

> **Note**
>
> O recurso de consulta em linguagem natural é um serviço regional. Para algumas regiões, o recurso faz chamadas entre regiões para regiões nos Estados Unidos para processar os prompts de consulta. Para obter mais informações, consulte Amazon CloudWatch expande o suporte regional para resumo de resultados de consultas em linguagem natural e geração de consultas.

Para gerar uma consulta do CloudWatch Logs Insights com esse recurso, abra o editor de consultas do CloudWatch Logs Insights, selecione o grupo de registros que você deseja consultar e escolha Gerar consulta.

> **Important**
>
> Para usar o recurso de consulta em linguagem natural, você deve estar conectado com as políticas CloudWatchLogsFullAccessCloudWatchLogsReadOnlyAccess,

AdministratorAccess,, ou ReadOnlyAccessdo IAM, ou ter a `cloudwatch:GenerateQuery` permissão.

Consultas de exemplo

Os exemplos nesta seção descrevem como gerar e atualizar consultas usando o recurso de linguagem natural.

> **Note**
>
> Para obter mais informações sobre o editor de consultas e a sintaxe do CloudWatch Logs Insights, consulte Sintaxe de consulta do CloudWatch Logs Insights.

Exemplos: gerar uma consulta em linguagem natural

Para gerar uma consulta usando linguagem natural, insira um prompt e escolha Gerar nova consulta. Este exemplo mostra consultas que executam uma pesquisa básica.

Prompt

O exemplo de prompt a seguir instrui o recurso a pesquisar as dez invocações de funções do Lambda mais lentas.

```
Find the 10 slowest requests
```

Consulta

A seguir está a consulta usando a linguagem de consulta do CloudWatch Logs Insights que o recurso de linguagem natural gerou com base na solicitação. Observe como o prompt aparece em um comentário antes da consulta. Depois da consulta, você pode ler uma explicação que descreve como a consulta funciona.

```
# 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**
>
> Para desativar o surgimento do prompt e a explicação de como a consulta funciona, use o ícone de engrenagem no editor.

Prompt

Para gerar uma consulta OpenSearch SQL, selecione a guia OpenSearch SQL e, em seguida, abra a caixa de solicitação do gerador de consultas para inserir sua solicitação de linguagem natural. Veja a seguir um exemplo de um prompt que usa o recurso de linguagem natural para gerar uma consulta OpenSearch SQL.

```
Give me the number of errors and exceptions per hour
```

Consulta

A seguir está a consulta SQL gerada por esse prompt que você pode usar para encontrar o número de erros e exceções agregados por hora:

```
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

Para gerar uma consulta OpenSearch PPL, selecione a guia OpenSearch PPL e, em seguida, abra a caixa de solicitação do gerador de consultas para inserir sua solicitação em linguagem natural. Veja a seguir um exemplo de um prompt que usa o recurso de linguagem natural para gerar uma consulta OpenSearch PPL.

```
Give me all unique exception messages
```

Consulta

Veja a seguir a consulta PPL gerada por esse prompt que você pode usar para encontrar as mensagens de exceção exclusivas em seus logs:

```
dedup @message
| fields @message
```

Exemplo: atualizar uma consulta em linguagem natural

Você pode atualizar uma consulta editando o prompt inicial e escolhendo Atualizar consulta.

Prompt atualizado

O exemplo a seguir mostra uma versão atualizada do prompt anterior. Em vez de um prompt que pesquisa as dez invocações de funções do Lambda mais lentas, esse prompt agora instrui o recurso a pesquisar as vinte invocações de funções do Lambda mais lentas e a incluir outra coluna para eventos de log adicionais.

```
Show top 20 slowest requests instead and display requestId as a column
```

Consulta atualizada

Veja a seguir um exemplo da consulta atualizada usando a linguagem de consulta CloudWatch Logs Insights. Observe como o prompt atualizado aparece em um comentário antes da consulta atualizada. Depois da consulta, você pode ler uma explicação que descreve como a consulta original foi atualizada.

```
# 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.
```

Optar por não usar seus dados para melhorar o serviço

Os dados dos prompts em linguagem natural que você fornece para treinar o modelo de IA e gerar consultas relevantes são usados exclusivamente para fornecer e manter seu serviço. Esses dados

podem ser usados para melhorar a qualidade do CloudWatch Logs Insights. Sua confiança e privacidade, além da segurança do seu conteúdo, são nossas maiores prioridades. Para obter mais informações, consulte Termos de Serviço da AWS e AWS responsible AI policy.

Você pode se recusar a ter seu conteúdo usado para desenvolver ou melhorar a qualidade das consultas em linguagem natural ao criar uma política de rejeição de serviços de IA. Para desativar a coleta de dados de todos os recursos do CloudWatch Logs AI, incluindo o recurso de geração de consultas, você deve criar uma política de exclusão para o Logs. CloudWatch Para obter mais informações, consulte Políticas de exclusão dos serviços de IA no Guia do usuário do AWS Organizations .

Logs compatíveis e campos descobertos

CloudWatch O Logs Insights oferece suporte a diferentes tipos de registros. Para cada registro enviado para um grupo de registros da classe Standard no Amazon CloudWatch Logs, o CloudWatch Logs Insights gera automaticamente cinco campos do sistema:

- `@message` contém o evento de log bruto não avaliado. Isso é o equivalente ao `message` campo em InputLogevent.
- `@timestamp` contém o timestamp do evento contido no campo `timestamp` do evento de log. Isso é o equivalente ao `timestamp` campo em InputLogevent.
- `@ingestionTime`contém a hora em que o CloudWatch Logs recebeu o evento de log.
- `@logStream` contém o nome do fluxo de logs ao qual o evento de log foi adicionado. Os fluxos de logs agrupam logs pelo mesmo processo que os gerou.
- `@log` é um identificador de grupo de logs na forma de *`account-id`*`:`*`log-group-name`*. Ao consultar vários grupos de logs, isso pode ser útil para identificar a que grupo de logs um determinado evento pertence.
- `@entity` contém JSON nivelado relacionado a entidades para o recurso de telemetria relacionado ao Explore.

  Por exemplo, esse JSON pode representar uma entidade.

```
{
  "Entity": {
    "KeyAttributes": {
      "Type": "Service",
      "Name": "PetClinic"
    },
```

```
    "Attributes": {
      "PlatformType": "AWS::EC2",
      "EC2.InstanceId": "i-1234567890123"
    }
  }
}
```

Para essa entidade, os campos extraídos do sistema seriam os seguintes:

```
@entity.KeyAttributes.Type = Service
@entity.KeyAttributes.Name = PetClinic
@entity.Attributes.PlatformType = AWS::EC2
@entity.Attributes.EC2.InstanceId = i-1234567890123
```

> **Note**
>
> A descoberta de campos só é compatível com os grupos de logs da classe de logs Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

CloudWatch O Logs Insights insere o símbolo @ no início dos campos que ele gera.

Para muitos tipos de registro, o CloudWatch Logs também descobre automaticamente os campos de registro contidos nos registros. Esses campos de descoberta automática são mostrados na tabela a seguir.

Para outros tipos de registros com campos que o CloudWatch Logs Insights não descobre automaticamente, você pode usar o `parse` comando para extrair e criar campos extraídos para uso nessa consulta. Para obter mais informações, consulte CloudWatch Sintaxe de consulta de linguagem do Logs Insights.

Se o nome de um campo de registro descoberto começar com o `@` caractere, o CloudWatch Logs Insights o exibirá com um adicional `@` anexado ao início. Por exemplo, se um nome de campo de log for `@example.com`, o nome desse campo será exibido como `@@example.com`.

> **Note**
>
> Com exceção de `@message`, `@timestamp` ou `@log`, é possível criar índices de campo para campos descobertos. Para ter mais informações sobre índices de campo, consulte

Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.

Tipo de log	Campos de log descobertos
Logs de fluxo do Amazon VPC	`@timestamp` , `@logStream` , `@message`, `accountId` , `endTime`, `interfaceId` , `logStatus` , `startTime` , `version`, `action`, `bytes`, `dstAddr`, `dstPort`, `packets`, `protocol`, `srcAddr`, `srcPort`
Logs do Route 53	`@timestamp` , `@logStream` , `@message`, `edgeLocation` , `ednsClientSubnet` , `hostZoneId` , `protocol`, `queryName` , `queryTimestamp` , `queryType` , `resolverIp` , `responseCode` , `version`
Logs do Lambda	`@timestamp` , `@logStream` , `@message`, `@requestId` , `@duration`, `@billedDuration` , `@type`, `@maxMemoryUsed` , `@memorySize` Se uma linha de log do Lambda contiver um ID de X-Ray rastreamento, ela também incluirá os seguintes campos: e. `@xrayTraceId` `@xraySegmentId` CloudWatch O Logs Insights descobre automaticamente os campos de log nos registros do Lambda, mas somente para o primeiro fragmento JSON incorporado em cada evento de log. Se um evento de log do Lambda contiver vários fragmentos JSON, será possível analisar e extrair os campos de log usando o comando **`parse`**. Para obter mais informações, consulte Campos em logs JSON.
CloudTrail troncos Logs em formato JSON	Para obter mais informações, consulte Campos em logs JSON.
Outros tipos de log	`@timestamp` , `@ingestionTime` , `@logStream` , `@message`, `@log`.

Campos em logs JSON

Com o CloudWatch Logs Insights, você usa a notação de pontos para representar campos JSON. Esta seção contém um exemplo de evento JSON e trecho de código que mostra como você pode acessar campos JSON usando notação de ponto.

Exemplo: evento 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"
                    }
                }
            ]
        }
    }
}
```

O exemplo de evento JSON contém um objeto chamado `userIdentity`. `userIdentity` contém um campo chamado `type`. Para representar o valor de `type` com notação de ponto, você usa `userIdentity.type`.

O exemplo de evento JSON contém matrizes que se nivelam em listas de nomes e valores de campos aninhados. Para representar o valor de `instanceId` para o primeiro item em `requestParameters.instancesSet`, use `requestParameters.instancesSet.items.0.instanceId`. O número `0` que é colocado antes do campo `instanceID` refere-se à posição dos valores para o campo `items`. O exemplo a seguir contém um trecho de código que mostra como você pode acessar campos JSON aninhados em um evento de log JSON.

Exemplo: consulta

```
fields @timestamp, @message
| filter requestParameters.instancesSet.items.0.instanceId="i-abcde123"
| sort @timestamp desc
```

O trecho de código mostra uma consulta que usa notação de ponto com o comando `filter` para acessar o valor do campo JSON aninhado `instanceId`. A consulta filtra as mensagens para as quais o valor de `instanceId` é igual a `"i-abcde123"` e retorna todos os eventos de log que contêm o valor especificado.

Notação de pontos e campos de JSON-encoded string

A notação de pontos percorre somente os campos que são armazenados como objetos JSON estruturalmente aninhados no momento da ingestão. Se o valor de um campo for uma JSON-encoded string (uma string cujo conteúdo é um JSON válido), a notação de pontos o trata como um valor de folha opaco e não acessa subcampos dentro dele.

Isso geralmente ocorre com:

- Registros transformados por pipelines OCSF (por exemplo, `api.request.data` são digitados como `json_t` no esquema OCSF)
- Qualquer fonte de registro em que cargas de estrutura variável sejam serializadas como cadeias de caracteres antes da ingestão

Por exemplo, se `api.request.data` contiver `{"startTime":1234,"endTime":5678}` como um valor de string, não `api.request.data.startTime` retornará nenhum resultado porque o CloudWatch Logs Insights trata a string inteira como um valor de folha única em vez de um objeto aninhado.

Para acessar subcampos em uma JSON-encoded string, use `jsonParse` para converter a string em um mapa percorrível.

Exemplo: consulta usando JSONParse para um campo de string JSON-encoded

```
fields jsonParse(api.request.data).startTime as startTime
```

Essa consulta é usada `jsonParse` para analisar a JSON-encoded string em um mapa e, `api.request.data` em seguida, acessa o `startTime` subcampo com notação de pontos. Para obter mais informações sobre`jsonParse`, consulte Tipos de estrutura.

> **Note**
>
> CloudWatch O Logs Insights pode extrair no máximo 200 campos de eventos de log de um log JSON. Para os campos adicionais que não são extraídos, você pode usar o comando `parse` para extrair os campos do evento de log não analisado bruto no campo de mensagem. Para obter mais informações sobre o `parse` comando, consulte Sintaxe de consulta no Guia do CloudWatch usuário da Amazon.

Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento

Você pode criar índices de campos em seus eventos de logs para fazer pesquisas eficientes com base na igualdade. Quando você usa um índice de campo em uma consulta do CloudWatch Logs Insights, a consulta tenta ignorar o processamento de eventos de registro que, sabidamente, não incluem o campo indexado. Isso reduz o volume de escaneamento de suas consultas que usam

índices de campo, possibilitando o retorno mais rápido dos resultados. Isso pode ajudar você a pesquisar rapidamente petabytes do total de logs em milhares de grupos de logs e a aprimorar os logs relevantes com mais rapidez. Os campos que você precisa consultar com mais frequência são bons para serem indexados. Campos com alta cardinalidade de valores também são bons candidatos para índices de campo, pois uma consulta usando esses índices de campo será concluída mais rapidamente, já que limita os eventos de logs que estão sendo correspondidos ao valor alvo.

Por exemplo, suponha que você tenha criado um índice de campo para `requestId`. Em seguida, qualquer consulta do CloudWatch Logs Insights nesse grupo de registros que inclua `requestId = value` ou `requestId IN [value, value, ...]` tente processar somente os eventos de registro que são conhecidos por conter esse campo indexado e o valor consultado, e que o CloudWatch Logs tenha detectado um valor para esse campo no passado.

Você também pode usar seus índices de campo para criar consultas eficientes de um número maior de grupos de registros. Quando você usa o comando `filterIndex` em sua consulta em vez do comando `filter`, a consulta é executada em grupos de logs selecionados em eventos de logs que têm índices de campo. Essas consultas podem verificar até 10.000 grupos de logs selecionados especificando até cinco prefixos de nome de grupos de logs. Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você pode escolher todas as contas de origem ou especificar contas de origem individuais para selecionar os grupos de registros.

Os campos indexados diferenciam maiúsculas e minúsculas. Por exemplo, um índice de campo de `RequestId` não corresponderá a um evento de logs que contenha `requestId`.

Os índices de campos são compatíveis somente com os formatos de log estruturados de JSON e logs de serviço.

CloudWatch Os registros fornecem índices de campo padrão para todos os grupos de registros na classe de registros padrão. Os índices de campo padrão ficam disponíveis automaticamente para os seguintes campos:

- `@logStream`
- `@aws.region`
- `@aws.account`
- `@source.log`
- `@data_source_name`
- `@data_source_type`

- `@data_format`
- `traceId`
- `severityText`
- `attributes.session.id`

CloudWatch O Logs também fornece índices de campo padrão para determinadas combinações de nome e tipo de fonte de dados. Os índices de campo padrão estão disponíveis automaticamente para as seguintes combinações de nome e tipo de fonte de dados:

Nome e tipo da fonte de dados	Índices de campo padrão

`amazon_vpc.flow`	`action`
	`logStatus`
	`region`
	`flowDirection`
	`type`

`amazon_route53.resolver_query`	`query_type`
	`transport`
	`rcode`

`aws_waf.access`	`action` `httpRequest.country`

`aws_cloudtrail.data`
`aws_cloudtrail.management`
	`eventSource`
	`eventName`
	`awsRegion`
	`userAgent`
	`errorCode`

Nome e tipo da fonte de dados	Índices de campo padrão

	`eventType`
	
	`managementEvent`
	
	`readOnly`
	
	`eventCategory`
	
	`requestId`

Os índices de campo padrão são adicionais aos índices de campo personalizados definidos em sua política. Os índices de campo padrão não são contabilizados em sua cota de índice de campo.

CloudWatch Os registros indexam somente os eventos de registro ingeridos após a criação de uma política de índice. Ele não indexa eventos de logs ingeridos antes da criação da política. Depois de criar um índice de campo, cada evento de logs correspondente permanece indexado por 30 dias a partir do tempo de ingestão do evento de logs.

> **Note**
>
> Se você criar uma política de índice de campo em uma conta de monitoramento, essa política não será usada para grupos de logs em contas de origem vinculadas. Uma política de índice de campo se aplica somente na conta em que foi criada.

O restante dos tópicos desta seção explica os campos indexados automaticamente e como criar índices de campo. Para obter informações sobre como se referir aos índices de campo em suas consultas, consulte filterIndex e filtrar.

Tópicos

- Campos indexados automaticamente
- Sintaxe e cotas do índice de campo
- Crie uma política de índice de campo ao nível da conta
- Crie uma política de índice de campo em nível de grupo de logs
- Opções de seleção de grupos de logs ao criar uma consulta
- Efeitos da exclusão de uma política de índice de campo

Campos indexados automaticamente

CloudWatch O Logs indexa automaticamente os campos com base em seu uso recente nos filtros de igualdade do CloudWatch Logs Insights que usam os `IN` operadores = ou. Esses índices são adicionais aos índices de campo padrão e aos índices de campo que você configura nas políticas. Você não precisa configurar uma política para campos indexados automaticamente.

CloudWatch Os registros atualizam o conjunto de campos indexados automaticamente com base na atividade de consulta recente. Quando o CloudWatch Logs remove um campo do conjunto, ele para de indexar eventos recém-ingeridos para esse campo. CloudWatch O Logs gerencia campos indexados automaticamente e os retém por 30 dias. Para manter um campo indexado permanentemente, adicione-o a uma política de índice de campo em nível de conta ou de grupo de registros.

⚠ Use filtro em vez de filterIndex para campos indexados automaticamente

Recomendamos que você use `filter` em vez de `filterIndex` com campos indexados automaticamente. CloudWatch Os registros podem atualizar ou remover esses campos com base nos padrões de consulta. CloudWatch Os registros os retêm por apenas 30 dias. O `filterIndex` comando retorna somente dados indexados. Como o CloudWatch Logs atualiza o conjunto com base na atividade da consulta, uma `filterIndex` consulta não pesquisa eventos ingeridos antes de o CloudWatch Logs selecionar o campo ou depois que o CloudWatch Logs parou de indexá-lo. Se você usar`filterIndex`, adicione o campo a uma política de índice de campo para mantê-lo indexado, independentemente da atividade de consulta. Para obter mais informações, consulte filterIndex comparado ao filtro.

A guia Índices de campos de um grupo de registros lista campos indexados automaticamente.

Você também pode usar a `DescribeFieldIndexes`operação para listar campos indexados automaticamente. Inclua `AUTO` no parâmetro da `indexCategories` solicitação. A resposta é definida `indexCategory` como `AUTO` quatro campos indexados automaticamente. Solicitações que são `indexCategories` omitidas não retornam campos indexados automaticamente.

Para adicionar um campo indexado automaticamente a uma política de índice em nível de grupo de registros

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.

2. No painel de navegação esquerdo, selecione Logs, Log groups (Grupos de log).
3. Escolha o nome do grupo de logs.
4. Escolha a guia Índices de campo e, em seguida, escolha Gerenciar índices de campo.
5. Na página Gerenciar índices de campo em nível de grupo de registros, revise os campos na política. Adicione ou remova campos conforme necessário e escolha Salvar.

Sintaxe e cotas do índice de campo

É possível gerar índices de campo criando políticas de índice de campo. Você pode criar políticas de índice em nível de conta que se aplicam a toda a sua conta, bem como políticas que se aplicam somente a um único grupo de logs. Para políticas de indexação em toda a conta, você pode ter uma que se aplique a todos os grupos de logs da conta. Também é possível criar políticas de índice em nível de conta que se apliquem a um subconjunto de grupos de logs na conta, selecionados pelos prefixos dos nomes dos grupos de logs. Se você tiver várias políticas em nível de conta na mesma conta, os prefixos do nome do grupo de logs para essas políticas não podem se sobrepor. Da mesma forma, você pode criar políticas de índice em nível de conta que se apliquem a uma combinação específica de nome e tipo de fonte de dados. Somente uma política de conta pode ser criada por combinação de nome e tipo de fonte de dados.

As políticas de índice de campo em nível de grupo de registros substituem as políticas de índice de campo em nível de conta: que se aplicam ao grupo de registros como um todo (como políticas em nível de conta sem critérios de seleção ou com critérios de seleção baseados no prefixo do nome do grupo de registros). Account-level políticas que correspondam ao nível do evento de log (como, por exemplo, para uma determinada combinação de nome e tipo de fonte de dados) serão aplicadas, além das políticas que correspondam ao grupo de log como um todo. Se você criar uma política de índice em nível de grupo de registros, esse grupo de registros não usará políticas em nível de conta que correspondam ao nível do grupo de registros.

As correspondências de eventos de logs com os nomes dos índices de campo diferenciam maiúsculas de minúsculas. Por exemplo, um índice de campo de `RequestId` não corresponderá a um evento de logs que contenha `requestId`.

Você pode ter até 40 políticas de índice em nível de conta. Dessas políticas, 20 podem usar critérios de seleção de prefixo de nome de grupo de log e 20 podem usar critérios de seleção baseados em fonte de dados. Se você tiver várias políticas de índice em nível de conta filtradas para prefixos de nome de grupo de logs, nenhuma delas poderá usar prefixos de nome de grupo de logs iguais ou sobrepostos. Por exemplo, se você tiver uma política filtrada para grupos de logs que começam

com `my-log`, não pode ter outra política de índice de campo filtrada para `my-logpprod` ou `my-logging`. Da mesma forma, se você tiver várias políticas de índice em nível de conta filtradas para combinações de nome e tipo de fonte de dados, nenhuma delas poderá usar o mesmo nome e tipo de fonte de dados. Por exemplo, se você tiver uma política filtrada pelo nome `amazon_vpc` e tipo da fonte de dados, não `flow` poderá criar outra política com essa combinação.

Se você tiver uma política de índice em nível de conta que não tenha prefixos de nome e se aplique a todos os grupos de registros, nenhuma outra política de índice em nível de conta com filtros de prefixo de nome de grupo de registros poderá ser criada; você pode criar políticas de índice em nível de conta que usem filtros de nome e tipo de fonte de dados.

Cada política de indexação tem as seguintes cotas e restrições:

- Podem ser incluídos até 20 campos na política.
- Cada nome de campo pode incluir até 100 caracteres.
- Para criar um índice de um campo personalizado em seus grupos de logs que comece com `@`, você deve especificar o campo com um `@` extra no início do nome do campo. Por exemplo, se seus eventos de logs incluírem um campo chamado `@userId`, você deverá especificar `@@userId` para criar um índice para esse campo.

Para políticas de índice em nível de conta com critérios de seleção baseados em nome de fonte de dados e tipo, uma restrição adicional se aplica: todos os campos devem ser tipos de dados primitivos, primitivos aninhados são compatíveis somente com estruturas.

Campos gerados e campos reservados

CloudWatch O Logs Insights gera automaticamente campos do sistema em cada evento de registro. Esses campos gerados são prefixados com `@`. Para obter mais informações sobre os campos gerados, consulte Logs compatíveis e campos descobertos.

Os seguintes campos gerados são compatíveis para uso como índices de campo:

- `@logStream`
- `@ingestionTime`
- `@requestId`
- `@type`
- `@initDuration`
- `@duration`

- `@billedDuration`
- `@memorySize`
- `@maxMemoryUsed`
- `@xrayTraceId`
- `@xraySegmentId`

Para indexar esses campos gerados, você não precisa adicionar um `@` extra ao especificá-los, como é necessário fazer com campos personalizados que começam com `@`. Por exemplo, para criar um índice de campo para `@logStream`, basta especificar `@logStream` como índice de campo.

CloudWatch Os registros fornecem índices de campo padrão para todos os grupos de registros na classe de registros padrão. Os índices de campo padrão ficam disponíveis automaticamente para os seguintes campos:

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

CloudWatch O Logs também fornece índices de campo padrão para determinadas combinações de nome e tipo de fonte de dados. Os índices de campo padrão estão disponíveis automaticamente para as seguintes combinações de nome e tipo de fonte de dados:

Nome e tipo da fonte de dados	Índices de campo padrão
`amazon_vpc.flow`	`action` `logStatus`

Nome e tipo da fonte de dados	Índices de campo padrão
	region flowDirection type
amazon_route53.resolver_query	query_type transport rcode
aws_waf.access	action httpRequest.country
aws_cloudtrail.data aws_cloudtrail.management	eventSource eventName awsRegion userAgent errorCode eventType managementEvent readOnly eventCategory requestId

Os índices de campo padrão são adicionais aos índices de campo personalizados definidos em sua política. Os índices de campo padrão não são contabilizados em sua cota de índice de campo.

Campos secundários e campos de matriz em logs JSON

Você pode indexar campos que são secundários aninhados ou de matriz em logs JSON.

Por exemplo, você pode criar um índice do campo filho `accessKeyId` dentro do campo `userIdentity` desse log:

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

Para criar esse campo, você se refere a ele usando a notação de ponto (`userIdentity.accessKeyId`) ao criar o índice do campo e ao especificá-lo em uma consulta. A consulta pode ser como esta:

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

```
| filterIndex userIdentity.accessKeyId = "11112222"
```

No evento de exemplo anterior, o campo `instanceId` está em uma matriz em `requestParameters.instancesSet.items`. Para representar esse campo ao criar o índice de campos e para consultar, indique-o como `requestParameters.instancesSet.items.0.instanceId`. O 0 se refere ao lugar desse campo no array.

Em seguida, uma consulta para esse campo pode ser:

```
fields @timestamp, @message
| filterIndex requestParameters.instancesSet.items.0.instanceId="i-abcde123"
```

Crie uma política de índice de campo ao nível da conta

Use as etapas desta seção para criar uma política de índice de campo que se aplique a todos ou vários grupos de logs na conta que tenham nomes começados com a mesma string.

Para criar uma política de índice de campo ao nível da conta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação à esquerda, escolha Configurações e, em seguida, escolha Logs.
3. Na seção Políticas de índice em nível de conta, escolha Gerenciar.
4. Escolha Criar política de índice.
5. Em Nome da política, insira um nome para sua nova política.
6. Em Selecionar escopo da política, faça o seguinte:
   - Escolha Todos os grupos de logs padrão para que a política de índice se aplique a todos os grupos de logs da Classe Padrão na conta.
   - Escolha Grupos de registros por correspondência de prefixo para aplicar a política a um subconjunto de grupos de registros, todos com nomes que começam com a mesma string. Em seguida, insira o prefixo desses grupos de logs em Inserir um nome de prefixo.

     Depois de inserir seu prefixo, você pode escolher Visualizar grupos de logs correspondentes ao prefixo para confirmar se seu prefixo corresponde aos grupos que você esperava.

Escolha Dados de registros por fonte de dados para aplicar a política a uma combinação específica de nome e tipo de fonte de dados. Em seguida, você pode selecionar a Fonte de dados e o Tipo de dados no menu suspenso.

Depois de selecionar o nome e o tipo da fonte de dados, você pode selecionar Obter campos para preencher a seção Configurar índices e facetas de campo com informações relevantes, como os campos disponíveis, os grupos de registros incluídos, bem como os índices de campo padrão e personalizados.

7. Em Configuração personalizada do campo de índice, selecione Adicionar caminho de campo para inserir o primeiro campo a ser indexado.

   Em seguida, insira a string a ser usada como valor do nome do campo ou selecione um campo no menu suspenso. Ela deve ter letras maiúsculas e minúsculas que correspondam exatamente ao que aparece nos eventos de logs. Por exemplo, se seus eventos de logs incluírem `requestId`, você deverá inserir `requestId` aqui. `RequestId`, `requestID` e `request Id` não corresponderão.

   Se você quiser indexar um campo de log personalizado que comece com o caractere `@`, inclua um caractere extra `@` ao inserir a string de índice. Por exemplo, se você tiver um campo de log personalizado `@emailname`, insira `@@emailname` na caixa Adicionar caminho do campo.

   Você também pode criar índices para os `@logStream` campos `@ingestionTime` e que o CloudWatch Logs gera automaticamente. Se você fizer isso, não será necessário adicionar um `@` extra ao especificá-los.
8. (Opcional) Além de especificar o caminho do campo, você pode selecionar Definir como uma faceta para criar o campo como uma faceta.
9. Repita a etapa anterior para adicionar até 20 índices de campo.
10. Quando terminar, selecione Create (Criar).

Crie uma política de índice de campo em nível de grupo de logs

Use as etapas desta seção para criar uma política de índice de campo que se aplique a um único grupo de logs.

Para criar uma política de índice de campo em nível de grupo de logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.

2. No painel de navegação esquerdo, selecione Logs, Log groups (Grupos de log).
3. Escolha o nome do grupo de logs.
4. Escolha a guia Índices de campo.
5. Escolha Gerenciar índices de campo para esse grupo de logs.
6. Em Gerenciar índices de campo em nível de grupo de logs, escolha Adicionar caminho de campo para inserir o primeiro campo a ser indexado.

   Em seguida, insira a string a ser usada como valor do nome do campo. Ela deve ter letras maiúsculas e minúsculas que correspondam exatamente ao que aparece nos eventos de logs. Por exemplo, se seus eventos de logs incluírem `requestId`, você deverá inserir `requestId` aqui. `RequestId`, `requestID` e `request Id` não corresponderão.

   Se você quiser indexar um campo de log personalizado que comece com o caractere `@`, inclua um caractere extra `@` ao inserir a string de índice. Por exemplo, se você tiver um campo de log personalizado `@emailname`, insira `@@emailname` na caixa Adicionar caminho do campo.

   Você também pode criar índices para os `@logStream` campos `@ingestionTime` e que o CloudWatch Logs gera automaticamente. Se você fizer isso, não será necessário adicionar um `@` extra ao especificá-los.
7. (Opcional) Além de especificar o caminho do campo, você pode selecionar Definir como uma faceta para criar o campo como uma faceta.
8. Repita a etapa anterior para adicionar até 20 índices de campo.
9. Quando terminar, escolha Salvar.

Opções de seleção de grupos de logs ao criar uma consulta

Esta seção explica as várias maneiras pelas quais você pode selecionar grupos de logs a serem incluídos em uma consulta.

Para selecionar grupos de logs para uma consulta no console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Logs Insights.
3. Selecione a linguagem de consulta que deseja usar para esta consulta. Você pode escolher entre: Logs Insights QL, OpenSearchPPL ou OpenSearch SQL.
4. Há três maneiras de selecionar grupos de logs para a consulta:

- Use a caixa Nome do grupo de logs. Este é o método de seleção padrão. Você pode inserir até 50 nomes de grupos de logs com esse método. Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você poderá selecionar grupos de registros nas contas de origem e na conta de monitoramento. Uma única consulta pode consultar logs de diferentes contas ao mesmo tempo.
- Use a seção Critérios do grupo de logs. Nesta seção, você pode escolher grupos de logs com base no prefixo dos nomes dos grupos de logs. É possível incluir até cinco prefixos em uma consulta. Serão selecionados grupos de logs que tenham esses prefixos nos nomes. Como alternativa, a opção Todos os grupos de logs seleciona todos os grupos da conta.
- Se for uma conta de monitoramento na observabilidade CloudWatch entre contas, você pode selecionar Todas as contas no menu suspenso de contas para selecionar os grupos de registros de todas as contas vinculadas. Como alternativa, você pode selecionar individualmente quais contas devem ser incluídas nessa consulta.

Se suas opções corresponderem a mais de 10.000 grupos de logs, será exibido um erro solicitando que você restrinja sua seleção.

5. A classe de log padrão para uma consulta é Padrão. Você pode usar a classe Log para alterá-la para Acesso infrequente.

Usando o AWS CLI

Para fazer esses tipos de seleções ao iniciar uma consulta na linha de comando, você pode usar o comando `source` na sua consulta. Para ter mais informações e exemplos, consulte SOURCE.

Efeitos da exclusão de uma política de índice de campo

Se você excluir uma política de índice de campo que está em vigor há algum tempo, pode acontecer o seguinte:

- Por até 30 dias após a exclusão da política, as consultas ainda podem se beneficiar dos eventos de logs indexados.
- Se você excluir uma política de índice em nível de grupo de logs e já existir uma política em nível de conta que se aplicaria a esse grupo, a política em nível de conta acabará se aplicando a ele.

Use facetas para agrupar e explorar registros

As facetas são úteis para analisar registros, pois permitem filtrar e agrupar seus dados de forma interativa sem executar consultas. Uma faceta é um campo em seus registros (como `ServiceName` ou`StatusCode`) que permite filtragem, agregação e análise em grupos de registros. Você pode ver a lista de campos facetados no console do CloudWatch Logs Insights, junto com a contagem de eventos de registro para cada valor de faceta com base no intervalo de tempo selecionado. Conforme você seleciona diferentes facetas e valores, os valores e contagens das facetas são atualizados em tempo real, permitindo que você explore seus registros de forma interativa.

Cada faceta mostra valores e contagens disponíveis extraídos automaticamente dos campos em seus registros com base no intervalo de tempo e no escopo da consulta selecionados, e retidos por 30 dias. As contagens de facetas mostradas são aproximadas. Você pode usar as facetas padrão, como nome da fonte de dados ou tipo de fonte de dados, para explorar seus registros ou criar facetas personalizadas em qualquer um dos campos em seus registros. O nome da fonte de dados é um serviço ou aplicativo da AWS que gera os registros (por exemplo, Route 53, Amazon VPC ou CloudTrail) e o tipo de fonte de dados é o tipo específico de log gerado por esse serviço. As facetas padrão são criadas por CloudWatch e incluem `@aws.region`@`data_source_name`,`@data_source_type`, e. `@data_format` Para obter mais informações, consulte Gerenciamento de logs. As facetas só estão disponíveis para registros que são ingeridos na conta. Se você configurou a observabilidade entre contas, a conta de monitoramento não pode visualizar facetas com base nos registros das contas de origem.

Se você tiver as tags de recursos para telemetria ativadas, os campos da tag de recurso (`@aws.tag.*`) serão adicionados automaticamente como facetas padrão. Essas facetas permitem filtrar e agrupar eventos de log pelas tags dos recursos que os produziram.

Para criar facetas adicionais, selecione os campos em seus registros que são relevantes para sua solução de problemas e configure-os usando as políticas de índice. Para facetas personalizadas, recomendamos criá-las em campos de baixa cardinalidade (campos com menos de 100 valores exclusivos por dia, como Status e). ApplicationName Facetas com mais de 100 valores exclusivos por dia são classificadas como de alta cardinalidade e os valores dessas facetas não são exibidos. Selecione uma ou mais facetas e opte por executar consultas em seus registros.

Para começar a usar facetas no CloudWatch Logs Insights:

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Logs Insights.

3. (Opcional) Use o seletor de intervalo de tempo para selecionar um período que você deseja analisar. Para o intervalo de tempo selecionado, as facetas e valores disponíveis são mostrados no painel.
4. Selecione facetas para explorar seus dados e ver atualizações em tempo real das distribuições de valor entre as facetas.

   Facetas com mais de 100 valores exclusivos não são exibidas. Para consultar valores específicos, use filtros em sua consulta em vez disso.

Para executar uma consulta baseada em facetas

1. Selecione um ou mais valores em todas as facetas.
2. A contagem de eventos será atualizada com base nas facetas e valores selecionados.
3. Conforme os valores das facetas são selecionados, o escopo da consulta é atualizado para refletir a seleção.
4. Depois de selecionar os valores das facetas, escolha executar para executar sua consulta.
5. O número máximo de valores exclusivos suportados por faceta é 100. Por exemplo, se houver mais de 100 valores para uma faceta, todas as contagens serão exibidas como “-”, indicando que os valores são desconhecidos.

Para salvar uma consulta baseada em facetas

1. Crie sua consulta usando um ou mais valores de faceta.
2. O restante das etapas é o mesmo que salvar uma consulta do Logs Insights. Consulte Salvando consultas do CloudWatch Logs Insights.
3. Suas consultas salvas estão disponíveis na seção Consultas salvas. Quando você recupera uma consulta salva, ela inclui automaticamente as facetas e os valores usados para a consulta, facilitando a análise de seus registros.

Para criar uma faceta no nível da conta

1. Para criar facetas, você precisa primeiro criar o campo como um índice e configurá-lo como uma faceta. No painel de navegação, escolha Configurações, Registros, Políticas de índice em nível de conta. Como alternativa, você pode selecionar Gerenciar facetas no painel de facetas.

2. Escolha Criar nova política de índice. Para obter detalhes sobre a criação de políticas de índice, consulteCrie uma política de índice de campo ao nível da conta.
3. Para criar uma faceta, marque Definir como faceta para o campo selecionado na página de criação da política de índice.

Gerenciamento de facetas usando APIs

O gerenciamento de facetas pode ser feito usando a política de índice de campo. Consulte as `field index`APIs para obter detalhes.

APIs de índice de campo

Não.	Name (Nome)	Description
1	PutIndexPolicy	Cria ou atualiza uma política de índice de campo para o grupo de registros específico
2	PutAccountPolicy	Cria uma política de proteção de dados em nível de conta, política de filtro de assinatura, política de índice de campo, política de transformador ou política de extração métrica que se aplica a todos os grupos de registros ou a um subconjunto de grupos de registros na conta
3	DeleteIndexPolicy	Exclui uma política de índice de campo em nível de grupo de registros que foi aplicada a um único grupo de registros
4	DeleteAccountPolicy	Exclui uma política de conta do CloudWatch Logs

Visualizando registros ao redor no CloudWatch Logs Insights

Você pode visualizar eventos de registro que ocorreram antes e depois de um registro de registro específico retornado pela sua consulta do CloudWatch Logs Insights. Os registros circundantes ajudam você a entender o contexto de erros, avisos ou outros eventos significativos.

Como funcionam os troncos circundantes

Quando você executa uma consulta do CloudWatch Logs Insights, cada registro de registro nos resultados inclui uma opção de registros circundantes. Quando você escolhe essa opção, o CloudWatch Logs Insights exibe eventos de registro adicionais do mesmo fluxo de registros que ocorreram imediatamente antes e depois do registro selecionado. Esses eventos aparecem mesmo que não correspondam aos filtros de consulta originais.

Você pode configurar o número de linhas de registro ao redor a serem exibidas. Escolha entre 5, 10, 20, 50 ou 100 linhas de registro acima e abaixo do registro selecionado.

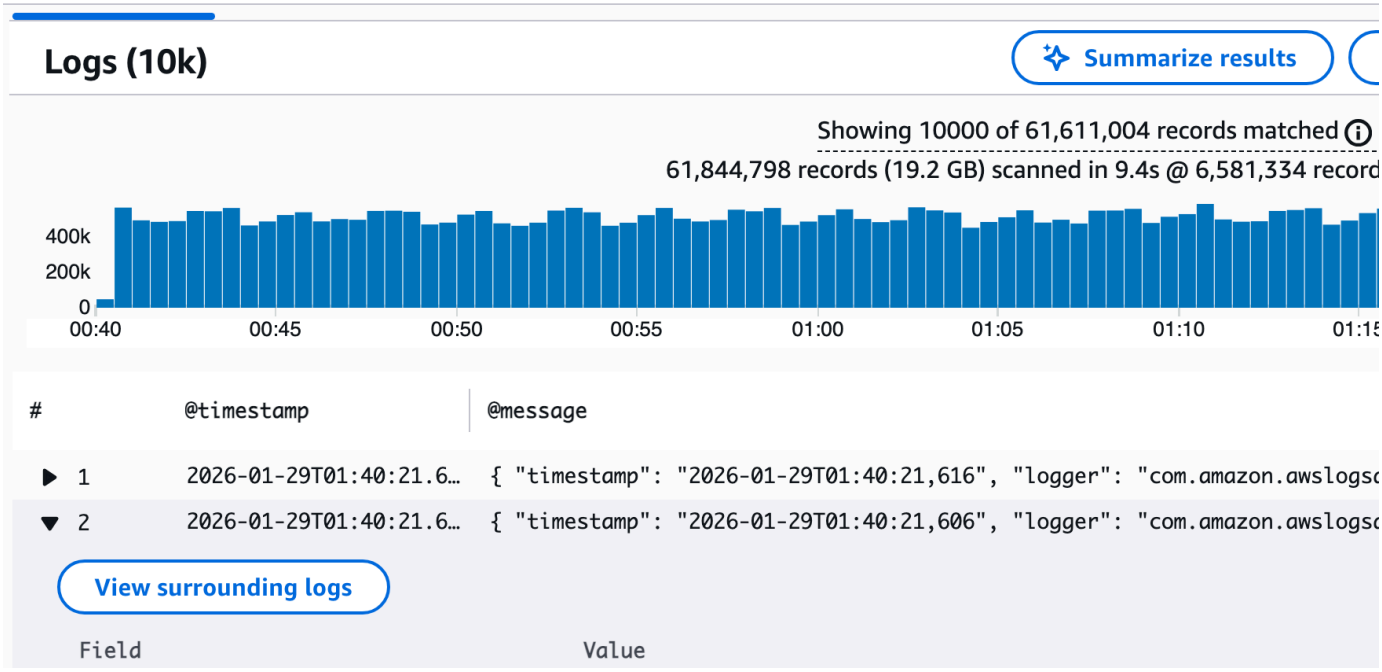
Os registros circundantes são úteis nos seguintes cenários:

- Depurando erros ao visualizar os eventos que levaram a uma falha
- Investigando avisos ou tempos limite em sistemas distribuídos
- Entendendo a sequência de eventos em seu aplicativo
- Analisando problemas em ambientes de registro de alto volume

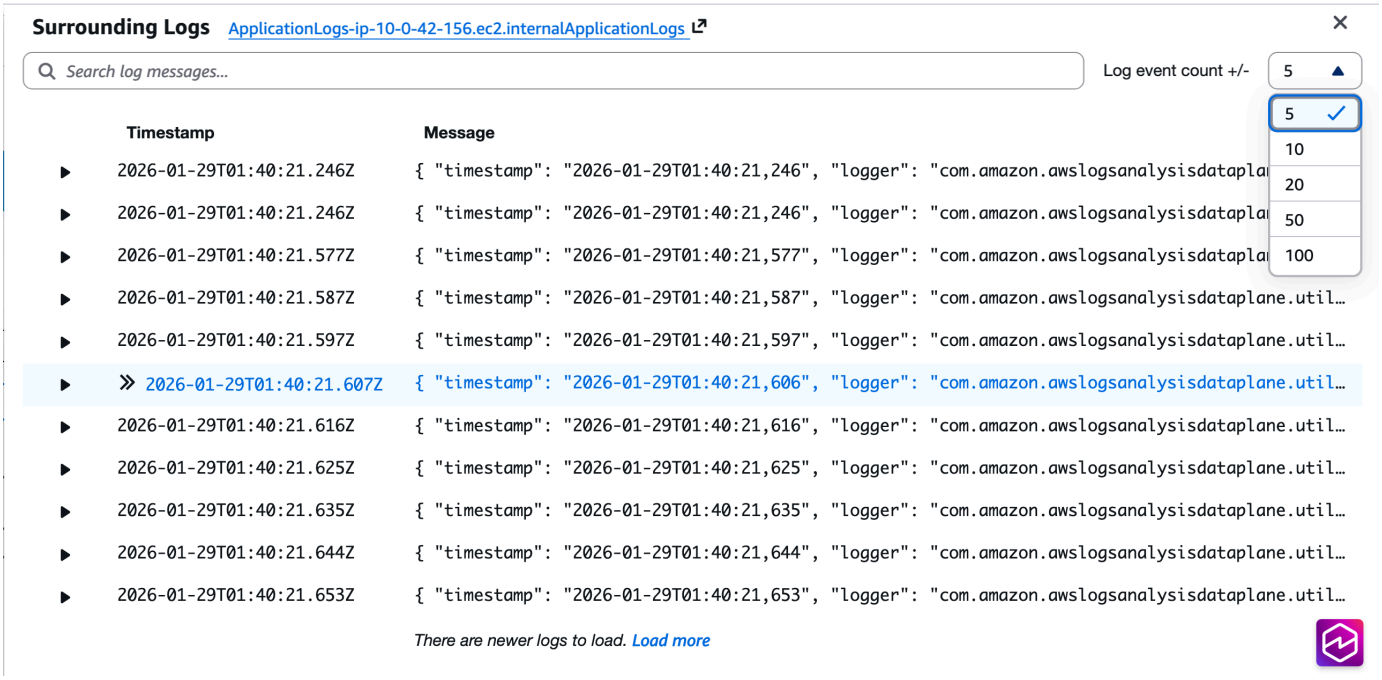
Veja os registros ao redor

Para ver os registros ao redor

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Logs Insights.
3. Execute a consulta.
4. Nos resultados da consulta, localize o registro de log que você deseja investigar.
5. Escolha Registros circundantes.



6. No menu suspenso Número de eventos, escolha o número de eventos de registro a serem exibidos acima e abaixo do registro selecionado. Você pode escolher 5, 10, 20, 50 ou 100.

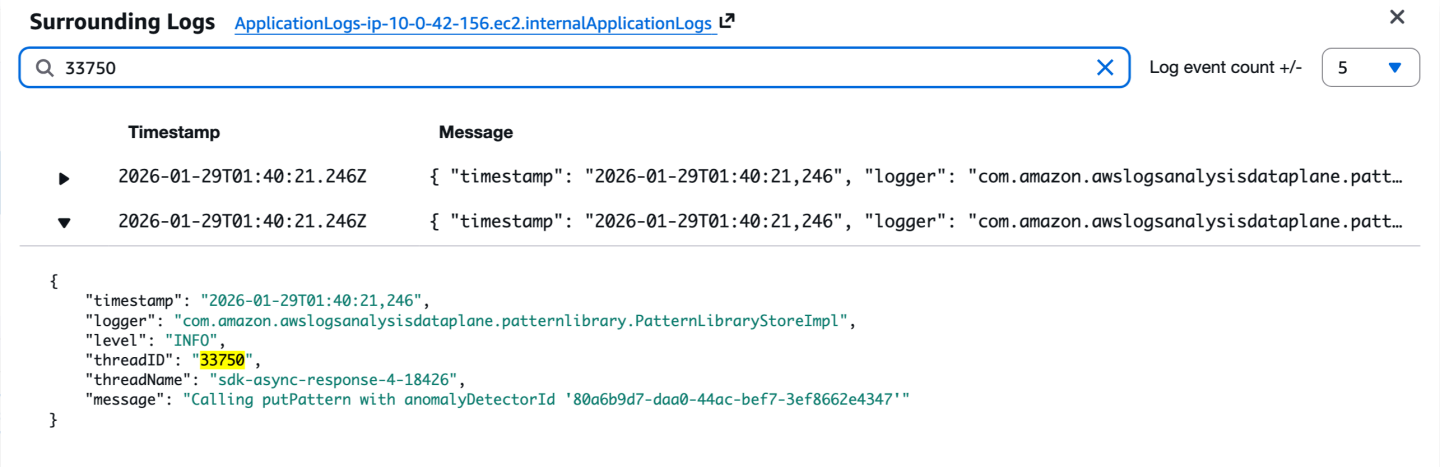


Pesquise nos registros ao redor

Depois de abrir o painel de registros ao redor, você pode pesquisar palavras-chave específicas para localizar o contexto relevante.

Para pesquisar nos registros circundantes

1. No painel de registros ao redor, insira seu termo de pesquisa na caixa de pesquisa.
2. Revise as linhas de registro correspondentes destacadas.
3. Use os controles de navegação para se mover entre as partidas.



Análise de padrões

CloudWatch O Logs Insights usa algoritmos de aprendizado de máquina para encontrar padrões quando você consulta seus registros. Um padrão é uma estrutura de texto compartilhada que se repete nos campos do log. Ao visualizar os resultados de uma consulta, você pode escolher a guia Padrões para ver os padrões que o CloudWatch Logs encontrou com base em uma amostra dos seus resultados. Ou então, você pode acrescentar o comando `pattern` ao final da consulta para analisar os padrões em todo o conjunto de eventos de log correspondentes.

Os padrões são úteis para analisar grandes conjuntos de logs porque um grande número de eventos de log geralmente pode ser resumido a apenas alguns padrões.

Considere o exemplo dos três eventos de log a seguir.

```
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
```

No exemplo anterior, todos os três eventos de log seguem um único padrão:

```
<Time-1> [INFO] Calling DynamoDB to store for resource id <ID-2>
```

Os campos dentro de um padrão são chamados de tokens. Os campos que variam dentro de um padrão, como um ID de solicitação ou um timestamp, são tokens dinâmicos. Cada token dinâmico é representado por <*string*-*number*>. *string*É uma descrição do tipo de dados que o token representa. *number*Mostra onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

Exemplos comuns de tokens dinâmicos incluem códigos de erro, timestamps e IDs de solicitação. Um valor de token representa um determinado valor de um token dinâmico. Por exemplo, se um token dinâmico representasse um código de erro HTTP, o valor de um token poderia ser `501`.

A detecção de padrões também é usada no detector de anomalias CloudWatch Logs e nos recursos de comparação. Para obter mais informações, consulte Detecção de anomalias de log e Compare (diff) com intervalos de tempo anteriores.

Introdução à análise de padrões

A detecção de padrões é realizada automaticamente em qualquer consulta do CloudWatch Logs Insights. Os resultados das consultas que não incluem o comando `pattern` apresentam tanto eventos quando padrões de log.

Se você incluir o comando `pattern` em sua consulta, a análise de padrões será realizada em todo o conjunto de eventos de log correspondente. Isso fornece resultados de padrões mais exatos, mas os eventos de log brutos não são retornados quando você usa o comando `pattern`. Quando uma consulta não inclui `pattern`, os resultados de padrões são baseados nos primeiros 1.000 eventos de log retornados ou no valor limite usado na consulta. Se você incluir `pattern` na consulta, os resultados exibidos na guia Padrões serão extraídos de todos os eventos de log que corresponderem à consulta.

Para começar com a análise de padrões no CloudWatch Logs Insights

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Logs Insights.

   Na página Logs Insights (Insights de log), o editor de consultas contém uma consulta padrão que retorna os 20 eventos de log mais recentes.
3. Remova a linha `| limit 20` da caixa de consulta para que a consulta fique assim:

```
fields @timestamp, @message, @logStream, @log
| sort @timestamp desc
```

4. No menu suspenso Selecionar grupos de logs, escolha um ou mais grupos de logs para consultar.
5. (Opcional) Use o seletor de tempo para selecionar o período que você deseja consultar.

   Você pode escolher intervalos de 5 minutos, 30 minutos, 1 hora, 3 horas, 12 horas ou um período personalizado.
6. Escolha Executar consulta para iniciar a consulta.

   Quando a consulta acaba de ser executada, a guia Logs exibe uma tabela dos eventos de log retornados pela consulta. Acima da tabela, vem uma mensagem sobre quantos registros corresponderam à consulta, como, por exemplo, Mostrando 10.000 dos 71.101 registros encontrados.
7. Escolha a guia Padrões.
8. A tabela agora exibe os padrões encontrados na consulta. Como a consulta não incluía o comando `pattern`, essa guia só exibe os padrões descobertos entre os 10.000 eventos de log que foram mostrados na tabela na guia Logs.

   Para cada padrão, as seguintes informações são exibidas:

   - O padrão, com cada token dinâmico exibido como <*string*-*number*>. *string*É uma descrição do tipo de dados que o token representa. *number*Mostra onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.
   - O número de eventos, que é o número de vezes que o padrão apareceu nos eventos de log consultados. Escolha o cabeçalho de coluna Número de eventos para classificar os padrões por frequência.
   - A Taxa de eventos, que é a porcentagem dos eventos de log consultados que contém esse padrão.
   - O Tipo de gravidade, que será um dos seguintes:
     - ERRO se o padrão contiver a palavra Erro.
     - AVISO se o padrão contiver a palavra Aviso, mas não contiver Erro.
     - INFORMAÇÕES se o padrão não contiver nem aviso nem erro.

Escolha o cabeçalho de coluna Informações sobre a gravidade para classificar os padrões por gravidade.

9. Agora altere a consulta. Substitua a linha `| sort @timestamp desc` da consulta por `| pattern @message`, de modo que a consulta completa fique assim:

```
fields @timestamp, @message, @logStream, @log
| pattern @message
```

10. Selecione Executar consulta.

    Quando a consulta termina, não há resultados na guia Logs. Porém, a guia Padrões provavelmente tem um número maior de padrões listados, dependendo do número total de eventos de log que foram consultados.

11. Independentemente de ter incluído `pattern` na sua consulta, você pode inspecionar melhor os padrões que a consulta retorna. Para fazer isso, escolha o ícone na coluna Inspecionar para um dos padrões.

    O painel Inspeção de padrões aparece e exibe o seguinte:

    - O padrão. Selecione um token no padrão para analisar os valores desse token.
    - Um histograma mostrando o número de ocorrências do padrão no intervalo consultado. Isso pode ajudar a identificar tendências interessantes, como um aumento repentino na ocorrência de um padrão.
    - A guia Exemplos de log exibe alguns eventos de log que correspondem ao padrão selecionado.
    - A guia Valores do token exibe os valores do token dinâmico selecionado, se você tiver selecionado um.

      > **Note**
      >
      > No máximo de 10 valores de token são capturados para cada token. A contagem de tokens pode não ser precisa. CloudWatch O Logs usa um contador probabilístico para gerar a contagem de tokens, não o valor absoluto.

    - A guia Padrões relacionados exibe outros padrões que ocorreram com frequência quase ao mesmo tempo que o padrão que você está inspecionando. Por exemplo, se um padrão para

uma mensagem `ERROR` geralmente foi acompanhado por outro evento de log marcado como `INFO` com detalhes adicionais, esse padrão será exibido aqui.

Detalhes sobre o comando de padrão

Esta seção contém mais detalhes sobre o comando `pattern` e seus usos.

- No tutorial anterior, removemos o comando `sort` quando o adicionamos `pattern` porque uma consulta não será válida se incluir um comando `pattern` após um comando `sort`. É válido ter um `pattern` antes de um `sort`.

  Para obter detalhes sobre a sintaxe de `pattern`, consulte pattern.
- Quando você usa `pattern` em uma consulta, `@message` deve ser um dos campos selecionados no comando `pattern`.
- Você pode incluir o comando `filter` antes de um comando `pattern` para que apenas o conjunto de eventos de log filtrado seja usado como entrada na análise de padrões.
- Para ver os resultados de padrões para um campo específico, como um campo obtido com o comando `parse`, use `pattern @fieldname`.
- Consultas com saídas que não são logs, como consultas com o comando `stats`, não retornam resultados de padrões.

Salve e execute novamente as consultas do CloudWatch Logs Insights

Depois de criar uma consulta, você pode salvá-la para que possa ser executada novamente mais tarde. As consultas são salvas em uma estrutura de pastas para que você possa organizá-las. Você pode salvar até 1.000 consultas por região e por conta.

As consultas são salvas em um Region-specific nível, não em um nível específico do usuário. Se você criar e salvar uma consulta, outros usuários com acesso aos CloudWatch registros na mesma região poderão ver todas as consultas salvas e suas estruturas de pastas na região.

Para salvar uma consulta, você deve estar conectado a uma função que tenha a permissão `logs:PutQueryDefinition`. Para ver uma lista de consultas salvas, você deve estar conectado a uma função que tenha a permissão`logs:DescribeQueryDefinitions`.

> **Note**
>
> Você pode criar e salvar consultas com parâmetros — modelos reutilizáveis com espaços reservados nomeados. Em vez de salvar várias variações da mesma consulta com valores diferentes, crie um modelo e forneça valores de parâmetros diferentes ao executá-lo. Atualmente, essa funcionalidade é compatível com consultas usando somente a linguagem de consulta do Logs Insights. Para obter mais informações, consulte Usando consultas salvas com parâmetros.

Console

Para salvar uma consulta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. No editor de consultas, crie uma consulta.
4. Escolha Salvar.
5. Insira um nome para a consulta.
6. (Opcional) Escolha uma pasta na qual deseja salvar a consulta. Selecione Criar novo para criar uma pasta. Se você criar uma pasta, poderá usar caracteres de barra (/) no nome da pasta para definir uma estrutura de pasta. Por exemplo, dar o nome **`folder-level-1/folder-level-2`** a uma nova pasta cria uma pasta de nível superior chamada **`folder-level-1`**, com outra pasta chamada **`folder-level-2`** dentro dela. A consulta é salva em **`folder-level-2`**.
7. (Opcional) Altere os grupos de log da consulta ou o texto da consulta.
8. (Opcional) Para usar parâmetros em sua consulta, siga estas etapas adicionais:
   a. Adicione parâmetros à sua consulta. Substitua valores estáticos por espaços reservados usando a `{{parameter}}` sintaxe (chaves duplas antes e depois do nome do parâmetro).

      Exemplo: consulta original com valores estáticos:

```
fields @timestamp, @message
| filter level = "Error"
| filter applicationName = "OrderService"
```

Consulta atualizada com parâmetros:

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

b. Defina os parâmetros usados na sua consulta. Para cada parâmetro de espaço reservado, especifique:

- Nome: deve corresponder exatamente ao nome do espaço reservado (por exemplo,`logLevel`,`applicationName`).
- Valor padrão (opcional): o valor a ser usado se nenhum valor de parâmetro for fornecido.
- Descrição (opcional): explica a finalidade do parâmetro.

c. As consultas com parâmetros podem ser executadas usando o nome da consulta com um $ prefixo e passando os nomes dos parâmetros como pares de valores-chave. Consulte Para executar uma consulta salva para obter detalhes.

9. Escolha Salvar.

AWS CLI

Para salvar uma consulta, use`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
```

(Opcional) Para salvar uma consulta com parâmetros, adicione a `--parameters` opção e use `{{parameterName}}` espaços reservados na sequência de caracteres de consulta:

```
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
```

Para salvar uma consulta em uma pasta, prefixe o nome da consulta com o caminho da pasta:

```
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

Para salvar uma consulta, chame PutQueryDefinition:

```
{
  "name": "ErrorsByLevel",
  "queryString": "fields @timestamp, @message | filter level = \"ERROR\"",
  "logGroupNames": ["/aws/lambda/my-function"]
}
```

(Opcional) Para salvar uma consulta com parâmetros, inclua o `parameters` campo e use `{{parameterName}}` espaços reservados na sequência de caracteres de consulta:

```
{
  "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**
>
> Você pode criar uma pasta para consultas salvas usando `PutQueryDefinition`. Para criar uma pasta para as consultas salvas, use uma barra (/) para prefixar o nome da consulta desejada com o nome da pasta desejada: <*folder-name*>/<*query-name*>. Para obter mais informações sobre a ação, consulte PutQueryDefinition.

Console

Como executar uma consulta salva

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. À direita, escolha Consultas.
4. Selecione sua consulta na lista Consultas salvas. O texto da consulta aparece no editor de consultas.
5. (Opcional) Para usar uma consulta com parâmetros:
   a. Escolha o ícone + ao lado do nome da consulta no painel lateral Consultas salvas.
   b. A consulta com parâmetros aparece no editor de consultas. Por exemplo, se você escolher o ícone + ao lado`ErrorsByLevel`, o editor de consultas será preenchido com: `$ErrorsByLevel(level=, applicationName=)`
   c. Forneça os valores para os parâmetros (level, applicationName) e execute a consulta. Por exemplo: `$ErrorsByLevel(level= "ERROR", applicationName= "OrderService")`
6. Escolha Executar.

AWS CLI

Para executar uma consulta salva com parâmetros

Use `start-query` com a `$QueryName()` sintaxe:

```
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

Para executar uma consulta salva com parâmetros

Ligue StartQuerycom a `$QueryName()` sintaxe no `queryString` campo:

```
{
  "logGroupNames": ["/aws/lambda/my-function"],
  "startTime": 1707566400,
  "endTime": 1707570000,
  "queryString": "$ErrorsByLevel(level=\"ERROR\", applicationName= \"OrderService
\")"
}
```

Como salvar uma nova versão de uma consulta salva

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. À direita, escolha Consultas.
4. Selecione sua consulta na lista de Consultas salvas. Ela aparece no editor de consulta.
5. Modifique a consulta. Se você precisar executá-la para verificar seu trabalho, escolha Executar consulta.
6. Quando estiver pronto para salvar a nova versão, escolha Ações, Salvar como.
7. Insira um nome para a consulta.
8. (Opcional) Escolha uma pasta na qual deseja salvar a consulta. Selecione Criar novo para criar uma pasta. Se você criar uma pasta, poderá usar caracteres de barra (/) no nome da pasta para definir uma estrutura de pasta. Por exemplo, dar o nome **`folder-level-1/folder-level-2`** a uma nova pasta cria uma pasta de nível superior chamada **`folder-level-1`**, com outra pasta chamada **`folder-level-2`** dentro dela. A consulta é salva em **`folder-level-2`**.

9. (Opcional) Altere os grupos de log da consulta ou o texto da consulta.
10. Escolha Save(Salvar).

Para excluir uma consulta, você deve estar conectado a uma função que tenha a permissão `logs:DeleteQueryDefinition`.

Como editar ou excluir uma consulta salva

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. À direita, escolha Consultas.
4. Selecione sua consulta na lista de Consultas salvas. Ela aparece no editor de consulta.
5. Escolha Ações, Editar ou Ações, Excluir.

Usando consultas salvas com parâmetros

As consultas salvas com parâmetros são modelos de consulta reutilizáveis com espaços reservados nomeados. Em vez de manter várias cópias de consultas quase idênticas, você pode salvar um modelo e fornecer valores de parâmetros diferentes ao executar a consulta. Os parâmetros só são compatíveis com a linguagem de consulta do CloudWatch Logs Insights.

Como funciona

Ao salvar uma consulta, os espaços reservados identificam os valores que você pode fornecer no momento da execução da consulta. Os espaços reservados usam a `{{parameterName}}` sintaxe. Veja a seguir um exemplo de uma consulta salva chamada `ErrorsByLevel` com dois parâmetros `logLevel` `applicationName` e.

```
fields @timestamp, @message
| filter level = {{logLevel}}
| filter applicationName = {{applicationName}}
```

Para executar uma consulta salva, você pode invocá-la usando o nome da consulta prefixado com $ e transmitindo os valores dos parâmetros. O mecanismo de consulta do CloudWatch Logs Insights substitui cada espaço reservado. Se um parâmetro contiver valores padrão, esses valores serão usados se nenhum outro valor for fornecido.

```
# 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()
```

Os nomes de consulta salvos contendo espaços ou caracteres especiais precisam ser colocados entre aspas:

```
$`Errors By Level`(logLevel = "WARN")
```

Amostra de consultas salvas com parâmetros

Adicionando um limite de resultados como parâmetro

Nome da consulta: `ErrorsByLevel` com parâmetros `logLevel` (padrão:`"ERROR"`), `applicationName` (padrão:`"OrderService"`) e `maxResults` (padrão:`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)
```

Usando várias consultas salvas com parâmetros

O exemplo abaixo usa `ErrorsByLevel` uma segunda consulta salva `RecentN` que é definida como `sort @timestamp desc | limit {{count}}` (com parâmetro`count`, padrão`20`). O mecanismo de consulta do CloudWatch Logs Insights expande cada consulta antes de executá-la.

```
# 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
```

Cotas e tratamento de erros

> **Note**
>
> Cada consulta salva pode ter no máximo 20 parâmetros.

A sequência de caracteres de consulta expandida não pode exceder 10.000 caracteres. Os nomes dos parâmetros devem começar com uma letra ou sublinhado. Uma consulta salva não pode fazer referência a outra consulta salva (não há suporte para invocações aninhadas).

Erros comuns

Erro	Causa
Os parâmetros são suportados somente para a linguagem de consulta CWLI	Os parâmetros só são compatíveis com a linguagem de consulta do CloudWatch Logs Insights.
Parâmetros obrigatórios não encontrados em QueryString	O nome de um parâmetro em `--parameters` não tem uma correspondência `{{placeholder}}` na sequência de caracteres de consulta.
A contagem de parâmetros excede o máximo de 20	No momento, as consultas salvas oferecem suporte a apenas 20 parâmetros.
Nome do parâmetro duplicado	A definição da consulta tem parâmetros duplicados em`parameters` .

> **Note**
>
> Para criar ou atualizar uma consulta salva com parâmetros, você precisa da `logs:PutQueryDefinition` permissão. Para executar um, você precisa `logs:StartQuery` `logs:DescribeQueryDefinitions` e.

Adicionar consulta ao painel ou exportar os resultados da consulta

Depois de executar uma consulta, você pode adicionar a consulta a um CloudWatch painel ou copiar os resultados para a área de transferência.

As consultas adicionadas aos painéis são executadas sempre que você carrega o painel e sempre que o painel é atualizado. Essas consultas contam para seu limite de 100 consultas simultâneas do CloudWatch Logs Insights.

Para adicionar resultados da consulta a um painel

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. Escolha um ou mais grupos de logs e execute uma consulta.
4. Escolha Add to dashboard (Adicionar ao painel).
5. Selecione o painel ou escolha Criar novo a fim de criar um painel para os resultados da consulta.
6. Selecione o tipo de widget a ser usado para os resultados da consulta.
7. Insira um nome para o widget.
8. Escolha Add to dashboard (Adicionar ao painel).

Como copiar os resultados da consulta para a área de transferência ou fazer download dos resultados da consulta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. Escolha um ou mais grupos de logs e execute uma consulta.
4. Escolha Exportar resultados e depois a opção desejada.

Exibir as consultas em execução ou o histórico de consultas

Exiba as consultas atualmente em andamento, bem como o histórico de consultas recente.

As consultas em execução no momento incluem as consultas adicionadas a um painel. Você está limitado a 100 consultas simultâneas do CloudWatch Logs Insights por conta, incluindo

consultas adicionadas aos painéis. Além disso, você pode executar 15 consultas simultâneas para OpenSearch Service PPL ou OpenSearch Service SQL.

Para exibir o histórico de consultas recente

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log Insights(Insights de log).
3. Escolha Histórico, se você estiver usando o novo design do console de CloudWatch registros. Se estiver usando o design antigo, escolha Ações, Exibir histórico de consultas para esta conta.

   Uma lista das consultas recentes é exibida. Você pode executar qualquer um deles novamente selecionando a consulta e escolhendo Executar.

   Em Status, CloudWatch Registros exibe Em andamento para todas as consultas em execução no momento.

Criptografe os resultados da consulta com AWS Key Management Service

Por padrão, o CloudWatch Logs criptografa os resultados armazenados de suas consultas do CloudWatch Logs Insights usando o método padrão de criptografia do lado do servidor do CloudWatch Logs. Em vez disso, você pode optar por usar uma AWS KMS chave para criptografar esses resultados. Se você associar uma AWS KMS chave aos resultados da criptografia, o CloudWatch Logs usará essa chave para criptografar os resultados armazenados de todas as consultas na conta.

Se, posteriormente, você desassociar a chave dos resultados da consulta, o CloudWatch Logs retornará ao método de criptografia padrão para consultas posteriores. Mas as consultas executadas enquanto a chave estava associada ainda são criptografadas com essa chave. CloudWatch Os registros ainda podem retornar esses resultados após a desassociação da chave KMS, porque CloudWatch os registros ainda podem continuar referenciando a chave. No entanto, se a chave for desativada posteriormente, o CloudWatch Logs não conseguirá ler os resultados da consulta que foram criptografados com essa chave.

> **Important**
>
> CloudWatch O Logs suporta somente chaves KMS simétricas. Não use uma chave assimétrica para criptografar os resultados da consulta. Para obter mais informações, consulte Usar chaves simétricas e assimétricas.

Limites

- Para executar as etapas a seguir, é necessário ter as seguintes permissões: `kms:CreateKey`, `kms:GetKeyPolicy` e `kms:PutKeyPolicy`.
- Após uma chave ser associada ou desassociada dos resultados da consulta, até cinco minutos poderão ser necessários para que a operação seja efetivada.
- Se você revogar o acesso do CloudWatch Logs a uma chave associada ou excluir uma chave KMS associada, seus dados criptografados no CloudWatch Logs não poderão mais ser recuperados.
- Você não pode usar o CloudWatch console para associar uma chave, é preciso usar a API AWS CLI ou CloudWatch Logs.

Etapa 1: criar um AWS KMS key

Para criar uma chave do KMS, use o seguinte comando create-key:

```
aws kms create-key
```

A saída contém o ID de chave e o nome do recurso da Amazon (ARN) da chave. A seguir está um exemplo de saída:

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

Etapa 2: definir permissões na chave do KMS

Por padrão, todas as chaves do KMS são privadas. Somente o proprietário do recurso pode usá-la para criptografar e descriptografar dados. No entanto, o proprietário do recurso pode conceder permissões para acessar a chave a outros usuários e recursos. Com essa etapa, você concede permissão principal ao serviço de CloudWatch registros para usar a chave. Esse principal de serviço deve estar na mesma AWS região em que a chave está armazenada.

Como prática recomendada, recomendamos que você restrinja o uso da chave somente às AWS contas que você especificar.

Primeiro, salve a política padrão para sua chave do KMS como `policy.json` usando o seguinte comando get-key-policy:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

Abra o arquivo `policy.json` em um editor de texto e adicione a seção em negrito de uma das instruções a seguir. Separe a instrução existente da nova instrução com uma vírgula. Essas declarações usam `Condition` seções para aumentar a segurança da AWS KMS chave. Para obter mais informações, consulte AWS KMS chaves e contexto de criptografia.

A `Condition` seção neste exemplo limita o uso da AWS KMS chave para os resultados da consulta do CloudWatch Logs Insights na conta especificada.

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"
                }
            }
        }
    ]
}
```

Finalmente, adicione a política atualizada usando o seguinte comando put-key-policy:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Etapa 3: associar uma chave do KMS aos resultados da consulta

Para associar a chave do KMS aos resultados da consulta na conta

Use o comando disassociate-kms-key da seguinte forma:

```
aws logs associate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-result:*" --kms-key-id "key-arn"
```

Etapa 4: desassociar uma chave dos resultados da consulta na conta

Para desassociar a chave do KMS associada aos resultados de consulta, use o seguinte comando disassociate-kms-key:

```
aws logs disassociate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-result:*"
```

Gere um resumo em linguagem natural a partir dos resultados da consulta do CloudWatch Logs Insights

Analisar os dados de registro é crucial para entender o comportamento de seus aplicativos, mas interpretar grandes volumes de entradas de registro pode ser demorado. CloudWatch O Logs Insights agora oferece um recurso de resumo em linguagem natural que transforma resultados complexos de consultas em resumos claros e concisos. Esse recurso ajuda você a identificar problemas rapidamente e obter insights acionáveis a partir de seus dados de log.

Como funciona

CloudWatch O Logs Insights pode gerar um resumo legível por humanos dos resultados da sua consulta usando o Amazon Bedrock. O recurso é compatível com todas as linguagens de consulta do CloudWatch Logs Insights e fornece insights claros e acionáveis dos seus dados de registro.

Disponibilidade regional e processamento de dados

> **Important**
>
> Quando você usa esse recurso, os resultados da consulta podem ser processados em um Região da AWS diferente. Por exemplo, se você executar uma consulta no Leste dos EUA (N. da Virgínia), o resumo pode ser gerado no Oeste dos EUA (Oregon).

A tabela a seguir lista o possível processamento Região da AWS para as diferentes regiões nas quais o recurso de resultados da consulta está disponível:

Geografia CloudWatch de registros suportada	Possível região de processamento
Estados Unidos (EUA)	Região Leste dos EUA (Norte da Virgínia) Região Leste dos EUA (Ohio) US West (Oregon) Region
Europa	Região Europa (Frankfurt) Região Europa (Irlanda) Região Europa (Paris) Região Europa (Estocolmo) Região Europa (Londres)
Ásia-Pacífico	Região Leste dos EUA (Norte da Virgínia) Região Leste dos EUA (Ohio) US West (Oregon) Region
América do Sul	Região Leste dos EUA (Norte da Virgínia) Região Leste dos EUA (Ohio) US West (Oregon) Region

Introdução

Para gerar um resumo em linguagem natural

1. Execute sua consulta do CloudWatch Logs Insights.
2. Depois que a consulta for concluída, selecione Resumir resultados.

Permissões

É necessário ter uma das seguintes opções:

- Permissão `CloudWatchLogsFullAccess`
- Permissão `CloudWatchLogsReadOnlyAccess`
- Política personalizada do IAM, incluindo as ações `cloudwatch:GenerateQueryResultsSummary`, `logs:GetQueryResults`, `logs:DescribeQueries` e `logs:FilterLogEvents`

Privacidade de dados

Os resultados da sua consulta são processados com segurança e não são usados para treinar ou melhorar o CloudWatch Logs Insights ou o Amazon Bedrock. Se você optar por fornecer feedback sobre o resumo dos resultados da consulta usando os botões de feedback, seu feedback indica seu nível de satisfação com o recurso fornecido no CloudWatch Logs Insights.

Automatizando a análise de registros com consultas agendadas

As consultas agendadas permitem que você automatize a execução de consultas de log em uma programação regular. Em vez de executar consultas manualmente para analisar seus dados de log, você pode configurar consultas agendadas para serem executadas automaticamente e entregar resultados para destinos como buckets do Amazon S3, barramentos de EventBridge eventos da Amazon ou tabelas de pesquisa. Essa automação é ideal para gerar relatórios regulares, monitorar tendências, manter as tabelas de pesquisa atualizadas com seus dados de registro mais recentes ou acionar processos posteriores com base nos resultados da análise de registros.

As consultas agendadas oferecem suporte a todos os três idiomas de consulta disponíveis para consultas de log:

- Linguagem de consulta do Logs Insights (Logs Insights QL)
- OpenSearch Linguagem de processamento canalizado de serviço (PPL)
- OpenSearch Linguagem de consulta estruturada de serviços (SQL)

Conteúdo

- Entendendo os conceitos de consultas agendadas
- Referência de expressão de cronograma
- Práticas recomendadas
- Começando com consultas agendadas
- Configurando destinos do S3 para consultas agendadas
- Configurando destinos da tabela de pesquisa para consultas agendadas
- Solução de problemas de consultas agendadas

Entendendo os conceitos de consultas agendadas

Antes de criar consultas agendadas, entenda esses conceitos-chave que afetam a forma como suas consultas são executadas e onde os resultados são entregues.

Separação de funções do IAM

As consultas agendadas exigem duas funções separadas do IAM: uma para executar consultas e outra para entregar resultados a destinos como buckets do Amazon S3, barramentos de EventBridge eventos da Amazon ou tabelas de pesquisa. Entender por que essa separação existe ajuda você a configurar as permissões corretamente e a usar os benefícios operacionais e de segurança que ela oferece.

A arquitetura de duas funções divide as responsabilidades entre acesso e entrega de dados. A função de execução de consultas acessa seus dados de log e executa consultas, enquanto a função de entrega de destino grava os resultados no destino escolhido. Essa separação segue o princípio do menor privilégio — cada função tem somente as permissões necessárias para sua função específica.

Função de execução da consulta

Permite que o CloudWatch CloudWatch Logs execute consultas do Logs Insights em seu nome. Essa função precisa de permissões para acessar seus grupos de registros e executar consultas, mas não precisa acessar os recursos de destino. Permissões obrigatórias:

- `logs:StartQuery`
- `logs:StopQuery`
- `logs:GetQueryResults`
- `logs:DescribeLogGroups`
- `logs:Unmask`se for necessário desmascarar dados

Para grupos de KMS-encrypted registros: `kms:Decrypt` e `kms:DescribeKey` permissões para a chave KMS usada para criptografar os grupos de registros. Essas permissões também precisam ser adicionadas.

Requisito de relacionamento de confiança: a função de execução da consulta deve incluir uma política de confiança que permita que o serviço CloudWatch Logs (`logs.amazonaws.com`) assuma a função. Sem essa relação de confiança, as consultas agendadas falharão com erros de permissão.

Exemplo de política de confiança para a função de execução da consulta:

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

```
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Exemplo de política de permissões para a função de execução da consulta:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:StartQuery",
                "logs:StopQuery",
                "logs:GetQueryResults",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        }
    ]
}
```

Função de entrega no destino

Permite que CloudWatch os registros entreguem os resultados da consulta ao destino escolhido. Essa função só precisa de permissões para o serviço de destino específico, seguindo o princípio do menor privilégio. As permissões necessárias variam de acordo com o tipo de destino.

Requisito de relacionamento de confiança: a função de entrega de destino também deve incluir uma política de confiança que permita que o serviço CloudWatch Logs (`logs.amazonaws.com`) assuma a função.

Exemplo de política de permissões para a função de entrega de destino do S3:

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::your-scheduled-query-results-bucket/*"
        }
    ]
}
```

Exemplo de política de permissões para uma função de entrega de destino da tabela de pesquisa:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLookupTable",
                "logs:UpdateLookupTable",
                "logs:GetQueryResults"
            ],
            "Resource": "*"
        }
    ]
}
```

Essa separação oferece benefícios práticos para suas operações. Do ponto de vista da segurança, se você precisar alterar o local onde os resultados são entregues, modifique apenas a função de entrega de destino sem alterar as permissões de execução da consulta. Para fins de conformidade e auditoria, você pode rastrear claramente qual função acessa dados de registro confidenciais e qual função grava em sistemas externos. Isso facilita a demonstração de que sua infraestrutura de análise de registros segue as melhores práticas de segurança.

Cross-region e uso entre contas

Uma consulta agendada é criada em uma região específica e executada nessa região. No entanto, você pode consultar grupos de registros e fornecer resultados em todas as regiões e contas. Você

precisa configurar uma ou mais AWS contas como contas de monitoramento e vinculá-las a várias contas de origem. Uma conta de monitoramento é uma AWS conta central que pode visualizar e interagir com os dados de observabilidade gerados pelas contas de origem. Uma conta de origem é uma AWS conta individual que gera dados de observabilidade para os recursos que residem nela. As contas de origem compartilham os dados de observabilidade com a conta de monitoramento. Assim, você pode configurar consultas agendadas a partir da conta de monitoramento usando os grupos de log de todas as contas vinculadas.

Consultando grupos de registros entre regiões

Sua consulta agendada pode acessar grupos de registros em qualquer região. Especifique grupos de log usando seu formato ARN completo:`arn:aws:logs:region:account-id:log-group:log-group-name`. As necessidades `logs:StartQuery` e `logs:GetQueryResults` permissões da função de execução da consulta para grupos de registros em todas as regiões de destino.

Important

Ao consultar grupos de registros ou fornecer resultados entre regiões, os dados de registro ultrapassam os limites regionais. Considere o seguinte:

- Requisitos de residência de dados - Garanta que a transferência de dados entre regiões esteja em conformidade com as políticas de governança de dados e os requisitos normativos da sua organização
- Custos de transferência de Cross-region dados - a transferência de dados incorre em custos adicionais
- Latência de rede - as consultas que acessam grupos de log em regiões distantes podem apresentar maior latência

Para otimizar o desempenho e a economia, crie consultas agendadas na mesma região dos seus grupos de registros primários.

Abordagem alternativa: use a centralização de CloudWatch registros para replicar dados de registro de várias contas e regiões em uma conta de monitoramento central. Isso permite que você crie consultas agendadas em uma única região que acesse todos os seus registros centralizados, evitando consultas entre regiões e simplificando o gerenciamento de permissões do IAM.

Expressões de cronograma e tratamento de fuso horário

A programação que você define determina quando sua consulta é executada e com que frequência ela é executada. Escolher a expressão de agendamento correta afeta o momento em que você recebe os resultados e a quantidade de dados que consulta. Compreender os tipos de expressão ajuda você a escolher entre simplicidade e precisão.

As expressões Cron fornecem controle preciso sobre o tempo, permitindo que você especifique horários, dias da semana ou dias do mês exatos. Use expressões cron quando precisar que as consultas sejam executadas em horários comerciais específicos ou se alinhem aos cronogramas operacionais. No console, você também pode agendar consultas usando opções fáceis de calendário.

Expressão cron

Execute consultas em horários específicos. Formato: `cron(minute hour day-of-month month day-of-week year)`. Exemplos:

- `cron(0 9 * * ? *)`- Todos os dias às 9:00 UTC
- `cron(0 18 ? * MON-FRI *)`- Dias úteis às 18:00 UTC
- `cron(0 0 1 * ? *)`- Primeiro dia de cada mês à meia-noite UTC
- `cron(0 12 ? * SUN *)`- Todos os domingos ao meio-dia UTC
- `cron(30 8 1 1 ? *)`- 1º de janeiro às 8:30 UTC

Todas as consultas agendadas são executadas em UTC, independentemente do fuso horário local ou da localização AWS dos recursos. Isso é particularmente importante quando você agenda consultas para horários comerciais ou análises urgentes. Por exemplo, se sua empresa opera no horário do leste dos EUA e você quer um relatório diário às 9h ET, você precisa considerar a diferença UTC (14:00 UTC durante o horário de verão, 13:00 UTC caso contrário). Planeje suas expressões de agendamento com o UTC em mente para garantir que as consultas sejam executadas nos horários pretendidos.

Escolhendo um idioma de consulta

As consultas agendadas suportam três linguagens de consulta diferentes, e sua escolha afeta a forma como você escreve as consultas e a facilidade com que sua equipe pode mantê-las. O idioma certo depende dos seus requisitos de análise e das habilidades existentes da sua equipe.

Se você está principalmente filtrando e agregando dados de log, a Linguagem de Consulta do CloudWatch Logs Insights oferece a sintaxe mais direta. Para transformações de dados complexas em que você precisa remodelar ou enriquecer dados em várias etapas, a abordagem de pipeline da PPL facilita o acompanhamento da lógica. Quando você precisa realizar uniões ou agregações complexas semelhantes às operações de banco de dados, o SQL fornece uma sintaxe familiar que equipes experientes em bancos de dados podem adotar rapidamente.

CloudWatch Linguagem de consulta do Logs Insights (CWLI)

Purpose-built para análise de registros com sintaxe intuitiva. Ideal para:

- Text-based análise e filtragem de registros
- Time-series agregações e estatísticas
- Equipes novas na análise de registros

OpenSearch Linguagem de processamento por tubulação de serviço (PPL)

Pipeline-based linguagem de consulta com poderosos recursos de transformação de dados. Ideal para:

- Enriquecimento e transformações de dados complexos
- Multi-step fluxos de trabalho de processamento de dados
- Equipes familiarizadas com o processamento baseado em pipeline

OpenSearch Linguagem de consulta estruturada de serviços (SQL)

Sintaxe SQL padrão para consultas familiares em estilo de banco de dados. Ideal para:

- Junções e agregações complexas
- Inteligência de negócios e relatórios
- Equipes com forte experiência em SQL

Seleção de destinos e casos de uso

O local onde você envia os resultados da consulta determina o que você pode fazer com eles. Essa escolha molda todo o seu fluxo de trabalho posterior, esteja você criando análises de longo prazo, acionando respostas automatizadas ou ambos. Entender os pontos fortes de cada tipo de destino ajuda você a projetar a arquitetura certa para seu caso de uso.

Os destinos do Amazon S3 são otimizados para armazenamento e processamento em lote. Quando você precisa manter os resultados das consultas por meses ou anos, analisar tendências ao longo

do tempo ou inserir dados em plataformas de análise, o Amazon S3 fornece armazenamento econômico com retenção ilimitada. EventBridge os destinos são otimizados para automação em tempo real. Quando os resultados da consulta devem acionar ações imediatas, como enviar alertas, iniciar fluxos de trabalho ou atualizar sistemas, os resultados EventBridge são apresentados como eventos aos quais seus aplicativos podem responder instantaneamente. Por padrão, todos os eventos de conclusão de consultas são enviados automaticamente como eventos para o barramento de eventos padrão, permitindo a integração com sistemas de processamento downstream, funções do Lambda ou outras arquiteturas orientadas por eventos. Os resultados só são publicados nos destinos quando a consulta é executada com êxito. Os destinos da tabela de pesquisa são otimizados para manter os dados de referência atualizados. O destino da tabela de pesquisa preenche ou atualiza automaticamente a tabela de pesquisa especificada com os resultados da consulta em cada execução agendada, para que outras consultas possam referenciar os dados mais recentes com o comando. `lookup`

Destinos do Amazon S3

Armazene os resultados da consulta como arquivos JSON para retenção a longo prazo e processamento em lote. Os destinos do Amazon S3 funcionam melhor nos seguintes cenários:

- Análise histórica e arquivamento de dados
- Integração com data lakes e plataformas de análise
- Requisitos de conformidade e auditoria
- Cost-effective armazenamento de grandes conjuntos de resultados

EventBridge destinos

Envie resultados de consultas como eventos para processamento e automação em tempo real. Use o `queryId` in the event para recuperar os resultados da consulta, que permanecem disponíveis por 30 dias após a execução da consulta. EventBridgeos destinos funcionam melhor nos seguintes cenários:

- Acionando respostas automatizadas aos resultados da consulta
- Integração com fluxos de trabalho sem servidor e funções do Lambda
- Real-time sistemas de alerta e notificação
- Event-driven arquiteturas e microsserviços

Destinos da tabela de pesquisa

Crie ou atualize automaticamente uma tabela de pesquisa com os resultados da consulta em cada execução agendada. Cada atualização é uma substituição completa do conteúdo da tabela. Os destinos da tabela de pesquisa funcionam melhor nos seguintes cenários:

- Manter os dados de referência atualizados para o `lookup` comando em suas consultas de log
- Manutenção de listas de permissões, listas de negação ou inventários de entidades derivados de dados de registro
- Enriquecer consultas com resumos de atividades recentes, como listas ativas de usuários ou recursos

Formato e estrutura do resultado da consulta

As consultas agendadas fornecem resultados no formato JSON, mas cada tipo de destino recebe uma carga diferente. Para um destino de tabela de pesquisa, os resultados da consulta se tornam o conteúdo da tabela de pesquisa e cada execução substitui esse conteúdo. Para obter mais informações, consulte Configurando destinos da tabela de pesquisa para consultas agendadas.

Os destinos do Amazon S3 recebem as linhas de resultados da consulta. Cada objeto contém uma matriz JSON com uma entrada para cada linha no conjunto de resultados, e cada entrada mapeia os nomes dos campos de saída da consulta de acordo com seus valores. O objeto não contém metadados de consulta nem estatísticas de consulta e omite o `@ptr` campo mesmo que a consulta solicite, porque esse campo só pode ser usado no console.

EventBridge os destinos recebem metadados da consulta, incluindo as estatísticas da consulta, mas nenhuma linha de resultados. Para recuperar as linhas de uma consulta concluída, chame GetQueryResults com o valor de `queryId` do evento.

O exemplo a seguir mostra o evento no qual o CloudWatch Logs é publicado EventBridge quando uma consulta agendada é concluída.

```
{
    "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\n| filter @message
 like /ERROR/\n| sort @timestamp desc\n| limit 10000",
        "logGroupIdentifiers": [
            "/aws/lambda/my-function"
        ],
        "status": "Complete",
        "startTime": 1763465460,
        "statistics": {
            "recordsMatched": 1842,
            "recordsScanned": 48325,
            "estimatedRecordsSkipped": 0,
            "bytesScanned": 12081250,
            "estimatedBytesSkipped": 0,
            "logGroupsScanned": 1,
            "resultCount": 1842
        }
    }
}
```


Essa consulta não é agregada e retornou menos linhas do que a `limit` de 10.000, portanto, cada um dos 1.842 eventos de log correspondentes se tornou uma linha de saída `recordsMatched` e `resultCount` é igual. Uma consulta que agrega, ou uma cujo conjunto de resultados é truncado por`limit`, produz um `resultCount` valor menor que. `recordsMatched` Para obter mais informações, consulte Compreendendo os registros digitalizados, os registros combinados e a contagem de resultados.

Os principais elementos incluem:

- `statistics`- Contadores que descrevem a quantidade de dados de registro que a consulta leu e o tamanho do conjunto de resultados. Para obter uma descrição de cada campo, consulte a tabela a seguir.
- `startTime`- Quando a execução da consulta começou (timestamp Unix)
- `queryString`- A consulta real que foi executada
- `queryId`- ID de consulta da consulta usando o qual os resultados podem ser recuperados

- `logGroupIdentifiers`- Lista de grupos de registros que foram consultados
- `status`- Status de execução da consulta (Concluída, Falha etc.)

A tabela a seguir descreve cada campo no `statistics` objeto. Para ver as definições de API desses campos, consulte QueryStatistics.

Campo	Description
`recordsScanned`	O número total de eventos de registro verificados durante a consulta.
`recordsMatched`	O número de eventos de log que corresponderam à string de consulta. Esse valor conta os eventos de registro, não as linhas de saída. Para o número de linhas no conjunto de resultados, use`resultCount` .
`resultCount`	O número de linhas no conjunto de resultados da consulta. Esse valor conta somente as linhas que sobreviveram a todas as operações na consulta, portanto, pode ser menor que`recordsMatched` . Ele abrange todas as páginas de resultados `GetQueryResults` retornados. Para obter mais informações, consulte Compreendendo os registros digitalizados, os registros combinados e a contagem de resultados.
`estimated RecordsSkipped`	Uma estimativa do número de eventos de log que foram ignorados ao processar essa consulta, porque a consulta continha um campo indexado. Ignorar essas entradas reduz os custos da consulta e melhora o tempo de desempenho da consulta. Para obter mais informações, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.
`bytesScanned`	O número total de bytes nos eventos de log verificados durante a consulta.
`estimated BytesSkipped`	Uma estimativa do número de bytes nos eventos de log que foram ignorados ao processar essa consulta, porque a consulta continha um campo indexado.
`logGroupsScanned`	O número de grupos de registros que foram examinados por essa consulta.

Compreendendo os registros digitalizados, os registros combinados e a contagem de resultados

Três das estatísticas de consulta contam coisas diferentes, e compará-las diretamente pode ser enganoso. Cada um mede um estágio diferente do processamento da consulta:

- `recordsScanned`- O número de eventos de registro que a consulta leu de seus grupos de registros. Essa é a entrada para a consulta.
- `recordsMatched`- O número desses eventos de log que corresponderam à string de consulta. Esse valor conta os eventos de registro.
- `resultCount`- O número de linhas no conjunto de resultados que a consulta produziu. Esse valor conta as linhas de saída.

Cada estágio restringe os dados. Um comando como o que `stats` combina vários eventos de log em uma única linha de saída, portanto, `resultCount` pode ser muito menor que `recordsMatched` o. Um grande `recordsMatched` com um pequeno `resultCount` não significa que faltam linhas no conjunto de resultados.

Exemplo — Agregação

A consulta a seguir conta as mensagens de erro em cada fluxo de log de um grupo de logs em um período de uma hora.

```
filter @message like /ERROR/
| stats count(*) as errorCount by @logStream
```

Se a consulta ler 1.500.000 eventos de log, 24.318 deles contiveremERROR, e os eventos de registro correspondentes vierem de 12 fluxos de log, a consulta será concluída com as estatísticas a seguir.

```
"statistics": {
    "recordsMatched": 24318,
    "recordsScanned": 1500000,
    "estimatedRecordsSkipped": 0,
    "bytesScanned": 450000000,
    "estimatedBytesSkipped": 0,
    "logGroupsScanned": 1,
    "resultCount": 12
}
```

`stats`produz uma linha para cada fluxo de log, então `resultCount` é 12 e `recordsMatched` é 24.318. Os 12 valores de `errorCount` somam 24.318.

Exemplo — Post-aggregation filtro

A consulta a seguir mantém somente os fluxos de log que produziram mais de 1.000 erros.

```
filter @message like /ERROR/
| stats count(*) as errorCount by @logStream
| filter errorCount > 1000
```

O `filter` comando final é executado após o agrupamento, então ele remove linhas do conjunto de resultados em vez de registrar eventos da verificação. Se 3 dos 12 fluxos de log tiverem um valor `errorCount` maior que 1.000, então será `resultCount` 3 em vez de 12. `recordsScanned`e `bytesScanned` não altere, porque a consulta lê os mesmos dados de log de qualquer maneira.

Exemplo — Limite

Um `limit` comando reduz `resultCount` sem qualquer agregação. A consulta a seguir retorna as 100 mensagens de erro mais recentes.

```
filter @message like /ERROR/
| sort @timestamp desc
| limit 100
```

Se a consulta verificar os mesmos 1.500.000 eventos de log e corresponder aos mesmos 24.318, então será 100, porque limita o `resultCount` conjunto de resultados a 100 `limit` linhas. No console, você vê essa relação como Mostrando 100 dos 24.318 registros correspondidos.

Cada estatística responde a uma pergunta diferente.

Quantas linhas essa consulta retornou?

- Use `resultCount`. Um valor de 0 significa que a consulta não produziu nenhuma linha. Não use `recordsMatched` para essa finalidade, pois ele conta eventos de log em vez de linhas.

Quantos dados de registro essa consulta leu?

- Utilizar `recordsScanned` e `bytesScanned`. O volume de digitalização determina o custo e o tempo de execução de uma consulta. Para reduzi-lo, reduza o intervalo de tempo, consulte

menos grupos de registros ou crie índices de campo. Para obter mais informações, consulte Crie índices de campo para melhorar o desempenho da consulta e reduzir o volume de escaneamento.

Quantos eventos de log corresponderam a essa consulta?

Use `recordsMatched`.

> **Note**
>
> CloudWatch Os registros omitem uma estatística do `statistics` objeto quando nenhum valor está disponível para ele, em vez de reportar a estatística como 0.
> Se o consumidor do seu evento não encontrar `resultCount` em um evento, trate o valor como desconhecido em vez de 0. Escreva para consumidores de eventos de forma que eles tolerem estatísticas ausentes e ignorem estatísticas que eles não reconheçam.

Referência de expressão de cronograma

Use essas tabelas de referência para criar expressões de agendamento para suas consultas agendadas. Todos os horários são em UTC.

Sintaxe da expressão Cron

Formato: `cron(minute hour day-of-month month day-of-week year)`.

Caso de uso	Expressão cron	Description	Use quando
Programações diárias	`cron(0 9 * * ? *)`	Todos os dias às 9:00 UTC	Relatórios diários
	`cron(0 */6 * * ? *)`	A cada 6 horas (00:00, 06:00, 12:00, 18:00 UTC)	Monitoramento frequente
	`cron(30 2 * * ? *)`	Todos os dias às 2:30 UTC	Off-peak análise

Caso de uso	Expressão cron	Description	Use quando
Horário comercial	`cron(0 9-17 ? * MON-FRI *)`	A cada hora, das 9h às 17h, de segunda a sexta-feira UTC	Monitoramento de negócios
	`cron(0 18 ? * MON-FRI *)`	Dias úteis às 18:00 UTC	Fim do dia útil
	`cron(0 8,12,17 ? * MON-FRI *)`	8h, meio-dia e 17h nos dias úteis UTC	Horários comerciais importantes
Programações semanais	`cron(0 12 ? * SUN *)`	Todos os domingos ao meio-dia UTC	Resumos semanais
	`cron(0 9 ? * MON *)`	Todas as segundas-feiras às 9:00 UTC	Relatórios de início da semana
	`cron(0 23 ? * FRI *)`	Todas as sextas-feiras às 23:00 UTC	Limpeza de fim de semana
Programações mensais	`cron(0 0 1 * ? *)`	Primeiro dia de cada mês à meia-noite UTC	Relatórios mensais
	`cron(0 9 L * ? *)`	Último dia de cada mês às 9:00 UTC	Processamento no final do mês
	`cron(0 10 1 1,4,7,10 ? *)`	Primeiro dia de cada trimestre às 10:00 UTC	Análise trimestral

Caso de uso	Expressão cron	Description	Use quando
Alta frequência	`cron(*/15 * * * ? *)`	A cada 15 minutos	Real-time monitoramento
	`cron(0,30 * * * ? *)`	A cada 30 minutos (às :00 e às :30)	Verificações frequentes
	`cron(0 */2 * * ? *)`	A cada 2 horas	Intervalos regulares
Casos especiais	`cron(30 8 1 1 ? *)`	1º de janeiro às 8:30 UTC	Relatórios anuais
	`cron(0 6 * * SAT,SUN *)`	Fins de semana às 6:00 UTC	Processamento de fim de
	`cron(0 0 ? * MON#1 *)`	Primeira segunda-feira de cada mês à meia-noite UTC	Planejamento mensal

Referência de campo de expressão Cron

Campo	Valores	Curingas	Exemplos
Minuto (1º)	0-59	`* , - /`	`0`(no máximo da hora), `*/15` (a cada 15 minutos), `0,30` (duas vezes por hora)
Hora (2ª)	0-23	`* , - /`	9(9h), `*/2` (a cada 2 horas), `9-17` (horário comercial)

Campo	Valores	Curingas	Exemplos
Day-of-month (3º)	1-3,1 L, W,	* , - / ?	1(1º dia), L (último dia), ? (ao usar o dia da semana)
Mês (4º)	1-12 ou JAN-DEC	* , - /	1(janeiro),JAN, 1,4,7,10 (trimestral)
Day-of-week (5º)	1-7 ou SUN-SAT	* , - / ? # L	MON-FRI(dias úteis),SUN, MON#1 (1ª segunda-feira)
Ano (6º)	1970-2199	* , - /	*(todos os anos), 2024 (ano específico), 2024-2026 (intervalo)

Caracteres curinga e expressões especiais

***** (asterisco)

Corresponde a todos os valores no campo. Exemplo: * no campo hora significa a cada hora.

?(ponto de interrogação)

Sem valor específico. Use no dia do mês ou no dia da semana quando o outro for especificado. Exemplo: use ? no dia do mês ao especificar MON-FRI no dia da semana.

- (traço)

Faixa de valores. Exemplo: MON-FRI (de segunda a sexta-feira), 9-17 (das 9h às 17h).

,(vírgula)

Vários valores específicos. Exemplo: MON,WED,FRI (segunda, quarta, sexta-feira), 8,12,17 (8h, meio-dia, 17h).

/(corte)

Valores ou incrementos de etapas. Exemplo: 0/15 em minutos significa a cada 15 minutos começando no minuto 0 (0, 15, 30, 45). */2em horas significa a cada 2 horas.

L(último)

Último dia do mês ou última ocorrência do dia da semana. Exemplo: L em dia do mês significa último dia do mês. FRILsignifica última sexta-feira do mês.

W(dia da semana)

Dia da semana mais próximo. Exemplo: `15W` significa o dia da semana mais próximo até o dia 15 do mês.

#(enésima ocorrência)

Enésima ocorrência de dia da semana em um mês. Exemplo: `MON#1` significa primeira segunda-feira do mês, `FRI#2` significa segunda sexta-feira do mês.

Padrões comuns e melhores práticas

- Para aplicativos comerciais: use o `MON-FRI` horário comercial (por exemplo,`9-17`) para evitar a execução de consultas durante fins de semana ou fora do horário comercial.
- Para monitoramento de alta frequência: use incrementos como `*/15` (a cada 15 minutos), mas esteja atento aos limites de simultaneidade de consultas.
- Para eficiência de recursos: agende consultas que consomem muitos recursos fora do horário de pico usando os horários matinais, como o UTC. `2-6`
- Para relatórios mensais: use `L` no último dia do mês ou em datas específicas, como `1` no primeiro dia, para garantir um tempo consistente.

Práticas recomendadas

Siga essas práticas recomendadas para garantir operações de consulta agendadas confiáveis e eficientes:

Otimização de consultas

- Teste as consultas manualmente antes de agendar para verificar o desempenho e os resultados
- Use índices de filtro no início de sua consulta para reduzir o processamento de dados
- Limite os intervalos de tempo para evitar tempos limite com grupos de registros de alto volume
- Considere a complexidade da consulta e os limites de tempo de execução

Planejamento do cronograma

- Evite a sobreposição de execuções garantindo que as consultas sejam concluídas antes da próxima execução agendada

- Considere atrasos na ingestão de registros ao definir intervalos de tempo
- Use expressões cron para horários específicos
- Distribua os cronogramas para garantir que você não atinja o limite de simultaneidade de consultas

Monitoramento e manutenção

- Monitore o histórico de execução regularmente para identificar falhas ou problemas de desempenho
- Revise e atualize as funções do IAM periodicamente para manter a segurança
- Teste a acessibilidade do destino antes de implantar na produção

Autorização

- Todas as APIs para consulta agendada são autorizadas no recurso de consulta agendada e não nos recursos que ele recebe na entrada, como grupos de log. Configure as políticas do IAM adequadamente
- Gerencie a autorização de grupos de log usando a função de execução passada nas APIs

Começando com consultas agendadas

Ao criar uma consulta agendada, você configurará vários componentes principais que definem como sua consulta é executada e onde os resultados são entregues. Compreender esses componentes ajudará você a configurar uma análise de registros automatizada eficaz.

Cada consulta agendada consiste nos seguintes componentes principais:

Configuração de consulta

A string de consulta do CloudWatch Logs Insights, os grupos de registros de destino e a linguagem de consulta a serem usados na análise.

Expressão de programação

Uma expressão cron ou calendário de frequência que define quando a consulta é executada. Você pode especificar as configurações de fuso horário para garantir que as consultas sejam executadas no horário local correto. O console exibe uma descrição legível por humanos de sua programação, como “Execute a consulta todas as terças-feiras às 15:10 por um intervalo de tempo de 5 minutos, com efeito imediato, no UTC, até indefinidamente”.

Intervalo de tempo

O período de lookback para cada execução de consulta, definido por uma diferença de horário de início em relação ao tempo de execução. Isso determina a quantidade de dados históricos que cada execução de consulta analisará.

Prévia do cronograma de execução

O console mostra as próximas três consultas agendadas com datas e horas exatas (por exemplo, 2025/10 /28 15:10, UTC; 2025/11 /04 15:10, UTC; 2025/11 /11 15:10, UTC), ajudando você a verificar se a programação está configurada corretamente.

Destinos

Onde os resultados da consulta são entregues após a execução bem-sucedida. Os destinos compatíveis incluem buckets do Amazon S3 e, por padrão, os metadados dos resultados são enviados para o barramento de eventos padrão.

Perfil de execução

Uma função do IAM que o CloudWatch Logs presume para executar a consulta e entregar os resultados aos destinos especificados.

Antes de criar consultas agendadas, verifique se você tem as permissões e os recursos necessários configurados.

Criação de uma consulta agendada

Crie uma consulta agendada que execute automaticamente as consultas do CloudWatch Logs Insights e forneça resultados nos destinos escolhidos.

Pré-requisitos

Antes de criar uma consulta agendada, verifique se você tem o seguinte:

- Grupos de registros - Um ou mais grupos de registros contendo os dados que você deseja analisar
- Função de execução do IAM - Uma função do IAM com as seguintes permissões:
  - `logs:StartQuery`- Permissão para iniciar consultas do CloudWatch Logs Insights
  - `logs:GetQueryResults`- Permissão para recuperar os resultados da consulta

  - `logs:DescribeLogGroups`- Permissão para acessar as informações do grupo de registros. Isso só é necessário para grupos de registros baseados em prefixos para descoberta de grupos de registros
- Permissões de destino — Permissões adicionais do IAM para o destino escolhido:
  - Para destinos do Amazon S3: `s3:PutObject`
- Para AWS CLI e uso da API - AWS Credenciais configuradas com permissões para chamar APIs de CloudWatch registros

Para obter exemplos detalhados de políticas do IAM, consulteGerenciamento de identidade e acesso para Amazon CloudWatch Logs. Além disso, você pode ter apenas 1000 consultas agendadas por conta.

Console

Para criar uma consulta agendada (console)

1. Abra o console do CloudWatch Logs em https://us-east-1.console.aws.amazon.com/cloudwatch/home? region=us-east-1 #logs. V2:logs-insights
2. No painel de navegação, escolha Logs Insights.
3. Escolha Criar consulta agendada.
4. Na seção Definição da consulta:
   a. Em Linguagem de consulta, escolha a linguagem de consulta a ser usada na lista.
   b. Em Cadeia de caracteres de consulta, insira sua consulta do CloudWatch Logs Insights na caixa.
   c. Em Grupos de registros, selecione os grupos de registros a serem consultados na lista.
5. Na seção Configuração do cronograma:
   a. Para a expressão Schedule, configure quando a consulta é executada. Escolha entre opções predefinidas ou insira uma expressão cron personalizada.
   b. Em Efetivo após a criação, especifique quando o cronograma se tornará ativo. Escolha começar imediatamente ou em uma data e hora específicas usando o YYYY/MM/DD formato.

c. Em Intervalo de tempo, especifique o período de lookback para cada execução de consulta. Insira a duração em minutos que define a distância entre o tempo de execução e a consulta.

d. Para Continuar indefinidamente, especifique quando a programação termina. Escolha executar indefinidamente ou até uma data e hora específicas usando o YYYY/MM/DD formato.

6. O console exibe as próximas três consultas agendadas com base em sua configuração, mostrando as datas e horas exatas em UTC em que a consulta será executada.

7. Na seção Publicar resultados da consulta no S3 - opcional (se estiver usando o destino S3):

a. Para o bucket S3, selecione Esta conta se o bucket de destino estiver na mesma AWS conta ou selecione Outra conta se o bucket estiver em uma AWS conta diferente e forneça o ID da conta proprietária do bucket como entrada.

b. Para o URI do Amazon S3, insira o bucket e o prefixo do Amazon S3 onde os resultados serão armazenados (por exemplo,). `s3://my-bucket/query-results/` Se você selecionou Esta conta, pode escolher Navegar no Amazon S3 para navegar e selecionar um local existente do Amazon S3.

c. (Opcional) Para o ARN da chave KMS, insira o ARN de uma AWS KMS chave gerenciada pelo cliente para criptografar os resultados da consulta usando. SSE-KMS A chave deve estar na mesma AWS região do bucket de destino do Amazon S3.

8. Na seção Função do IAM para publicar resultados de consultas no Amazon S3, escolha uma das seguintes opções:

a. Escolha Auto-create uma nova função com permissões padrão para configurar automaticamente uma função do IAM com as permissões necessárias para que os CloudWatch Logs entreguem os resultados da consulta ao Amazon S3.

b. Escolha Usar uma função existente para selecionar uma função existente do IAM com as políticas necessárias para que os CloudWatch Logs entreguem os resultados da consulta ao Amazon S3. Use o campo de pesquisa para encontrar e selecionar a função do IAM apropriada na lista.

> **Note**
>
> Se o gerente de funções estiver habilitado em sua conta, CloudWatch anexará a função para você e as opções de função descritas aqui (por exemplo, o botão

Criar nova função) serão substituídas pela opção Personalizar. Para usar um perfil diferente, escolha Personalizar. Para obter mais informações sobre a criação de funções do IAM, consulte Criação de funções do IAM no Guia do usuário do IAM.

9. Na seção Função do IAM para execução agendada de consultas, escolha uma das seguintes opções:

   a. Escolha Auto-create uma nova função com permissões padrão para configurar automaticamente uma função do IAM com as permissões necessárias para que o CloudWatch Logs execute consultas agendadas.

   b. Escolha Usar uma função existente para selecionar uma função existente do IAM com as políticas necessárias para que CloudWatch os registros executem consultas agendadas. Use o campo de pesquisa para encontrar e selecionar a função do IAM apropriada na lista.

> **Note**
>
> Se o gerente de funções estiver habilitado em sua conta, CloudWatch anexará a função para você e as opções de função descritas aqui (por exemplo, o botão Criar nova função) serão substituídas pela opção Personalizar. Para usar um perfil diferente, escolha Personalizar. Para obter mais informações sobre a criação de funções do IAM, consulte Criação de funções do IAM no Guia do usuário do IAM.

10. Escolha Criar agendamento para criar a consulta agendada.

AWS CLI

Para criar uma consulta agendada (AWS CLI)

- Use o `create-scheduled-query` comando para criar uma nova consulta agendada:

```
aws logs create-scheduled-query \
    --name "ErrorAnalysisQuery" \
    --query-language "CWLI" \
    --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

Para criar uma consulta agendada (API)

- Use a `CreateScheduledQuery` ação para criar uma nova consulta agendada. O exemplo a seguir cria uma consulta agendada que é executada a cada hora:

```
{
    "name": "ErrorAnalysisQuery",
    "queryLanguage": "CWLI",
    "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"
}
```

Depois de criar a consulta agendada, você pode visualizá-la e gerenciá-la na página de consultas agendadas e usando a ListScheduledQueries API, que mostra todas as suas consultas agendadas com seus nomes, datas de criação, status da última execução, hora da última ativação e frequência de repetição.

Visualizando e gerenciando consultas agendadas

As informações a seguir estão disponíveis para cada consulta:

Nome

O nome exclusivo que você atribuiu à consulta agendada. Selecione o nome para ver a configuração detalhada e o histórico de execução.

Data de Criação

A data em que a consulta agendada foi criada, exibida em YYYY-MM-DD formato.

Status da última execução

O status de execução da consulta mais recente executada. Os possíveis valores incluem:

- Concluída - A consulta foi executada com êxito e os resultados foram entregues a todos os destinos configurados.
- Falha - A execução da consulta ou a entrega do resultado falharam. Verifique o histórico de execução para obter detalhes do erro.
- Consulta inválida - A consulta é inválida e tem problemas sintáticos
- Tempo limite - A consulta atingiu o tempo limite. Uma consulta atinge o tempo limite automaticamente após 60 minutos

Hora da última ativação

A data e a hora em que a consulta foi executada pela última vez, exibidas em YYYY-MM-DD HH:MM:SS formato. Mostra Nunca se a consulta ainda não tiver sido executada.

Repetindo cada

A frequência do agendamento da consulta. Mostra Personalizado para consultas usando expressões cron ou descrições de frequência específicas para agendas mais simples.

A página de consultas agendadas fornece uma visão geral de todas as suas consultas agendadas, mostrando seu status atual e histórico de execução para que você possa visualizar, monitorar e gerenciar todas as suas consultas agendadas a partir de um local centralizado. Use essas informações para monitorar o desempenho da consulta, identificar problemas e gerenciar seus fluxos de trabalho automatizados de análise de registros.

Console

Para visualizar consultas agendadas (console)

1. Abra o console do CloudWatch Logs em https://us-east-1.console.aws.amazon.com/cloudwatch/home? region=us-east-1 #logs. V2:logs-insights
2. No console do CloudWatch Logs, escolha Consulta agendada, Exibir consultas agendadas.

AWS CLI

Para listar consultas agendadas (AWS CLI)

- Use o `list-scheduled-queries` comando para listar todas as consultas agendadas:

```
aws logs list-scheduled-queries --max-results 10
```

API

Para listar consultas agendadas (API)

- Use a `ListScheduledQueries` ação para recuperar todas as consultas agendadas:

```
{
    "maxResults": 10
}
```

O cabeçalho da página de consultas agendadas mostra o número total de consultas agendadas em sua conta, ajudando você a monitorar seu uso e gerenciar seus fluxos de trabalho automatizados de análise de registros com eficiência.

Visualizando o histórico de execução de consultas agendadas

Use o histórico de execução para monitorar o desempenho de suas consultas agendadas e solucionar quaisquer problemas com a execução da consulta ou a entrega de resultados.

O histórico de execução mostra o status de cada consulta executada, incluindo execuções bem-sucedidas, falhas e resultados do processamento de destino. Você pode usar essas informações para identificar padrões, diagnosticar problemas e verificar se suas consultas estão sendo executadas conforme o esperado.

Console

Para visualizar o histórico de execução (console)

1. No console do CloudWatch Logs, escolha Consulta agendada, Exibir consultas agendadas.
2. Selecione a consulta agendada que você deseja examinar.
3. Escolha a guia Execution history (Histórico de execução).

AWS CLI

Para visualizar o histórico de execução (AWS CLI)

1. Use o `get-scheduled-query-history` comando para recuperar o histórico de execução de uma consulta agendada:

```
aws logs get-scheduled-query-history \
    --identifier "DailyErrorMonitoring" \
    --start-time 1743379200 \
    --end-time 1743465600 \
    --max-results 10
```

2. Para filtrar por status de execução, adicione o `--execution-statuses` parâmetro:

```
aws logs get-scheduled-query-history \
    --identifier "DailyErrorMonitoring" \
    --start-time 1743379200 \
    --end-time 1743465600 \
    --max-results 1 \
    --execution-statuses "SUCCEEDED"
```

API

Para visualizar o histórico de execução (API)

- Use a `GetScheduledQueryHistory` ação para recuperar o histórico de execução:

```
{
    "identifier": "DailyErrorMonitoring",
    "startTime": 1743379200,
    "endTime": 1743465600,
    "maxResults": 10,
    "executionStatuses": ["SUCCEEDED", "FAILED"]
}
```

O histórico de execução exibe:

- Status de execução - Em execução, concluído, com falha, tempo limite ou InvalidQuery
- Hora de disparo - Quando a consulta foi executada

- Destinos - Status de processamento para cada destino configurado EventBridge, incluindo S3 e e tabelas de pesquisa
- Mensagens de erro - Detalhes sobre qualquer falha na execução da consulta ou no processamento do destino

Atualizando uma consulta agendada

Modifique sua configuração de consulta agendada para alterar a sequência de caracteres, o cronograma, os destinos ou a função de execução da consulta à medida que seus requisitos evoluem.

Você pode atualizar qualquer aspecto de uma consulta agendada, incluindo a cadeia de caracteres da consulta, a expressão do cronograma, os destinos e a função de execução. As alterações entram em vigor imediatamente para futuras execuções.

Console

Para atualizar uma consulta agendada (console)

1. No console do CloudWatch Logs, escolha Consulta agendada, Exibir consultas agendadas.
2. Selecione a consulta agendada que você deseja atualizar.
3. Escolha Editar.
4. Modifique a configuração conforme necessário.
5. Escolha Salvar alterações.

AWS CLI

Para atualizar uma consulta agendada (AWS CLI)

- Use o `update-scheduled-query` comando para modificar uma consulta agendada existente:

```
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

Para atualizar uma consulta agendada (API)

1. Use a `UpdateScheduledQuery` ação para modificar a configuração da consulta agendada:

```
{
    "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. Para atualizar vários parâmetros de configuração de uma só vez:

```
{
    "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"
        }
    }
}
```

Configurando destinos do S3 para consultas agendadas

Configure o Amazon S3 como destino para armazenar os resultados da consulta agendada como arquivos JSON para retenção e análise de longo prazo.

Ao usar o Amazon S3 como destino, os resultados da consulta são armazenados como arquivos JSON no bucket e no prefixo especificados. Essa opção é ideal para arquivar resultados, realizar análises em lote ou integrar-se a outros AWS serviços que processam dados do S3.

Você pode entregar os resultados da consulta para um bucket do Amazon S3 na mesma AWS conta da consulta agendada ou para um bucket em uma AWS conta diferente. Opcionalmente, você também pode criptografar os resultados da consulta usando uma AWS KMS chave gerenciada pelo cliente (). SSE-KMS

Entrega de resultados para um bucket do Amazon S3 na mesma conta

Quando o bucket Amazon S3 de destino está na mesma AWS conta da consulta agendada, você pode navegar e selecionar o bucket diretamente do console.

Para configurar um destino do Amazon S3 (console) com a mesma conta

1. Na seção Publicar resultados da consulta no S3, para bucket do S3, selecione Esta conta.
2. Para o URI do Amazon S3, insira o bucket e o prefixo do Amazon S3 em que os resultados serão armazenados (por exemplo,`s3://my-bucket/query-results/`) ou escolha Browse Amazon S3 para navegar e selecionar um local existente do Amazon S3.
3. (Opcional) Para criptografar os resultados com uma AWS KMS chave gerenciada pelo cliente, insira o ARN da AWS KMS chave no campo ARN da chave KMS. A chave deve estar na mesma AWS região do bucket de destino do Amazon S3. Se você não especificar uma AWS KMS chave, as configurações de criptografia padrão do bucket serão aplicadas.
4. Na seção Função do IAM para publicar os resultados da consulta no Amazon S3, escolha Auto-create uma nova função com permissões padrão para configurar automaticamente as permissões necessárias ou escolha Usar uma função existente para selecionar uma função existente do IAM com as políticas necessárias.

A função IAM de entrega de destino exige as seguintes permissões:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

Entrega de resultados para um bucket do Amazon S3 em outra conta

Você pode entregar resultados de consultas agendadas para um bucket do Amazon S3 em uma AWS conta diferente. Ao usar um bucket de contas cruzadas, você deve fornecer o URI do Amazon S3 e o ID da conta do proprietário do bucket.

Para configurar um destino do Amazon S3 (console) entre contas

1. Na seção Publicar resultados da consulta no S3, para bucket S3, selecione Outra conta e forneça o ID da conta proprietária do bucket como entrada.
2. Para o URI do Amazon S3, insira o URI completo do Amazon S3 do bucket de destino e o prefixo na outra conta (por exemplo,). `s3://cross-account-bucket/query-results/`
3. (Opcional) Para criptografar os resultados com uma AWS KMS chave gerenciada pelo cliente, insira o ARN da AWS KMS chave no campo ARN da chave KMS. A chave deve estar na mesma AWS região do bucket de destino do Amazon S3.
4. Na seção Função do IAM para publicar os resultados da consulta no Amazon S3, escolha Auto-create uma nova função com permissões padrão para configurar automaticamente as permissões necessárias ou escolha Usar uma função existente para selecionar uma função existente do IAM com as políticas necessárias.

Cross-account a entrega requer permissões de ambos os lados. A função IAM de entrega de destino na conta de origem exige as seguintes permissões:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::cross-account-bucket/prefix/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                }
            }
        }
    ]
}
```

A política de bucket do Amazon S3 na conta de destino deve conceder permissão à função IAM da conta de origem para escrever objetos:

```
{
    "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/*"
        }
    ]
}
```

Criptografando resultados com um cliente gerenciado AWS KMS key

Opcionalmente, você pode especificar uma AWS KMS chave gerenciada pelo cliente para criptografar os resultados da consulta entregues ao Amazon S3 usando. SSE-KMS A AWS KMS chave pode estar na mesma conta da consulta agendada ou em uma conta diferente.

Quando você especifica uma AWS KMS chave, a consulta agendada usa essa chave para criptografar os resultados via. SSE-KMS Quando você não especifica uma AWS KMS chave, as configurações de criptografia padrão do bucket se aplicam. Se o bucket estiver configurado com

SSE-KMS criptografia padrão usando uma chave gerenciada pelo cliente, a função IAM de entrega de destino ainda deverá ter `kms:GenerateDataKey` permissão nessa chave.

A função IAM de entrega de destino exige `kms:GenerateDataKey` permissão na AWS KMS chave. O exemplo a seguir mostra as permissões necessárias para um destino do Amazon S3 com uma AWS KMS chave gerenciada pelo cliente:

```
{
    "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*"
                }
            }
        }
    ]
}
```

Quando a AWS KMS chave está em uma conta diferente da função IAM de entrega de destino, a política de chaves na conta proprietária da AWS KMS chave deve conceder explicitamente o acesso à função:

```
{
    "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**
>
> Quando a AWS KMS chave e a função IAM de entrega de destino estão na mesma conta, a política de identidade do IAM por si só é suficiente se a política de AWS KMS chaves incluir a declaração raiz padrão “Habilitar políticas do IAM”. Uma concessão explícita de política de AWS KMS chaves só é necessária se a política de chaves não delegar ao IAM.

A função do IAM para publicar os resultados da consulta no Amazon S3 deve ser configurada separadamente da função do IAM para a execução programada da consulta. Essa separação permite um controle de acesso refinado, em que a função de execução pode executar consultas, enquanto a função Amazon S3 lida especificamente com a entrega dos resultados. Ambas as funções devem incluir uma política de confiança que permita que o serviço CloudWatch Logs (`logs.amazonaws.com`) assuma a função.

Configurando destinos da tabela de pesquisa para consultas agendadas

Configure uma tabela de pesquisa como destino para atualizar automaticamente uma tabela de pesquisa com os resultados da consulta agendada. Em cada execução agendada, o CloudWatch Logs preenche a tabela de pesquisa especificada com os resultados da consulta, mantendo a tabela atualizada com seus dados de registro mais recentes. Em seguida, você pode referenciar a tabela com o `lookup` comando em outras consultas de log. Para obter mais informações sobre tabelas de pesquisa, consultelookup.

Se a tabela de pesquisa não existir, o CloudWatch Logs a criará na primeira execução bem-sucedida. Cada execução subsequente é uma operação de substituição completa — todo o conteúdo da tabela existente é substituído pelos novos resultados da consulta.

A configuração da tabela de pesquisa inclui as seguintes configurações:

- Nome da tabela — O nome da tabela de pesquisa a ser criada ou atualizada. O nome pode conter somente caracteres alfanuméricos e sublinhados.
- Função de entrega — O ARN da função do IAM que concede permissões ao CloudWatch Logs para criar ou atualizar a tabela de pesquisa com os resultados da consulta.
- Descrição — (Opcional) Uma descrição da tabela de pesquisa.
- AWS KMS key — (Opcional) O ARN da AWS KMS chave a ser usada para criptografar os dados da tabela de pesquisa. Se você não especificar uma chave, os dados serão criptografados com uma chave AWS de propriedade.
- Tags — Key-value pares (opcionais) a serem associados à tabela de pesquisa. As tags são aplicadas somente quando a tabela é criada inicialmente.

A função IAM de entrega de destino exige as seguintes permissões e deve incluir uma política de confiança que permita que o serviço CloudWatch Logs (`logs.amazonaws.com`) assuma a função:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLookupTable",
```

```
                "logs:UpdateLookupTable",
                "logs:GetQueryResults"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Os resultados de destino da tabela de pesquisa estão sujeitos às mesmas cotas da tabela de pesquisa que as tabelas que você cria diretamente, como o número máximo de tabelas de pesquisa por Região da AWS conta. Se uma execução agendada falhar em atualizar a tabela, o conteúdo da tabela existente permanecerá inalterado e a falha aparecerá no histórico de execução da consulta agendada.

Solução de problemas de consultas agendadas

Use esses tópicos de solução de problemas para resolver problemas comuns com consultas agendadas.

A execução da consulta falha com erros de permissão

Resolva erros de permissão que impedem que consultas agendadas sejam executadas ou entreguem resultados aos destinos.

Os erros de permissão ocorrem quando a função de execução não tem as permissões necessárias para ler dos grupos de registros ou gravar nos recursos de destino.

Para resolver erros de permissão

1. Verifique se a função de execução tem `logs:StartQuerylogs:GetQueryResults`, e `logs:DescribeLogGroups` permissões para os grupos de log de destino.
2. Certifique-se de que a função de execução tenha permissões de gravação para os recursos de destino (como `s3:PutObject` para buckets do S3).
3. Verifique se a política de confiança permite que CloudWatch os Logs assumam a função de execução. A função deve confiar no principal do serviço de registros (`logs.amazonaws.com`) em sua política de confiança.

As causas comuns incluem a falta de permissões do IAM, ARNs de recursos incorretos na política ou problemas de configuração da política de confiança.

Para evitar erros de permissão, use o princípio do menor privilégio ao criar funções de execução e testar permissões antes de implantar consultas agendadas na produção.

O tempo limite da consulta

Resolva os erros de tempo limite que ocorrem quando as consultas agendadas excedem o limite máximo de tempo de execução.

O tempo limite de consulta ocorre quando a consulta leva mais de 60 minutos para processar o intervalo de dados especificado, geralmente devido a grandes conjuntos de dados ou lógica de consulta complexa.

Para resolver erros de tempo limite

1. Reduza o intervalo de tempo diminuindo a diferença da hora de início para processar menos dados por execução.
2. Otimize a consulta adicionando filtros no início da consulta para reduzir a quantidade de dados processados. Use índices de filtro para reduzir o tamanho da varredura de dados.
3. Considere dividir consultas complexas em consultas mais simples e focadas.

As causas comuns incluem consultar grandes intervalos de tempo, processar grupos de registros de alto volume ou usar agregações complexas sem a filtragem adequada.

Para evitar tempos limite, teste as consultas manualmente no CloudWatch Logs Insights com o volume de dados esperado e otimize o desempenho antes de agendar.

O processamento de destino falha

Resolva falhas que ocorrem quando os resultados da consulta agendada não podem ser entregues aos destinos configurados.

As falhas de processamento de destino acontecem quando o bucket ou barramento de EventBridge eventos do Amazon S3 de destino está inacessível ou configurado incorretamente.

Para resolver falhas em que os resultados da consulta não estão sendo publicados no destino

1. Verifique se o bucket especificado do Amazon S3 existe e está acessível.

2. Verifique a configuração de destino para ver se os URIs estão corretos.
3. Certifique-se de que a função de execução tenha as permissões necessárias para gravar no destino.

As causas comuns incluem recursos de destino excluídos ou renomeados, URIs de destino incorretos ou problemas de conectividade de rede.

Para evitar falhas no destino, valide regularmente as configurações de destino e monitore a disponibilidade dos recursos de destino.

Erros de consulta inválidos

Resolva erros de sintaxe e lógica em cadeias de consulta agendadas que impedem a execução bem-sucedida.

Erros de consulta inválidos ocorrem quando a string de consulta contém erros de sintaxe, faz referência a campos inexistentes ou usa recursos de linguagem de consulta não suportados.

Para resolver erros de consulta inválidos

1. Teste sua consulta manualmente no CloudWatch Logs Insights para verificar a sintaxe e a lógica.
2. Verifique se todos os campos de registro referenciados existem em seus grupos de registro de destino.
3. Verifique se os recursos de linguagem de consulta que você está usando são compatíveis com consultas agendadas.

As causas comuns incluem erros de digitação em nomes de campo, sintaxe de consulta incorreta ou uso de recursos de consulta não suportados no ambiente de execução agendada.

Para evitar erros de consulta inválidos, sempre teste as consultas interativamente antes de agendar e use os recursos de descoberta de campo para verificar os nomes dos campos.

Erros de simultaneidade de consultas

Há alguns pontos importantes mencionados abaixo que você deve ter em mente quando os erros de simultaneidade são vistos, pois as consultas agendadas usam a mesma cota das consultas de

insights do Cloudwatch Logs. É recomendável distribuir suas agendas para evitar atingir o limite de simultaneidade.

- Cota: você pode executar até 100 consultas simultâneas do CloudWatch Logs Insights por conta. AWS
- Painéis: as consultas adicionadas aos CloudWatch painéis também contam para esse limite de simultaneidade, pois são executadas quando o painel é carregado ou atualizado.
- OpenSearch Serviço PPL/SQL: você pode executar até 15 consultas OpenSearch PPL ou OpenSearch SQL simultâneas por conta. AWS
- Cross-account consultas: a cota de simultaneidade se aplica a consultas de conta única e cruzada. Ao usar a observabilidade CloudWatch entre contas, as consultas iniciadas em uma conta de monitoramento em uma conta de origem vinculada também contam para o limite de simultaneidade da conta de monitoramento.
- Grupos de registros de acesso ocasionais: para grupos de registros na classe de registro de acesso não frequente, o número máximo de consultas simultâneas do Logs Insights é limitado a cinco.

Detecção de anomalias de log

Você pode detectar anomalias em seus dados de registro de duas maneiras: criando um detector de anomalias de log para monitoramento contínuo ou usando o `anomaly detection`comando nas consultas do CloudWatch Logs Insights para análise sob demanda.

Um detector de anomalias de log verifica os eventos de logs ingeridos no grupo de logs e encontra anomalias nos dados de log automaticamente. A detecção de anomalias usa machine learning e reconhecimento de padrões para estabelecer linhas de base de conteúdo típico dos logs. Para análise sob demanda, você pode usar o `anomaly detection` comando nas consultas do CloudWatch Logs Insights para identificar padrões incomuns em dados de séries temporais. Para obter mais informações sobre detecção de anomalias baseada em consulta, consulte Usando a detecção de anomalias no CloudWatch Logs Insights.

Depois que você cria um detector de anomalias para um grupo de logs, ele é treinado usando as últimas duas semanas de eventos de log do grupo de logs para treinamento. O período de treinamento pode levar até 15 minutos. Depois que o treinamento é concluído, ele começa a analisar os registros recebidos para identificar anomalias, e as anomalias são exibidas no console de CloudWatch registros para você examinar.

CloudWatch O reconhecimento de padrões de registros extrai padrões de registros identificando conteúdo estático e dinâmico em seus registros. Os padrões são úteis para analisar grandes conjuntos de logs porque um grande número de eventos de log geralmente pode ser resumido a apenas alguns padrões.

Por exemplo, veja o exemplo de três eventos de log a seguir.

```
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
```

No exemplo anterior, todos os três eventos de log seguem um único padrão:

```
<Date-1> <Time-2> [INFO] Calling DynamoDB to store for resource id <ResourceID-3>
```

Os campos dentro de um padrão são chamados de tokens. Os campos que variam dentro de um padrão, como um ID de solicitação ou um timestamp, são chamados de tokens dinâmicos. Cada valor diferente encontrado para um token dinâmico é chamado de valor de token.

Se o CloudWatch Logs puder inferir o tipo de dados que um token dinâmico representa, ele exibirá o token como<*string*-*number*>. *string*É uma descrição do tipo de dados que o token representa. *number*Mostra onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

CloudWatch O Logs atribui a parte da string do nome com base na análise do conteúdo dos eventos de log que a contêm.

Se o CloudWatch Logs não conseguir inferir o tipo de dados que um token dinâmico representa, ele exibe o token como **<**Token- *number* **>** e *number* indica onde esse token aparece no padrão, em comparação com os outros tokens dinâmicos.

Exemplos comuns de tokens dinâmicos incluem códigos de erro, endereços IP, registros de data e hora e solicitações. IDs

A detecção de anomalias de log usa esses padrões para encontrar anomalias. Após o período de treinamento de modelo do detector de anomalias, os logs são avaliados em relação às tendências conhecidas. O detector de anomalias sinaliza flutuações significativas como anomalias.

Este capítulo descreve como habilitar a detecção de anomalias, visualizar anomalias, criar alarmes para detectores de anomalias de log e as métricas que os detectores de anomalias de log publicam. Também descreve como criptografar o detector de anomalias e seus resultados com. AWS Key Management Service

A criação de detectores de anomalias de log não é cobrada.

Gravidade e prioridade de anomalias e padrões

A cada anomalia encontrada por um detector de anomalias de log é atribuída uma prioridade. A cada padrão encontrado é atribuída uma gravidade.

- A prioridade é calculada automaticamente com base no nível de gravidade do padrão e na quantidade de desvio dos valores esperados. Por exemplo, se um determinado valor de token aumentar repentinamente em 500%, essa anomalia pode ser designada como tendo prioridade `HIGH`, mesmo que sua gravidade seja `NONE`.
- A gravidade é baseada apenas nas palavras-chave encontradas nos padrões, como `FATAL`, `ERROR` e `WARN`. Se nenhuma dessas palavras-chave for encontrada, a gravidade de um padrão será marcada como `NONE`.

Tempo de visibilidade das anomalias

Ao criar um detector de anomalias, você especifica o período máximo de visibilidade das anomalias para ele. Esse é o número de dias em que a anomalia é exibida no console e é retornada pela operação da ListAnomaliesAPI. Depois de decorrido esse tempo para uma anomalia, se ela continuar ocorrendo, será automaticamente aceita como um comportamento normal e o modelo do detector de anomalias não a sinalizará mais como uma anomalia.

Se você não ajustar o tempo de visibilidade quando criar um detector de anomalias, o padrão de 21 dias será usado.

Supressão de uma anomalia

Depois que uma anomalia é encontrada, você pode optar por suprimi-la temporária ou permanentemente. A supressão de uma anomalia faz com que o detector de anomalias pare de sinalizar essa ocorrência como uma anomalia durante o tempo que você especificar. Ao suprimir uma anomalia, você pode optar por suprimir apenas essa anomalia específica ou todas as anomalias relacionadas ao padrão no qual a anomalia foi encontrada.

Você continua podendo visualizar as anomalias suprimidas no console. Você também pode escolher deixar de suprimi-las.

Perguntas frequentes

AWS Usa meus dados para treinar algoritmos de aprendizado de máquina para AWS uso ou para outros clientes?

Não. O modelo de detecção de anomalias criado pelo treinamento é baseado nos eventos de registro em um grupo de registros e é usado somente nesse grupo de registros e nessa AWS conta.

Que tipos de eventos de log funcionam bem com a detecção de anomalias?

A detecção de anomalias de log é adequada para: logs de aplicações e outros tipos de log em que a maioria das entradas de log se enquadra nos padrões típicos. Grupos de logs com eventos que contêm um nível de log ou palavras-chave de gravidade, como INFO, ERROR e DEBUG, são especialmente adequados para a detecção de anomalias de log.

A detecção de anomalias de log não é adequada para: Registrar eventos com estruturas JSON extremamente longas, como CloudTrail Logs. A análise de padrões só analisa os primeiros 1.500 caracteres de uma linha de log, portanto, todos os caracteres além desse limite são ignorados.

Logs de auditoria ou acesso, como logs de fluxo de VPC, também terão menos sucesso com a detecção de anomalias. A detecção de anomalias serve para encontrar problemas de aplicações, assim sendo, pode não ser adequada para anomalias de rede ou acesso.

Para ajudá-lo a determinar se um detector de anomalias é adequado para um determinado grupo de CloudWatch registros, use a análise de padrões de registros para encontrar o número de padrões nos eventos de registro no grupo. Se a quantidade de padrões não passar de cerca de 300, a detecção de anomalias poderá funcionar bem. Para obter mais informações sobre análise de padrões, consulte Análise de padrões.

O que é sinalizado como anomalia?

As seguintes ocorrências podem fazer um evento de log ser sinalizado como uma anomalia:

- Um evento de log com um padrão nunca visto antes no grupo de logs.
- Uma variação significativa de um padrão conhecido.
- Um novo valor para um token dinâmico que tem um conjunto discreto de valores usuais.
- Uma grande alteração no número de ocorrências de um valor para um token dinâmico.

Embora todos os itens anteriores possam estar sinalizados como anomalias, nem todos significam que a aplicação está funcionando mal. Por exemplo, higher-than-usual vários valores de `200` sucesso podem ser sinalizados como uma anomalia. Em casos como esse, você pode considerar suprimir essas anomalias que não indicam problemas.

O que acontece com os dados confidenciais que estão sendo mascarados?

Todas as partes de eventos de log que são mascaradas como dados confidenciais não são varridas para detectar anomalias. Para obter mais informações, consulte Help protect sensitive log data with masking.

Usando a detecção de anomalias no CloudWatch Logs Insights

Além de criar detectores de anomalias de log para monitoramento contínuo, você também pode usar o `anomaly` comando nas consultas do CloudWatch Logs Insights para identificar padrões incomuns em seus dados de log sob demanda. Esse comando amplia a funcionalidade `pattern` existente e utiliza machine learning para detectar cinco tipos de anomalias, incluindo alterações de frequência de padrões, novos padrões e variações de tokens.

O comando `anomaly` é útil principalmente para:

- Análise ad hoc de dados históricos de log para identificar padrões incomuns
- Investigar períodos específicos quanto a comportamento anômalo
- Monitorar aplicativos, como funções Lambda, em busca de problemas de execução

Para obter mais informações sobre como usar o comando `anomaly` na suas consultas, confira anomalia.

Essa detecção de anomalias baseada em consultas complementa os detectores contínuos de anomalias descritos nas seções a seguir, oferecendo recursos de monitoramento em tempo real e análise sob demanda.

Habilitar a detecção de anomalias em um grupo de logs

Use as etapas a seguir para usar o CloudWatch console para criar um detector de anomalias de log que escaneia um grupo de registros em busca de anomalias.

Você também pode criar detectores de anomalias por meio de um programa. Para obter mais informações, consulte CreateLogAnomalyDetector.

Para criar um detector de anomalias de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Logs, Anomalias de log.
3. Escolha Criar detector de anomalias.
4. Selecione o grupo de logs para o qual esse detector de anomalias será criado.
5. Insira um nome para o detector em Nome do detector de anomalias.
6. (Opcional) Altere a Frequência de avaliação do padrão de 5 minutos. Defina esse valor de acordo com a frequência com que o grupo de logs recebe novos logs. Por exemplo, se o grupo de logs receber novos eventos de log em lotes a cada 10 minutos, pode ser apropriado definir a frequência de avaliação como 15 minutos.
7. (Opcional) Para configurar o detector de anomalias para só procurar anomalias em eventos de log que contenham certas palavras ou strings, escolha Filtrar padrões.

   Em seguida, insira um padrão em Padrão de filtro de detecção de anomalias. Para obter mais informações sobre a sintaxe dos padrões, consulte Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura, filtros de eventos de log e Live Tail..

(Opcional) Para testar o padrão de filtro, insira algumas mensagens de log em Mensagens de eventos de log e selecione Testar padrão.

8. (Opcional) Para alterar o período de visibilidade da anomalia do padrão ou para associar uma AWS KMS chave a esse detector de anomalias, escolha Configuração avançada.

   a. Para alterar o período de visibilidade de anomalias do padrão, insira um novo valor em Período máximo de visibilidade de anomalias (dias).

   b. Para associar uma AWS KMS chave a esse detector de anomalias, insira o ARN no ARN da chave KMS. Se você atribuir uma chave, as informações de anomalias encontradas por esse detector serão criptografadas em repouso com essa chave. Os usuários devem ter permissões para essa chave e para que o detector de anomalias recupere as informações sobre as anomalias encontradas.

      Você também deve garantir que o responsável pelo serviço de CloudWatch registros tenha permissão para usar a chave. Para obter mais informações, consulte Criptografe um detector de anomalias e seus resultados com AWS KMS.

9. Escolha Habilitar a detecção de anomalias.

   O detector de anomalias é criado e começa a treinar seu modelo com base nos eventos de log que o grupo de logs está ingerindo. Após cerca de 15 minutos, a detecção de anomalias está ativa e começa a encontrar e exibir anomalias.

Visualizar as anomalias que foram encontradas

Depois de criar um ou mais detectores de anomalias de log, você pode usar o CloudWatch console para ver as anomalias que eles encontraram.

Você pode visualizar anomalias por meio de um programa. Para obter mais informações, consulte ListAnomalies.

Para visualizar as anomalias encontradas por todos os detectores de anomalias de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Logs, Anomalias de log.

A tabela Anomalias de log é exibida. O número no alto, ao lado de Anomalias de log, mostra quantas anomalias de log estão listadas na tabela. Cada linha da tabela exibe as seguintes informações:

- A coluna Anomalia exibe um breve resumo da anomalia. Esses resumos são gerados pelo CloudWatch Logs.
- A prioridade da anomalia. A prioridade é calculada automaticamente com base na quantidade de alteração nos eventos de log, palavras-chave como `Exception` encontradas em um evento de log etc.
- O padrão de log no qual a anomalia se baseia. Para obter mais informações sobre padrões, consulte Detecção de anomalias de log.
- Tendência dos logs de anomalias exibe um histograma mostrando o volume de logs que seguem ao padrão.
- Hora da última detecção exibe a última vez que essa anomalia foi encontrada.
- Hora da última detecção exibe quando essa anomalia foi encontrada pela última vez.
- Detector de anomalias exibe o nome do grupo de logs que contém os eventos de log relacionados a essa anomalia. Você pode escolher esse nome para ver a página de detalhes do grupo de logs.

3. Para inspecionar melhor uma anomalia, escolha o botão de opção na linha correspondente.

   O painel Inspeção de padrões aparece e exibe o seguinte:

   - O padrão em que a anomalia se baseia. Selecione um token no padrão para analisar os valores desse token.
   - Um histograma mostrando o número de ocorrências da anomalia durante o intervalo de tempo consultado.
   - A guia Exemplos de log exibe alguns eventos de log que fazem parte da anomalia.
   - A guia Valores do token exibe os valores do token dinâmico selecionado, se você tiver selecionado um.

> **Note**
>
> No máximo de 10 valores de token são capturados para cada token. A contagem de tokens pode não ser precisa. CloudWatch O Logs usa um contador probabilístico para gerar a contagem de tokens, não o valor absoluto.

4. Para suprimir uma anomalia, escolha o botão de opção na linha correspondente e faça o seguinte:

   a. Escolha Ações, Suprimir anomalia.

   b. Depois, especifique por quanto tempo deseja que a anomalia seja suprimida.

   c. Para suprimir todas as anomalias relacionadas a esse padrão, selecione Suprimir padrão.

   d. Escolha Suprimir anomalia.

Para visualizar as anomalias encontradas em um único grupo de logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Logs, Grupos de logs.
3. Escolha o nome de um grupo de logs e depois escolha a guia Detecção de anomalias.

   A tabela Detecção de anomalias é exibida. O número no alto, ao lado de Anomalias de log, mostra quantas anomalias de log estão listadas na tabela. Cada linha da tabela exibe as seguintes informações:

   - A coluna Anomalia exibe um breve resumo da anomalia. Esses resumos são gerados pelo CloudWatch Logs.
   - A prioridade da anomalia. A prioridade é calculada automaticamente com base na quantidade de alteração nos eventos de log, palavras-chave como `Exception` encontradas em um evento de log etc.
   - O padrão de log no qual a anomalia se baseia. Para obter mais informações sobre padrões, consulte Detecção de anomalias de log.
   - Tendência dos logs de anomalias exibe um histograma mostrando o volume de logs que seguem ao padrão.
   - Hora da última detecção exibe a última vez que essa anomalia foi encontrada.
   - Hora da última detecção exibe quando essa anomalia foi encontrada pela última vez.

4. Para inspecionar melhor uma anomalia, escolha o botão de opção na linha correspondente.

   O painel Inspeção de padrões aparece e exibe o seguinte:

   - O padrão em que a anomalia se baseia. Selecione um token no padrão para analisar os valores desse token.
   - Um histograma mostrando o número de ocorrências da anomalia durante o intervalo de tempo consultado.
   - A guia Exemplos de log exibe alguns eventos de log que fazem parte da anomalia.
   - A guia Valores do token exibe os valores do token dinâmico selecionado, se você tiver selecionado um.

   > **Note**
   >
   > No máximo de 10 valores de token são capturados para cada token. A contagem de tokens pode não ser precisa. CloudWatch O Logs usa um contador probabilístico para gerar a contagem de tokens, não o valor absoluto.

5. Para suprimir uma anomalia, escolha o botão de opção na linha correspondente e faça o seguinte:

   a. Escolha Ações, Suprimir anomalia.
   b. Depois, especifique por quanto tempo deseja que a anomalia seja suprimida.
   c. Para suprimir todas as anomalias relacionadas a esse padrão, selecione Suprimir padrão.
   d. Escolha Suprimir anomalia.

Criar alarmes em detectores de anomalias de log

Você pode criar um alarme para um detector de anomalias de log em um grupo de logs. Você pode especificar que o alarme entre no estado `ALARM` quando um determinado número de anomalias for encontrado no grupo de logs durante um período especificado. Você também pode usar filtros para que apenas anomalias com determinadas prioridades sejam contadas pelo alarme.

Para criar um alarme para um detector de anomalias de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Anomalias de log.

A tabela de detectores de anomalias de log é exibida.

3. Escolha o botão de opção do detector de anomalias para o qual você deseja definir o alarme e escolha Criar alarme.

   O assistente CloudWatch de criação de alarmes é exibido. O LogAnomalyDetectorcampo exibe o nome do detector de anomalias que você escolheu. O campo Nome da métrica é exibido AnomalyCount.

4. (Opcional) Para filtrar esse alarme por prioridade da anomalia, faça uma das seguintes alternativas:

   - Para que o alarme conte somente anomalias de alta prioridade, digite para. **`HIGH`** LogAnomalyPriority
   - Para que o alarme conte somente anomalias de alta e média prioridade, digite para. **`MEDIUM`** LogAnomalyPriority

   Para obter mais informações sobre níveis de prioridade, consulte Gravidade e prioridade de anomalias e padrões.

5. Escolha entre usar um limite de detecção de anomalias estático ou métrico para o alarme. Essa seleção determina como o limite do alarme é definido. Um limite estático significa que o limite de alarme é um número estático e constante que você escolhe. Um limite de detecção de anomalias significa que CloudWatch determina uma faixa de valores usuais, e o alarme é acionado se a contagem real ultrapassar o limite dessa banda. Você não precisa escolher Detecção de anomalias para um alarme de detecção de anomalias de log. Para obter mais informações sobre a detecção métrica de anomalias, consulte Usando a detecção de CloudWatch anomalias.

6. Para Whenever ***`your-metric-name`*** is... , escolha Maior, Maior/Igual, Menor/Igual ou Menor. Em seguida, para então …, especifique um número para o valor limite. O alarme entrará no estado `ALARM` se o detector de anomalias encontrar mais do que esse número de alarmes durante um tempo especificado por Período.

7. Escolha Additional configuration (Configuração adicional). Em Datapoints to alarm (Pontos de dados para alarme), especifique quantos períodos de avaliação (pontos de dados) devem estar no estado `ALARM` para disparar o alarme. Se os dois valores forem correspondentes, você criará um alarme que passa para o estado `ALARM` se esses períodos consecutivos estiverem violando.

   Para criar um alarme M de um alarme N, especifique um número para o primeiro valor que seja menor do que o segundo valor. Para obter mais informações, consulte Avaliar um alarme.

8. Em Missing data treatment (Tratamento de dados ausentes), escolha como deseja que o alarme se comporte quando alguns pontos de dados estiverem ausentes. Para obter mais informações, consulte Configurando como os CloudWatch alarmes tratam os dados perdidos.
9. Escolha Próximo.
10. Em Notificação, escolha Adicionar notificação e especifique um tópico do Amazon SNS a ser notificado quando o alarme passar para o estado `ALARM`, `OK` ou `INSUFFICIENT_DATA`.

    a. (Opcional) Para enviar várias notificações para o mesmo estado de alarme ou para diferentes estados de alarme, escolha Add notification (Adicionar notificação).

    > **Note**
    >
    > Recomendamos configurar o alarme para executar ações quando entrar no estado Dados insuficientes, além de para quando entrar no estado Alarme. Isso ocorre porque muitos problemas com a função do Lambda que se conecta à fonte de dados podem fazer com que o alarme transite para Dados insuficientes.

    b. (Opcional) Para não enviar notificações do Amazon SNS, escolha Remover.
11. (Opcional) Se você quiser que seu alarme execute ações para Amazon EC2 Auto Scaling, Amazon EC2, tickets ou AWS Systems Manager, escolha o botão apropriado e especifique o estado e a ação do alarme.

    > **Note**
    >
    > O alarme pode executar ações do Systems Manager somente ao entrar no estado `ALARM`. Para obter informações sobre as ações do Systems Manager, consulte Configuração CloudWatch para criar OpsItems e Criação de incidentes.
12. Escolha Próximo.
13. Em Nome e descrição, insira um nome e uma descrição para o alarme e selecione Próximo. O nome deve conter somente caracteres UTF-8, e não poderá conter caracteres de controle ASCII. A descrição pode incluir a formatação markdown, que é exibida somente na guia Detalhes do alarme no CloudWatch console. O markdown pode ser útil para adicionar links para runbooks ou outros recursos internos.

> **Tip**
>
> O nome do alarme deve conter somente caracteres UTF-8. Ele não pode conter caracteres de controle ASCII.

14. Em Pré-visualizar e criar, confirme se as informações e condições do seu alarme estão corretas e escolha Criar alarme.

Métricas publicadas por detectores de anomalias de log

CloudWatch O Logs publica a AnomalyCountmétrica em CloudWatch métricas. Essa métrica e publicada no namespace `AWS/Logs`.

A AnomalyCountmétrica é publicada com as seguintes dimensões:

- LogAnomalyDetector— O nome do detector de anomalias
- LogAnomalyPriority— O nível de prioridade da anomalia

Criptografe um detector de anomalias e seus resultados com AWS KMS

Os dados do detector de anomalias são sempre criptografados nos CloudWatch registros. Por padrão, o CloudWatch Logs usa criptografia do lado do servidor para os dados em repouso. Como alternativa, você pode usar AWS Key Management Service essa criptografia. Se você fizer isso, a criptografia será feita usando uma AWS KMS chave. O uso da criptografia do AWS KMS é habilitado ao nível do detector de anomalias, associando uma chave do KMS a um detector de anomalias.

> **Important**
>
> CloudWatch O Logs suporta somente chaves KMS simétricas. Não use uma chave assimétrica para criptografar os dados em seus grupos de logs. Para obter mais informações, consulte Usar chaves simétricas e assimétricas.

Limites

- Para executar as etapas a seguir, é necessário ter as seguintes permissões: `kms:CreateKey`, `kms:GetKeyPolicy` e `kms:PutKeyPolicy`.
- Depois que você associa ou desassocia uma chave de um detector de anomalias, a operação pode levar até cinco minutos para ter efeito.
- Se você revogar o acesso do CloudWatch Logs a uma chave associada ou excluir uma chave KMS associada, seus dados criptografados no CloudWatch Logs não poderão mais ser recuperados.

Etapa 1: criar uma AWS KMS chave

Para criar uma chave do KMS, use o seguinte comando create-key:

```
aws kms create-key
```

A saída contém o ID de chave e o nome do recurso da Amazon (ARN) da chave. A seguir está um exemplo de saída:

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

Etapa 2: definir permissões na chave do KMS

Por padrão, todas AWS KMS as chaves são privadas. Somente o proprietário do recurso pode usá-la para criptografar e descriptografar dados. No entanto, o proprietário do recurso pode conceder permissões para acessar a chave do KMS a outros usuários e recursos. Com essa etapa, você concede permissão principal ao serviço de CloudWatch registros para usar a chave. Esse principal de serviço deve estar na mesma AWS região em que a chave KMS está armazenada.

Como prática recomendada, recomendamos que você restrinja o uso da chave KMS somente às AWS contas ou detectores de anomalias que você especificar.

Primeiro, salve a política padrão para sua chave KMS `policy.json` usando o seguinte get-key-policycomando:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

Abra o arquivo `policy.json` em um editor de texto e adicione a seção em negrito de uma das instruções a seguir. Separe a instrução existente da nova instrução com uma vírgula. Essas declarações usam `Condition` seções para aumentar a segurança da AWS KMS chave. Para obter mais informações, consulte AWS KMS chaves e contexto de criptografia.

A seção `Condition` deste exemplo limita o uso da chave do AWS KMS à conta especificada, mas ela pode ser usada para qualquer detector de anomalias.

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"
                }
            }
        }
    ]
}
```

Por fim, adicione a política atualizada usando o seguinte put-key-policycomando:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Etapa 3: associar uma chave do KMS a um detector de anomalias

Você pode associar uma chave KMS a um detector de anomalias ao criá-la no console ou usando o AWS CLI ou. APIs

Etapa 4: desassociar uma chave de um detector de anomalias

Depois que uma chave é associada a um detector de anomalias, você não pode mais atualizá-la. A única maneira de remover a chave é excluir o detector de anomalias e, em seguida, recriá-lo.

Solução de problemas com o CloudWatch Logs Live Tail

CloudWatch O Logs Live Tail ajuda você a solucionar incidentes rapidamente, visualizando uma lista de streaming de novos eventos de log à medida que são ingeridos. É possível visualizar, filtrar e realçar logs consumidos quase em tempo real, o que ajuda a detectar e resolver problemas rapidamente. Os logs podem ser filtrados com base em termos especificados, e você também pode realçar logs que contêm termos específicos para ajudar a encontrar rapidamente o que está procurando.

As sessões Live Tail acumulam custos por tempo de uso da sessão, por minuto. Para obter mais informações sobre preços, consulte a guia Logs em Amazon CloudWatch Pricing.

O Live Tail só é compatível com grupos de logs da classe de logs Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

As seções a seguir explicam como usar Live Tail no console e na AWS CLI. Você também pode iniciar uma sessão de Live Tail por meio de um programa. Para obter mais informações, consulte StartLiveTail. Para exemplos de SDK, consulte Iniciar uma sessão do Live Tail usando um AWS SDK.

Você também pode usar o Live Tail no AWS Toolkit for Visual Studio Code. Para iniciar uma sessão do Live Tail a partir da paleta de comandos do VS Code, consulte a seção Amazon CloudWatch Logs Live Tail do Guia do AWS Toolkit for Visual Studio Code usuário.

O recurso Live Tail está disponível em todas as AWS regiões comerciais. Ele não está disponível nas regiões da China nem nas regiões AWS GovCloud (EUA).

> **Note**
>
> A `StartLiveTail` API encaminha as solicitações usando a injeção de prefixo de host do SDK. As versões do SDK lançadas antes de 1º de abril de 2026 são roteadas para `streaming-logs.Region.amazonaws.com`, o que não é compatível com VPC endpoints. As versões do SDK lançadas em ou após 1º de abril de 2026 são roteadas para `streaming-logs.Region.amazonaws.com`, que são compatíveis com VPC endpoints. Para configurar um VPC endpoint para essa API, consulte Como criar um VPC endpoint para registros. CloudWatch

Inicie uma sessão do Live Tail usando o AWS CLI

O `start-live-tail` AWS CLI comando inicia uma sessão de streaming do Live Tail para um ou mais grupos de log em um terminal. Uma sessão do Live Tail pode durar até três horas. Se mais de 500 eventos de log por segundo corresponderem ao filtro, os eventos de log exibidos serão uma amostra do total de eventos de log, para fornecer uma experiência de rastreamento em tempo real. Para obter mais informações sobre o comando `start-live-tail`, consulte start-live-tail

Você pode usar o `start-live-tail` em dois modos:

- somente impressão: esse é o modo padrão
- interativo

somente impressão

No modo `print-only`, o fluxo de eventos de log é gerado no terminal. Novos eventos são adicionados no fim da lista a cada segundo, criando uma experiência de rastreamento quase em tempo real, similar a `tail -f` no Linux.

Para iniciar uma sessão do Live Tail no modo somente impressão, insira o comando a seguir.

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs
```

Quando você usa o modo somente impressão, também pode canalizá-lo com outros comandos do Linux para aumentar seus recursos analíticos. O exemplo a seguir filtra eventos de log com a palavra-chave `error` e imprime a segunda e a quarta colunas desses eventos para ajudar você a extrair determinadas informações.

```
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}'
```

interativo

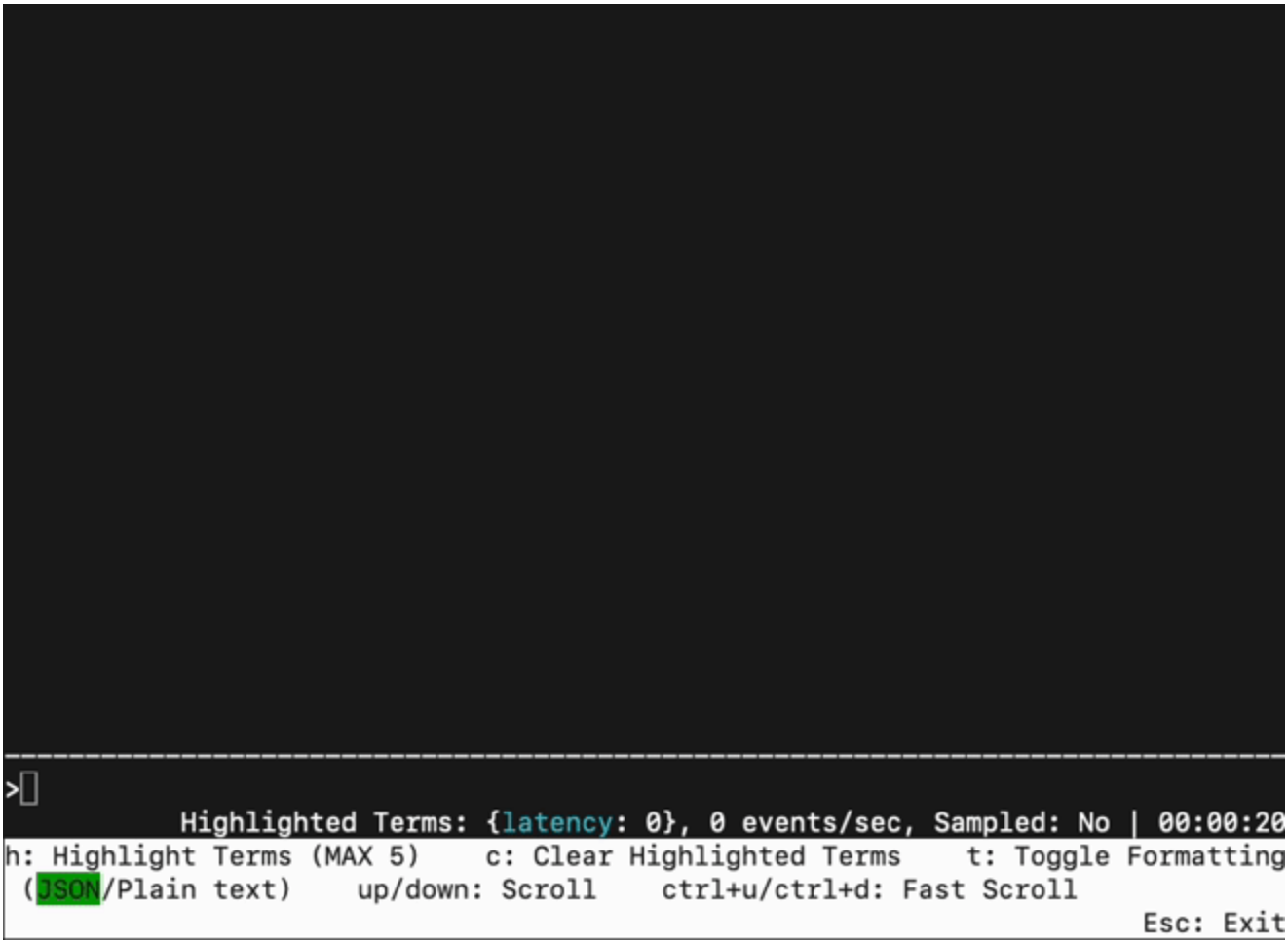
No modo `interactive`, você pode destacar termos e alternar o formato dos eventos do log de saída entre JSON e texto sem formatação. O modo interativo também exibe informações sobre

a sessão do Live Tail, como a duração da sessão, se a sessão está sendo amostrada, os termos atualmente em destaque e o número de vezes que foram encontrados.

Para iniciar uma sessão do Live Tail no modo interativo, insira o comando a seguir.

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs --mode interactive
```

A sessão do Live Tail começa. O vídeo a seguir mostra parte de um exemplo de sessão.



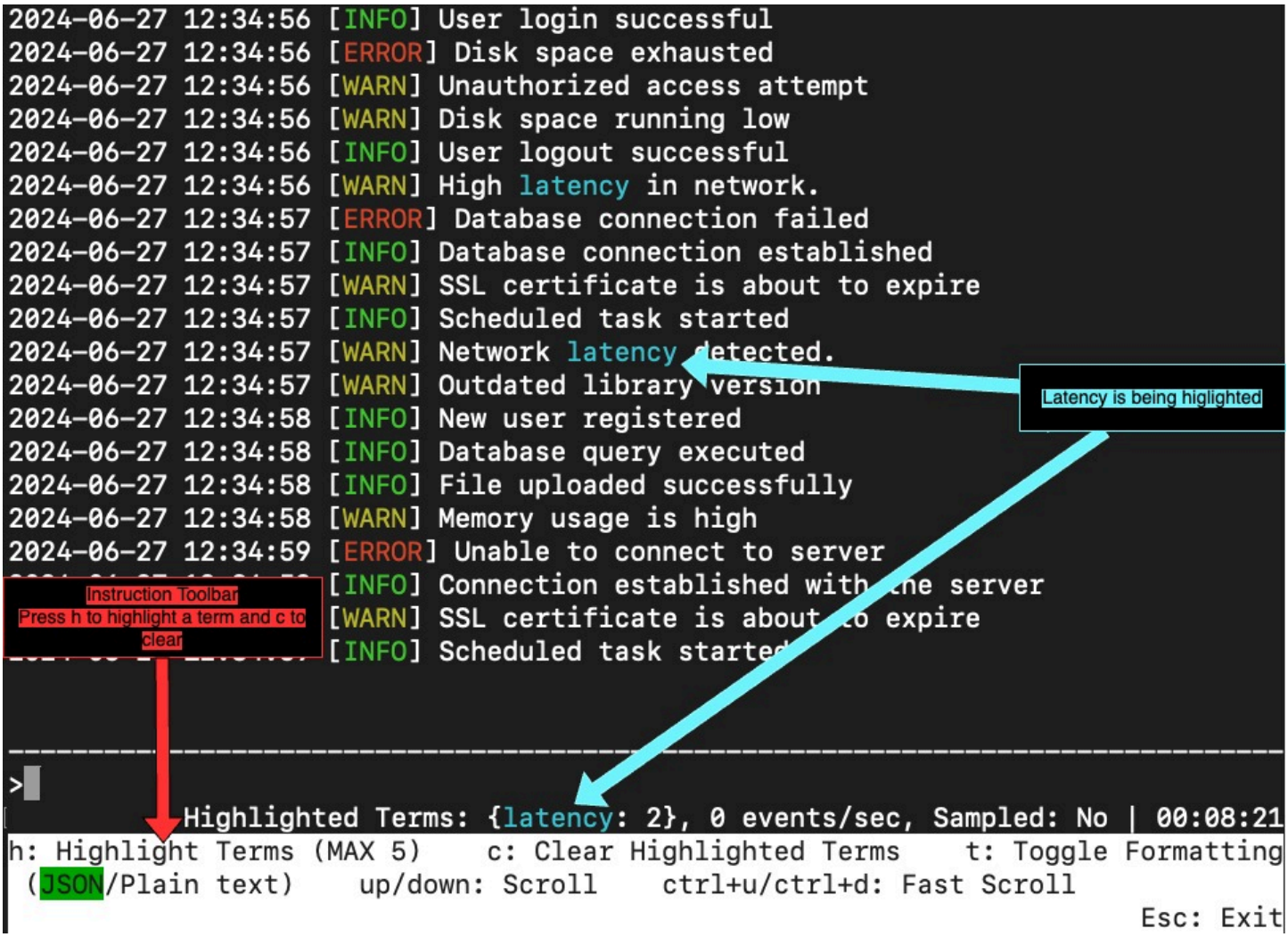
Para destacar um termo nos logs de streaming, pressione h e insira o termo. O exemplo a seguir mostra a tela após o termo `latency` ter sido destacado.

Para limpar um termo destacado, pressione c e digite o número que representa o termo que você não deseja mais destacar.

Você pode pressionar t para alternar o formato de exibição dos eventos recebidos entre JSON e texto sem formatação. A funcionalidade de alternância é habilitada quando possível e só acontecerá se o formato do evento de logs for compatível.

Você pode usar as teclas de seta para cima e seta para baixo para rolar, e usar `CTRL+u` e `CTRL+d` para rolar mais rápido.

A imagem a seguir mostra o destaque do termo `latency` durante uma sessão do Live Tail.



Iniciar uma sessão do Live Tail no console

Você usa o CloudWatch console para iniciar uma sessão do Live Tail. O procedimento a seguir explica como iniciar uma sessão Live Tail usando a opção Live Tail no painel de navegação esquerdo. Você também pode iniciar sessões do Live Tail na página Grupos de registros ou na página CloudWatch Logs Insights.

Se você estiver usando políticas de proteção de dados para mascarar dados confidenciais em um grupo de logs que está visualizando com o Live Tail, esses dados sempre aparecerão mascarados na sessão Live Tail. Para obter mais informações sobre como mascarar dados confidenciais em grupos de logs, consulte Ajude a proteger dados de log confidenciais com mascaramento.

> **Important**
>
> Se sua equipe de segurança de rede não permite o uso de soquetes web, você não pode acessar atualmente a parte Live Tail do CloudWatch console. Você pode usar o Live Tail com as APIs AWS CLI ou. Para obter mais informações, consulte Inicie uma sessão do Live Tail usando o AWS CLI e StartLiveTail.

Para iniciar uma sessão Live Tail

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs, Live tail.
3. Em Selecionar grupos de logs, selecione os grupos de logs cujos eventos você deseja visualizar na sessão Live Tail. Você pode selecionar até 10 grupos de logs.
4. (Opcional) Se você tiver selecionado somente um grupo de logs, poderá filtrar ainda mais sua sessão Live Tail selecionando um ou mais fluxos de logs dos quais visualizar eventos de log. Para fazer isso, em Selecionar fluxos de logs, selecione os nomes dos fluxos de logs na lista suspensa. Como alternativa, você pode usar a segunda caixa em Selecionar fluxos de logs para inserir um prefixo de nome de fluxo de logs. Todos os fluxos de log com nomes que corresponderem ao prefixo serão selecionados.
5. (Opcional) Para exibir somente eventos de log que contenham certas palavras ou outras strings, insira a palavra ou string em `Add filter patterns`.

   Por exemplo, para exibir somente eventos de log que incluam a palavra `Warning`, insira **`Warning`**. Os filtros diferenciam maiúsculas de minúsculas. É possível incluir vários termos e operadores de padrão nesse campo:

   - **`error 404`** exibe somente eventos de log que incluem tanto `error` quanto `404`
   - **`?Error ?error`** exibe eventos de log que incluem `Error` ou `error`
   - **`-INFO`** exibe todos os eventos de log que não incluem `INFO`
   - **`{ $.eventType = "UpdateTrail" }`** exibe todos os eventos de log JSON em que o valor do campo de tipo de evento é `UpdateTrail`

Você também pode usar a expressão regular (regex) para filtrar:

- **`%ERROR%`** usa regex para exibir todos os eventos de log que consistem na palavra-chave ERRO
- **`{ $.names = %Steve% }`** usa regex para exibir eventos de log JSON em que Steve está na propriedade `"name"`
- **`[ w1 = %abc%, w2 ]`** usa regex para exibir eventos de log delimitados por espaço em que a primeira palavra é abc

Para obter mais informações sobre a sintaxe do padrão, consulte Sintaxe de padrões de filtros.

6. (Opcional) Para realçar alguns dos eventos de log exibidos, insira um termo para procurar e realçar em Live Tail. Insira um por vez os termos de realce. Se você adicionar vários termos para realçar, uma cor diferente será atribuída para representar cada um. Um indicador de realce é exibido à esquerda de qualquer evento de log que contenha o termo especificado e também abaixo do próprio termo quando você expande o evento de log na janela principal para visualizar o evento de log completo.

   Você pode usar a filtragem e o realce para solucionar problemas rapidamente. Por exemplo, você pode filtrar os eventos para exibir somente aqueles que contêm `Error` e, em seguida, realçar os eventos que contêm `404`.

7. Para iniciar a sessão, escolha Aplicar filtros.

   Os eventos de log correspondentes começam a aparecer na janela. Também são exibidas as seguintes informações:

   - O timer mostra por quanto tempo a sessão Live Tail está ativa.
   - events/secexibe quantos eventos de log ingeridos por segundo correspondem aos filtros que você definiu.
   - Para evitar que a sessão seja rolada muito rápido porque muitos eventos correspondem aos filtros, o CloudWatch Logs pode exibir somente alguns eventos correspondentes. Se isso acontecer, a porcentagem de eventos correspondentes exibidos na tela é mostrada em % de exibidos.

8. Para pausar o fluxo de eventos para investigar o que está sendo exibido atualmente, escolha qualquer lugar na janela de eventos.

9. Durante a sessão, é possível usar o seguinte para ver mais detalhes sobre cada evento de log.

   - Para exibir o texto inteiro de um evento de log na janela principal, escolha a seta ao lado desse evento de log.
   - Para exibir o texto inteiro de um evento de log em uma janela lateral, escolha a lupa + ao lado desse evento de log. O fluxo de eventos é interrompido, e a janela lateral é exibida.

     A exibição do texto de um evento de log na janela lateral pode ser útil para comparar seu texto com outros eventos na janela principal.
10. Para interromper a sessão Live Tail, escolha Parar.
11. Para reiniciar a sessão, uma opção é usar o painel Filtro para modificar os critérios de filtragem e escolher Aplicar filtros. Depois, selecione Start (Iniciar).

Cross-account Centralização de registros entre regiões

A centralização de dados do Amazon CloudWatch Logs funciona com AWS Organizations a coleta de dados de log de várias contas de membros em um repositório de dados usando regras de centralização entre contas e regiões. Você define as regras que replicam automaticamente os dados de logs de várias contas e Regiões da AWS em uma conta centralizada em sua organização. Esse recurso simplifica a consolidação de registros para melhorar o monitoramento, a análise e a conformidade centralizados em toda a AWS sua infraestrutura.

CloudWatch A centralização de dados de registros oferece flexibilidade de configuração para atender aos requisitos operacionais e de segurança, como a capacidade de configurar uma região de backup durante a configuração da regra na conta de destino para garantir maior resiliência. Além disso, você tem controle total sobre o comportamento de criptografia de grupos de registros copiados das contas de origem para lidar com dados originalmente criptografados com chaves KMS gerenciadas pelo cliente.

> **Note**
>
> O recurso de centralização de CloudWatch registros processa somente novos dados de registro que chegam às contas de origem após a criação da regra de centralização. Os dados históricos de log (logs que existiam antes da criação da regra) não são centralizados.

Conceitos da centralização de dados

Antes de começar a usar a centralização de dados do CloudWatch Logs, familiarize-se com os seguintes conceitos:

Regra de centralização

Uma configuração que define como os dados de log das contas e regiões de origem são replicados em uma conta e região de destino. As regras especificam os critérios de origem e as configurações de destino.

Conta de origem

A AWS conta na qual os dados de registro se originam. Os eventos de logs das contas de origem são replicados para a conta de destino com base nas regras de centralização que você define.

Conta de destino

A AWS conta de destino na qual os dados de registro replicados são armazenados. Essa conta serve como local centralizado para análise e monitoramento de logs.

Região de backup

Uma região secundária opcional na conta de destino na qual os dados de log podem ser replicados para obter maior resiliência e no caso de recuperação de desastres.

Propagação de tags

Um recurso opcional que propaga tags de recursos dos grupos de registros de origem para os grupos de registros de destino correspondentes. A propagação de tags usa uma função do IAM gerenciada pelo cliente na conta de destino para adicionar, atualizar e remover tags nos grupos de registros de destino. Você configura a propagação de tags adicionando um `TagPropagationConfiguration` bloco à configuração de destino da regra de centralização.

Criptografia em CloudWatch registros

Os dados do grupo de registros são sempre criptografados em CloudWatch Registros. Por padrão, o CloudWatch Logs usa criptografia do lado do servidor com o Galois/Counter Modo Padrão de Criptografia Avançada de 256 bits (AES-GCM) para criptografar dados de log em repouso. Como alternativa, você pode usar o Serviço de Gerenciamento de AWS Chaves para essa criptografia. Para obter mais informações, consulte a documentação sobre criptografia de CloudWatch registros.

- Como a criptografia funciona durante a centralização: a centralização de CloudWatch registros copia ativamente os dados de registro no momento da ingestão das contas de origem para as contas de destino. Durante esse processo, seus dados permanecem criptografados em trânsito usando uma chave AWS de serviço própria. Os dados em repouso nos grupos de log de origem e de destino são criptografados usando o método de criptografia escolhido (chaves KMS gerenciadas ou de AWS propriedade do cliente). Se você estiver usando a chave KMS gerenciada pelo cliente em seus grupos de registros de destino, adicione a tag `LogsManaged = true` à chave kms para que o serviço de centralização possa acessá-la.
- Quando as permissões do KMS são necessárias:
  - Se você estiver usando chaves KMS gerenciadas pelo cliente em suas contas de origem, o CloudWatch Logs exigirá permissões de KMS nos seguintes cenários de exemplo:
    - Gerenciamento de taxa de transferência: quando os limites de taxa de transferência de centralização são atingidos, os dados de registro são temporariamente armazenados

criptografados com sua chave KMS gerenciada pelo cliente até que a largura de banda fique disponível.

- Proteção e redação de dados: quando os grupos de registros de origem têm políticas de proteção de dados ativadas, CloudWatch os registros exigem permissões de descriptografia para acessar dados de registro brutos e centralizá-los.

> **Important**
>
> As regras de centralização são gerenciadas pela conta de gerenciamento da AWS organização ou pelo administrador delegado. Para excluir da centralização grupos de KMS-encrypted registros gerenciados pelo cliente, defina as configurações da regra como “Não centralizar grupos de registros criptografados com chave AWS KMS”.

Configurando a centralização de logs

Para configurar a centralização de CloudWatch registros, você precisa configurar regras de centralização que definam como os dados de registro fluem dos grupos de registro nas contas de origem para os grupos de registro na sua conta de destino.

Depois que a regra de centralização estiver ativada e os eventos de log estiverem sendo replicados na conta de destino, você poderá criar filtros de métrica, de assinatura e de conta em grupos de logs centralizados com recursos aprimorados de filtragem. Esses filtros podem direcionar eventos de logs de contas e regiões de origem específicas e podem emitir informações da conta de origem e da região como dimensões métricas. Para obter mais informações, consulte Criar métricas de eventos de log usando filtros.

Pré-requisitos

- AWS Organizations devem ser configuradas e as contas de origem e de destino devem pertencer à organização.
- O acesso confiável deve estar habilitado para CloudWatch a conta de gerenciamento e a conta de destino, portanto, forneça acesso aos dados de registro.

> **Note**
>
> É recomendável habilitar o acesso confiável pelo console, o que cria automaticamente o perfil vinculado a serviços (SLR). Se o acesso confiável for habilitado por meio de outros métodos, a função vinculada ao serviço precisará ser criada separadamente.

Pré-requisitos de propagação de tags

Para ativar a propagação de tags, você deve concluir a seguinte configuração adicional:

Crie uma função do IAM gerenciada pelo cliente

Você deve criar uma função do IAM gerenciada pelo cliente na conta de destino com a seguinte configuração:

Política de confiança

A política de confiança da função deve permitir que a função vinculada ao serviço de centralização a assuma. O exemplo de política de confiança a seguir concede à função `AWSServiceRoleForObservabilityAdmin_LogsCentralization` vinculada ao serviço permissão para assumir a função de destino. `<destination-account-id>`Substitua pelo ID da sua conta de destino e `<organization-id>` pelo seu AWS Organizations ID.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::<destination-account-id>:role/aws-
service-role/logs-centralization.observabilityadmin.amazonaws.com/
AWSServiceRoleForObservabilityAdmin_LogsCentralization"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "<organization-id>"
      }
    }
  }]
```

```
}
```

Política de permissões

A política de permissões da função deve conceder operações de tag nos grupos de registros de destino. O exemplo a seguir concede as permissões mínimas necessárias. `<destination-account-id>`Substitua pelo seu valor. Você pode `Resource` ampliar o escopo para padrões específicos de ARN de grupos de registros.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "logs:ListTagsForResource",
      "logs:TagResource",
      "logs:UntagResource"
    ],
    "Resource": "arn:aws:logs:*:<destination-account-id>:log-group:*"
  }]
}
```

Conceder a iam: PassRole permissão

Você deve ter `iam:PassRole` permissão na função de destino, com escopo definido para o serviço de centralização por meio da chave de `iam:PassedToService` condição. O exemplo a seguir concede permissão para passar a função. Substitua `<destination-account-id>` e `<your-tag-role-name>` pelos seus valores.

> **Note**
>
> Essa declaração se aplica à sua política de identidade (a chamada principal `CreateCentralizationRuleForOrganization` ou`UpdateCentralizationRuleForOrganization`), não à função de destino em si.

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

```
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "arn:aws:iam::<destination-account-id>:role/<your-tag-role-name>",
    "Condition": {
      "StringEquals": {
        "iam:PassedToService": "logs-centralization.observabilityadmin.amazonaws.com"
      }
    }
  }]
}
```

Personalizando nomes de grupos de registros de destino

Ao criar uma regra de centralização, você pode personalizar como os nomes dos grupos de registros de destino são estruturados usando atributos. Esses atributos são automaticamente substituídos por valores reais quando grupos de registros são criados, permitindo que você organize os registros hierarquicamente na sua conta de destino. Por padrão, somente o `${source.logGroup}` atributo é usado, que mescla todos os grupos de registros com o mesmo nome na conta de destino. Se uma variável não puder ser resolvida, ela herdará o valor de sua variável principal na hierarquia.

Atributos disponíveis

Você pode usar os seguintes atributos em seu padrão de nome de grupo de registros de destino:

Atributos do nome do grupo de registros de destino

Atributo	Description
`${source.accountId}`	O ID da AWS conta em que o registro se originou.
`${source.region}`	O Região da AWS local de origem do registro.
`${source.logGroup}`	O nome original do grupo de registros da conta de origem.
`${source.org.id}`	Seu AWS Organizations ID da conta de origem.
`${source.org.ouId}`	O ID da unidade organizacional da conta de origem
`${source.org.rootId}`	O ID raiz da organização
`${source.org.path}`	O caminho organizacional completo da conta até a raiz

Exemplos

Preserve a estrutura original do grupo de registros

Padrão: `/centralized/${source.accountId}${source.logGroup}`

Result: `/centralized/123456789012/aws/lambda/my-function`

Organize por conta e região

Padrão: `/centralized/${source.accountId}/${source.region}`

Result: `/centralized/123456789012/us-east-1`

Organizar por estrutura organizacional

Padrão: `/logs/${source.org.id}/${source.org.ouId}/${source.accountId}`

Result: `/logs/o-abc123/ou-xyz-12345678/123456789012`

Estrutura plana simples

Padrão: `/centralized-logs`

Result: `/centralized-logs`

Práticas recomendadas

- Inclua o ID da conta de origem para identificar facilmente de quais registros da conta vieram.
- Inclua a região de origem se você estiver centralizando em várias regiões.
- Estruture os nomes dos grupos de registros de destino com menos de 512 caracteres. CloudWatch Os registros impõem um tamanho máximo de nome de grupo de registros de 512 caracteres.
- Se você habilitar a propagação de tags, o padrão deverá conter todos os três atributos: `${source.logGroup}${source.accountId}`, e. `${source.region}` Isso garante que cada grupo de registros de origem seja mapeado para um grupo de registros de destino exclusivo.

Criação de uma regra de centralização

Use o procedimento a seguir para criar uma regra de centralização que replica os dados de log das contas de origem para sua conta de destino.

Para criar uma regra de centralização

1. Navegue até o CloudWatch console na conta de gerenciamento ou administrador delegado da organização.
2. Escolha Configurações.
3. Acesse a guia Organização.
4. Escolha Configurar regra.
5. Especifique os detalhes da fonte definindo os seguintes campos e, em seguida, escolha Avançar:

   a. Nome da regra de centralização: insira um nome exclusivo para a regra de centralização.

   b. Contas de origem: defina os critérios de seleção da origem para escolher contas a partir das quais os dados de telemetria serão centralizados. Os critérios de seleção podem incluir:

      - Uma ista das contas de membros da organização
      - Uma lista de unidades organizacionais na organização
      - Toda a organização

      Você pode fornecer os critérios de seleção de dois modos:

      - Builder: uma experiência baseada em cliques para gerar os critérios de seleção da origem
      - Editor: uma caixa de texto de formato livre para fornecer os critérios de seleção da origem

      Sintaxe suportada para critérios de seleção de origem:

      - Teclas suportadas: OrganizationId | OrganizationUnitId | AccountId | *
      - Operadores compatíveis: = | IN | OR

   c. Regiões de origem: selecione uma lista de regiões onde procurar os dados de telemetria a serem centralizados.

6. Especifique os detalhes do destino definindo os seguintes campos e, em seguida, escolha Avançar:

   a. Conta de destino: selecione uma conta na organização que atue como destino central para os dados de telemetria.

b. Região de destino: selecione uma região primária que armazene uma cópia dos dados de telemetria centralizada.

c. Região de backup: opcionalmente, selecione uma região que armazenará uma segunda cópia dos dados de telemetria centralizados.

7. Especifique os dados de telemetria definindo os seguintes campos e, em seguida, escolha Avançar:

a. Grupos de logs: escolha uma das seguintes opções:

- Todos os grupos de logs: centralize os logs de todos os grupos de logs nas contas de origem.
- Grupo de registros de filtros: centralize os registros de um subconjunto de grupos de registros nas contas de origem, atendendo aos critérios de seleção. Você pode fornecer os critérios de seleção de dois modos:
  - Builder: uma experiência baseada em escolhas para gerar os critérios de seleção
  - Editor: uma caixa de texto de formato livre para fornecer os critérios de seleção

  Há dois critérios de seleção que você pode usar para filtrar registros:
  - Critérios de seleção de grupos de registros: os critérios de seleção que especificam quais grupos de registros de origem devem ser centralizados.
    - Teclas suportadas: LogGroupName | *
    - Operadores compatíveis: = | != | IN | NOT IN | AND | OR | LIKE | NOT LIKE
  - Critérios de seleção da fonte de dados: os critérios de seleção que especificam quais fontes de dados devem ser centralizadas.
    - Teclas suportadas: DataSourceName | DataSourceType
    - Operadores compatíveis: = | != | IN | NOT IN | AND | OR | LIKE | NOT LIKE

  Quando os critérios de seleção do grupo de registros e os critérios de seleção da fonte de dados são especificados, um evento de log deve corresponder aos dois critérios para ser centralizado.

b. Grupo de logs criptografados do KMS

> **Important**
>
> CloudWatch as regras de centralização não entregarão os registros da conta de origem aos grupos de registros de destino se a chave KMS fornecida na regra

de centralização não permitir que CloudWatch os registros a usem. Se você estiver usando a chave KMS gerenciada pelo cliente em seus grupos de registros de destino, adicione a tag LogsManaged = true à chave kms. Para obter mais informações, consulte Etapa 2: definir permissões na chave do KMS.

Escolha uma das seguintes opções:

- Centralize grupos de registros de origem criptografados com chaves KMS gerenciadas pelo cliente usando uma chave KMS gerenciada pelo cliente específica de destino: centralize os eventos de registro dos grupos de registro de origem criptografados com chaves KMS gerenciadas pelo cliente em grupos de registro de destino criptografados com uma chave KMS gerenciada pelo cliente na conta de destino.

  Quando esta configuração é selecionada, você também deve definir o seguinte:

  - ARN da chave de criptografia de destino: ARN da chave KMS gerenciada pelo cliente na conta de destino e na região de destino principal, a ser associada aos grupos de registros de destino recém-criados.
  - ARN da chave de criptografia de destino do backup (se a região de backup estiver selecionada): ARN da chave KMS gerenciada pelo cliente na conta de destino e na região de destino do backup, a ser associada aos grupos de registros de destino recém-criados.
  - Pule a centralização para grupos de registros de destino não criptografados (opcional): se um grupo de registros já existir sem uma chave KMS gerenciada pelo cliente, CloudWatch não é possível atualizar sua criptografia. Escolha essa opção para ignorar a centralização de eventos de log de grupos de log de origem criptografados com chaves KMS gerenciadas pelo cliente para grupos de log de destino que não estão associados a uma chave KMS gerenciada pelo cliente.
- Centralize grupos de log criptografados com chaves KMS gerenciadas pelo cliente na conta de destino com chave KMS AWS própria: centralize eventos de log de grupos de log de origem criptografados com chaves KMS gerenciadas pelo cliente em grupos de log de destino recém-criados criptografados usando uma chave KMS própria. AWS
- Não centralize grupos de log criptografados com chaves KMS gerenciadas pelo cliente: ignore a centralização de eventos de log de grupos de log de origem criptografados com chaves KMS gerenciadas pelo cliente.

8. Revise a regra de centralização, opcionalmente, faça edições de última hora e escolha Criar política de centralização.

Modificando uma regra de centralização

Use o procedimento a seguir para modificar uma regra de centralização existente.

Para modificar uma regra de centralização

1. Navegue até o CloudWatch console na conta de gerenciamento ou administrador delegado da organização.
2. Escolha Configurações.
3. Acesse a guia Organização.
4. Escolha Gerenciar regras.
5. Selecione a regra a ser atualizada e escolha Editar.
6. Atualize a configuração da regra conforme necessário, escolhendo Avançar para prosseguir com cada etapa.
7. Na Etapa 4, Revisar e configurar, escolha Atualizar política de centralização.

Visualizar uma regra de centralização

Use o procedimento a seguir para exibir detalhes de uma regra de centralização existente.

Para visualizar uma regra de centralização

1. Navegue até o CloudWatch console na conta de gerenciamento ou administrador delegado da organização.
2. Escolha Configurações.
3. Acesse a guia Organização.
4. Escolha Gerenciar regras.
5. Veja uma lista de todas as regras de centralização existentes e escolha um nome de regra específico para ver seus detalhes.

Excluindo uma regra de centralização

Use o procedimento a seguir para excluir uma regra de centralização existente.

Para excluir uma regra de centralização

1. Navegue até o CloudWatch console na conta de gerenciamento ou administrador delegado da organização.
2. Escolha Configurações.
3. Acesse a guia Organização.
4. Escolha Gerenciar regras.
5. Selecione a regra para excluir e escolha Excluir.
6. Confirme a exclusão e escolha Excluir.

Monitoramento e solução de problemas de centralização

Você pode monitorar o status e o desempenho de suas regras de centralização usando CloudWatch métricas, o console do CloudWatch Logs e AWS CloudTrail os registros. Isso ajuda você a garantir que os dados de log sejam replicados com sucesso e a identificar quaisquer problemas com sua configuração de centralização.

CloudWatch O Logs fornece:

1. Integridade da regra por regra de centralização
   a. Escolha Configurações.
   b. Acesse a guia Organização.
   c. Escolha Gerenciar regras.
2. Registra chamadas de API com AWS CloudTrail
3. CloudWatch também publica métricas para centralização, incluindo eventos de log replicados, erros e limitação. Para obter mais informações sobre essas métricas e suas dimensões, consulteMétricas e dimensões de centralização.

Status de integridade da regra de centralização

Cada regra de centralização tem um status de integridade que indica se ela está operando corretamente. Você pode verificar a integridade das regras por meio do console ou programaticamente com a API.

Os status de integridade das regras incluem:

- `HEALTHY`: a regra está operando normalmente e replicando os dados de log conforme configurado
- `UNHEALTHY`: a regra encontrou problemas e pode não estar replicando os dados corretamente
- `PROVISIONING`: a centralização da organização está em processo de configuração.

Quando uma regra é marcada como NÃO ÍNTEGRA, o campo `FailureReason` fornece detalhes sobre o problema específico que precisa ser resolvido.

Status de integridade da propagação de tags

A propagação de tags tem seu próprio status de integridade (`TagPropagationStatus`) que é independente do geral `RuleHealth` para entrega de registros. Se você configurar incorretamente a função de destino, a integridade geral da regra não será afetada — somente a propagação de tags será exibida como insalubre.

- `Healthy`: A tentativa mais recente de propagação de tags foi bem-sucedida.
- `Unhealthy`: A tentativa mais recente de propagação de tags falhou. Verifique o `TagPropagationFailureReason` campo para obter detalhes. Consulte a seção de solução de problemas abaixo para ver as etapas de correção.

Monitoramento de chamadas de API de centralização com AWS CloudTrail

AWS CloudTrail registra chamadas de API feitas para o serviço de centralização, permitindo que você acompanhe alterações de configuração e solucione problemas em contas que são membros da sua. AWS Organizations

CloudTrail Os principais eventos para centralização incluem:

- `CreateCentralizationRuleForOrganization`: quando uma nova regra de centralização é criada

- `UpdateCentralizationRuleForOrganization`: quando uma regra existente é modificada
- `DeleteCentralizationRuleForOrganization`: quando uma regra é excluída
- `GetCentralizationRuleForOrganization`: quando os detalhes da regra são recuperados
- `ListCentralizationRulesForOrganization`: quando as regras são listadas

Você pode usar CloudTrail registros para auditar alterações na configuração de centralização e correlacioná-las com problemas de desempenho ou falhas de replicação.

Recomendações de monitoramento

Para garantir que a centralização esteja funcionando corretamente, recomendamos configurar CloudWatch alarmes nas principais métricas de centralização que vendemos à Metrics. CloudWatch Esse monitoramento proativo ajuda você a detectar problemas precocemente e a manter uma centralização confiável de registros em toda a organização.

As principais métricas a serem monitoradas incluem:

- `IncomingCopiedBytes`: monitore o volume de dados de registro que estão sendo replicados com sucesso em sua conta de destino. Uma queda repentina ou a ausência dessa métrica pode indicar problemas de centralização.
- `CentralizationError`: configure alarmes para qualquer erro no processo de centralização para identificar e resolver problemas rapidamente.
- `CentralizationThrottled`: monitore os eventos de limitação que podem afetar o desempenho da replicação de registros.

Para obter uma lista completa das métricas de centralização disponíveis e suas dimensões, consulteMétricas e dimensões de centralização.

Se os registros não estiverem sendo centralizados conforme o esperado, analise os seguintes cenários comuns que podem impedir a centralização de registros.

Dados históricos de registro

> O recurso de centralização de CloudWatch registros processa somente novos dados de registro que chegam às contas de origem após a criação da regra de centralização. Os dados históricos de log (logs que existiam antes da criação da regra) não são centralizados.

Permissões da chave KMS

As regras de centralização não entregarão os registros da conta de origem aos grupos de registros de destino se a chave KMS fornecida na regra de centralização não permitir que CloudWatch os registros a usem. Certifique-se de que a política de chaves do KMS conceda as permissões necessárias aos CloudWatch registros. Para obter mais informações, consulte Etapa 2: definir permissões na chave do KMS.

Configuração de chave KMS gerenciada pelo cliente

Se você selecionou Não centralizar grupos de log criptografados com a chave KMS gerenciada pelo cliente durante a criação da regra, os eventos de log dos grupos de log de origem criptografados com a chave KMS gerenciada pelo cliente serão ignorados e não centralizados.

Incompatibilidade de criptografia de destino

Se o grupo de registros de destino já existir com uma configuração de criptografia KMS diferente da especificada pela regra de centralização e a resolução de conflitos estiver definida como SKIP, os registros serão descartados e um `DestinationEncryptionMismatch` erro será emitido. Por exemplo, isso ocorre quando o destino tem criptografia padrão, mas a regra especifica uma chave KMS gerenciada pelo cliente.

Acesso confiável não ativado

O acesso confiável deve estar ativado para CloudWatch que a conta de gerenciamento e a conta de destino forneçam acesso aos dados de registro. AWS Organizations

Critérios de seleção da fonte

Verifique se os critérios de seleção de origem da regra de centralização estão configurados corretamente:

- Contas e regiões: certifique-se de que as contas de origem e as regiões de origem dos registros estejam incluídas na regra. Grupos de registros de contas ou regiões não especificadas na regra não serão centralizados.
- Filtros de grupos de registros: se você configurou filtros de grupos de registros, somente grupos de registros que correspondam aos critérios especificados serão centralizados. Verifique se seus critérios de seleção de grupos de registros incluem os grupos de registros que você espera centralizar.
- Associação à organização: as contas de origem e de destino devem pertencer à mesma AWS Organizations organização. Contas fora da organização não podem participar da centralização.

Limite de cota do grupo de registros atingido

Se a conta de destino atingir o limite de cota do grupo de registros, novos grupos de registros não poderão ser criados para centralização. Verifique se a conta de destino tem cota suficiente para acomodar grupos de registros centralizados de todas as contas de origem. É possível solicitar um aumento de cota, se necessário.

Limite de tamanho do nome do fluxo de log excedido

Os nomes dos fluxos de log têm restrições de tamanho máximo. Quando a centralização replica fluxos de log para a conta de destino, um sufixo é adicionado ao nome do stream de log. Se o nome do fluxo de registros resultante exceder o tamanho máximo permitido, os registros serão descartados e um `InvalidLogStream` erro será emitido na conta do cliente.

Estado de saúde da regra

Verifique o status de integridade da regra de centralização no console ou usando a `GetCentralizationRuleForOrganization` API. Se a regra estiver marcada como NÃO SAUDÁVEL, revise o `FailureReason` campo para obter detalhes específicos sobre o problema.

Status de integridade da propagação de tags

Se `TagPropagationStatus` aparecer`Unhealthy`, verifique o `TagPropagationFailureReason` campo:

- `RoleNotAssumable`: o serviço não pode assumir a função de destino. Verifique se a política de confiança da função permite que a função vinculada ao serviço de centralização a assuma e se ela `sts:ExternalId` corresponde à ID da sua organização.
- `RoleLacksPermissions`: A função foi assumida, mas a chamada da API de tags foi negada. Garanta que a política de permissões da função conceda `logs:ListTagsForResourcelogs:TagResource`, e `logs:UntagResource` nos grupos de registros de destino.

Para diagnosticar problemas de centralização, revise o status de integridade da regra de centralização no console, verifique as CloudWatch métricas em busca de erros e limitação e examine os AWS CloudTrail registros de falhas nas chamadas de API. Para obter mais informações sobre métricas de centralização, consulteMétricas e dimensões de centralização.

Trabalhar com grupos de logs e fluxos de logs

Fluxo de logs é uma sequência de eventos de log que compartilham a mesma origem. Cada fonte separada de registros no CloudWatch Logs compõe um fluxo de registros separado.

Um grupo logs é um grupo de fluxos de log que compartilham as mesmas configurações de retenção, monitoramento e controle de acesso. Você pode definir grupos de logs e especificar quais fluxos colocar em cada grupo. Não há limite para o número de streams de log que podem pertencer a um grupo de logs.

Para organizações que precisam consolidar dados de registro de várias contas e regiões, você pode usar a centralização de CloudWatch registros para replicar automaticamente grupos de registros em uma conta central. Para obter mais informações, consulte Cross-account Centralização de registros entre regiões.

Use os procedimentos desta seção para trabalhar com grupos e fluxos de logs.

Crie um grupo de registros em CloudWatch Registros

Quando você instala o agente CloudWatch Logs em uma instância do Amazon EC2 usando as etapas nas seções anteriores do Amazon CloudWatch Logs User Guide, o grupo de logs é criado como parte desse processo. Você também pode criar um grupo de registros diretamente no CloudWatch console.

Para criar um grupo de logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Gerenciamento de registros.
3. Selecione Actions (Ações) e selecione Create log group (Criar grupo de logs).
4. Digite um nome para o grupo de logs e escolha Create log group (Criar grupo de logs).

> **Tip**
>
> Você pode dar preferência a grupos de log, bem como a painéis e alarmes, no menu Favorites and recents (Favoritos e recentes) no painel de navegação. Na coluna Recently

visited (Visitados recentemente), passe o mouse sobre o grupo de logs ao qual deseja dar preferência e escolha o símbolo da estrela ao lado dele.

Enviar logs a um grupo de logs

CloudWatch O Logs recebe automaticamente eventos de log de vários AWS serviços. Você também pode enviar outros eventos de registro para o CloudWatch Logs usando um dos seguintes métodos:

- CloudWatch agente — O CloudWatch agente unificado pode enviar métricas e registros para o CloudWatch Logs. Para obter informações sobre como instalar e usar o CloudWatch agente, consulte Coletando métricas e registros de instâncias do Amazon EC2 e servidores locais com o CloudWatch agente no Guia do usuário da Amazon CloudWatch .
- AWS CLI—O put-log-eventsupload de lotes de eventos de log para CloudWatch o Logs.
- Programaticamente — A PutLogEventsAPI permite que você faça upload programático de lotes de eventos de log para o Logs. CloudWatch

Exibir dados de registro enviados para CloudWatch Logs

Você pode visualizar e percorrer os dados de registro stream-by-stream conforme enviados ao CloudWatch Logs pelo agente do CloudWatch Logs. Você pode especificar o intervalo de tempo para os dados de log a serem visualizados.

Para visualizar dados de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Gerenciamento de registros.
3. Em Grupos de logs, escolha o grupo de logs para visualizar os fluxos.
4. Na lista de grupos de logs, escolha o nome do grupo de logs que deseja visualizar.
5. Na lista de fluxos de logs, escolha o nome do fluxo de log que deseja visualizar.
6. Para alterar a forma como os dados de log são exibidos, faça o seguinte:
   - Para expandir um único evento de log, escolha a seta ao lado dele.
   - Para expandir todos os eventos de log e visualizá-los como texto simples, acima da lista de eventos de log, escolha Texto.

- Para filtrar os eventos de log, insira o filtro de pesquisa desejado no campo de pesquisa. Para obter mais informações, consulte Criar métricas de eventos de log usando filtros.
- Para visualizar dados de log de um intervalo de data e hora especificado, escolha a seta ao lado da data e hora ao lado do filtro de pesquisa. Para especificar um intervalo de data e hora, escolha Absolute (Absoluto). Para escolher um número predefinido de minutos, horas, dias ou semanas, escolha Relative (Relativo). Também é possível alternar entre UTC e fuso horário local.

Pesquisar dados de log usando padrões de filtro

Você pode pesquisar seus dados de log usando Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura, filtros de eventos de log e Live Tail.. Você pode pesquisar todos os fluxos de log em um grupo de registros ou, usando o, AWS CLI você também pode pesquisar fluxos de log específicos. Na execução de cada pesquisa, ela retorna para a primeira página de dados encontrada e um token para recuperar a próxima página de dados ou continuar a pesquisa. Se não houver resultados retornados, você poderá continuar a pesquisar.

Você pode definir o intervalo de tempo que deseja consultar para limitar o escopo da pesquisa. Você pode começar com um intervalo maior para ver onde se encontram as linhas de log de interesse e, em seguida, reduzir o intervalo de tempo para limitar a exibição aos logs no intervalo de tempo de interesse.

Você também pode passar diretamente de suas métricas extraídas dos logs para os logs correspondentes.

Se você estiver conectado a uma conta configurada como uma conta de monitoramento na observabilidade CloudWatch entre contas, poderá pesquisar e filtrar eventos de log das contas de origem vinculadas a essa conta de monitoramento. Para obter mais informações, consulte Observabilidade entre contas do CloudWatch .

Pesquisar entradas de log usando o console

Você pode procurar entradas de log que atendam a critérios especificados usando o console.

Para pesquisar seus logs usando o console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Gerenciamento de registros.

3. Em Grupos de logs, escolha o nome do grupo de logs que contém o fluxo de log a ser pesquisado.
4. Em Fluxos de log, escolha o nome do fluxo de log para pesquisa.
5. Em Eventos de log, insira a sintaxe do filtro a ser usada.

Para pesquisar todas as entradas de log em um intervalo de tempo usando o console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Gerenciamento de registros.
3. Em Grupos de logs, escolha o nome do grupo de logs que contém o fluxo de log a ser pesquisado.
4. Escolha Pesquisar grupo de logs.
5. Para Eventos de log, selecione o intervalo de data e hora e insira a sintaxe do filtro.

Pesquisar entradas de registro usando o AWS CLI

Você pode pesquisar entradas de registro que atendam a um critério específico usando AWS CLI o.

Para pesquisar entradas de registro usando o AWS CLI

Em um prompt de comando, execute o seguinte comando da filter-log-events. Use `--filter-pattern` para limitar os resultados para o padrão de filtro especificado e `--log-stream-names` para limitar os resultados para os fluxos de logs especificado.

```
aws logs filter-log-events --log-group-name my-group [--log-stream-
names LIST_OF_STREAMS_TO_SEARCH] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

Para pesquisar entradas de registro em um determinado intervalo de tempo usando o AWS CLI

Em um prompt de comando, execute o seguinte filter-log-eventscomando:

```
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]
```

Passar de métricas para logs

Você pode obter entradas de log específicas de outras partes do console.

Para passar de widgets do painel para logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Painéis.
3. Escolha um painel.
4. No widget, escolha Exibir logs e, em seguida, escolha Exibir logs neste período. Se houver mais de um filtro de métrica, selecione um na lista. Se houver mais filtros de métrica do que podemos exibir na lista, escolha Mais filtros de métrica e selecione ou procure um filtro de métrica.

Para passar de métricas para logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Métricas.
3. No campo de pesquisa na guia Todas as métricas, digite o nome da métrica e pressione Enter.
4. Selecione uma ou mais métricas nos resultados da pesquisa.
5. Escolha Ações, Exibir logs. Se houver mais de um filtro de métrica, selecione um na lista. Se houver mais filtros de métrica do que podemos exibir na lista, escolha Mais filtros de métrica e selecione ou procure um filtro de métrica.

Solução de problemas

A pesquisa leva muito tempo para ser concluída

Se você tiver uma grande quantidade de dados de log, pode demorar muito tempo para a pesquisa ser concluída. Para acelerar uma pesquisa, você pode fazer o seguinte:

- Se você estiver usando o AWS CLI, você pode limitar a pesquisa apenas aos fluxos de log nos quais está interessado. Por exemplo, se seu grupo de registros tiver 1.000 fluxos de registros, mas você quiser ver apenas três fluxos de registros que você sabe que são relevantes, você pode usar o AWS CLI para limitar sua pesquisa somente aos três fluxos de registros dentro do grupo de registros.
- Use um intervalo de tempo menor, mais granular, o que reduz a quantidade de dados a serem pesquisados e acelera a consulta.

Alterar a retenção de dados de CloudWatch registro em registros

Por padrão, os dados de registro são armazenados em CloudWatch Registros indefinidamente. No entanto, você pode configurar quanto tempo armazenar os dados de log em um grupo de logs. Todos os dados mais antigos que a configuração de retenção atual serão excluídos. É possível alterar a retenção de logs de cada grupo de logs a qualquer momento.

> **Note**
>
> CloudWatch Logs não exclui imediatamente os eventos de registro quando eles atingem a configuração de retenção. Normalmente, demora até 72 horas depois disso para que os eventos de log sejam excluídos, mas em raras situações pode demorar mais.
> Isso significa que, se você alterar um grupo de logs para ter uma configuração de retenção mais longa quando ele contiver eventos de log que já passaram da data de expiração, mas que não foram realmente excluídos, esses eventos de log levarão até 72 horas para serem excluídos depois que a nova data de retenção for atingida. Para garantir que os dados de log sejam excluídos permanentemente, mantenha um grupo de logs em sua configuração de retenção inferior a 72 horas após o término do período de retenção anterior ou você ter confirmado que os eventos de log mais antigos foram excluídos.
> Quando os eventos de log atingem sua configuração de retenção, eles são marcados para exclusão. Depois de marcados para exclusão, eles não aumentam mais seus custos de armazenamento de arquivamento, mesmo que não sejam realmente excluídos no momento. Esses eventos de logs marcados para exclusão também não são incluídos quando você usa uma API para recuperar o valor `storedBytes` para ver quantos bytes um grupo de logs está armazenando.

Para alterar a configuração de retenção de logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Localize o grupo de logs a ser atualizado.
4. Na coluna Retenção para esse grupo de logs, escolha a configuração de retenção atual, como Nunca expirar.
5. Em Configuração de retenção, em Expirar eventos após, escolha um valor de retenção de logs e escolha Salvar.

Protegendo grupos de registros contra exclusão

Opcionalmente, você pode ativar a proteção contra exclusão para evitar a exclusão acidental de grupos de registros importantes. Para obter informações detalhadas sobre a proteção contra exclusão, consulteProtegendo grupos de registros contra exclusão.

Protegendo grupos de registros contra exclusão

Ativando a proteção contra exclusão

Você pode ativar a proteção contra exclusão ao criar um novo grupo de registros ou em grupos de registros existentes. Durante a criação do grupo de registros, selecione “Proteção de exclusão ativada” ou passe o parâmetro`--deletion-protection-enabled`. Por padrão, a proteção contra exclusão não está habilitada.

Para ativar ou desativar a proteção contra exclusão em um grupo de registros existente (console)

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Gerenciamento de registros.
3. Selecione o grupo de registros que você deseja proteger.
4. Escolha Ações, Editar proteção contra exclusão.
5. Na caixa de diálogo, revise e envie as alterações.

Se estiver usando o AWS CLI, para ativar a proteção contra exclusão em um grupo de registros existente:

```
aws logs put-log-group-deletion-protection \
--log-group-identifier "/my-application/logs" \
--deletion-protection-enabled
```

Para remover a proteção contra exclusão em um grupo de registros existente:

```
aws logs put-log-group-deletion-protection \
--log-group-identifier "/my-application/logs" \
--no-deletion-protection-enabled
```

Tratamento de erros

Se você tentar excluir um grupo de registros com a proteção de exclusão ativada, receberá uma mensagem `ValidationException` com a mensagem: “Não é possível excluir um grupo de registros com a proteção de exclusão ativada. Desative primeiro a proteção contra exclusão.”

Marque grupos de registros no Amazon CloudWatch Logs

Você pode atribuir seus próprios metadados aos grupos de registros que você cria no Amazon CloudWatch Logs na forma de tags. Marca é um par de chave-valor que você define para um grupo de logs. Usar tags é uma maneira simples, porém poderosa, de gerenciar AWS recursos e organizar dados, incluindo dados de faturamento.

> **Note**
>
> Você pode usar tags para controlar o acesso aos recursos do CloudWatch Logs, incluindo grupos e destinos de registros. O acesso aos fluxos de log é controlado no nível do grupo de log, devido à relação hierárquica entre grupos de log e fluxos de log. Para obter mais informações sobre como usar etiquetas para controlar o acesso, consulte Controlar acesso a recursos da Amazon Web Services usando etiquetas de recursos.

Conteúdo

- Conceitos Básicos de Tags
- Monitorar custos usando a marcação
- Restrições de tags
- Marcando grupos de registros usando o AWS CLI
- Como marcar grupos de registros usando a API CloudWatch Logs

Conceitos Básicos de Tags

Você usa AWS CloudFormation a API AWS CLI, ou CloudWatch Logs, para concluir as seguintes tarefas:

- Adicione tags a um grupo de logs ao criá-lo.
- Adicione tags a um grupo de logs existente.

- Liste as tags para um grupo de logs.
- Remova tags de um grupo de logs.

Você pode usar marcas para categorizar seus grupos de logs. Por exemplo, você pode categorizá-las por finalidade, proprietário ou ambiente. Como você define a chave e o valor para cada marca, é possível criar um conjunto de categorias personalizado para atender às suas necessidades específicas. Por exemplo, você pode definir um conjunto de marcas que ajude a monitorar os grupos de log por proprietário e aplicação associada. Aqui estão alguns exemplos de tags:

- Projeto: nome do projeto
- Proprietário: nome
- Objetivo: testes de carga
- Aplicação: nome da aplicação
- Ambiente: produção

Monitorar custos usando a marcação

Você pode usar tags para categorizar e monitorar seus AWS custos. Quando você aplica tags aos seus AWS recursos, incluindo grupos de registros, seu relatório de alocação de AWS custos inclui o uso e os custos agregados por tags. É possível aplicar tags que representem categorias de negócios (como centros de custos, nomes de aplicações ou proprietários) para organizar seus custos de vários serviços. Para obter mais informações, consulte Usar etiquetas de alocação de custos para relatórios de faturamento personalizados no Manual do usuário do AWS Billing .

Restrições de tags

As restrições a seguir se aplicam a tags.

Restrições básicas

- O número máximo de marcas por grupo de logs é 50.
- As chaves e valores das tags diferenciam maiúsculas de minúsculas.
- Você não pode alterar nem editar as marcas de um grupo de logs excluído.

Restrições de chaves de marcas

- Cada chave de tag deve ser exclusiva. Se uma tag for adicionada com uma chave que já estiver em uso, a nova tag substituirá o par de chave-valor existente.
- Você não pode iniciar uma chave de tag com `aws:` porque esse prefixo é reservado para uso por AWS. AWS cria tags que começam com esse prefixo em seu nome, mas você não pode editá-las nem excluí-las.
- As chaves de tag devem ter entre 1 e 128 caracteres Unicode.
- As chaves de tag devem conter os seguintes caracteres: letras Unicode, dígitos, espaço em branco e os seguintes caracteres especiais: `_ . / = + - @`.

Restrições de valor das tags

- Os valores das tags devem ter entre 0 e 255 caracteres Unicode.
- Os valores das tags podem estar em branco. Caso contrário, eles devem conter os seguintes caracteres: letras Unicode, dígitos, espaço em branco e qualquer um dos seguintes caracteres especiais: `_ . / = + - @`.

Marcando grupos de registros usando o AWS CLI

É possível adicionar, listar e remover tags usando a AWS CLI. Para obter exemplos, consulte a seguinte documentação:

create-log-group

Cria um grupo de logs. Você também pode adicionar marcas ao criar o grupo de logs.

tag-resource

Atribui uma ou mais tags (pares de valores-chave) ao recurso de registros especificado CloudWatch .

list-tags-for-resource

Exibe as tags associadas a um recurso de CloudWatch registros.

untag-resource

Remove uma ou mais tags do recurso de CloudWatch registros especificado.

Como marcar grupos de registros usando a API CloudWatch Logs

Você pode adicionar, listar e remover tags usando a API CloudWatch Logs. Para obter exemplos, consulte a seguinte documentação:

CreateLogGroup

Cria um grupo de logs. Você também pode adicionar marcas ao criar o grupo de logs.

TagResource

Atribui uma ou mais tags (pares de valores-chave) ao recurso de registros especificado CloudWatch .

ListTagsForResource

Exibe as tags associadas a um recurso de CloudWatch registros.

UntagResource

Remove uma ou mais tags do recurso de CloudWatch registros especificado.

Criptografe dados de registro no CloudWatch Logs usando AWS Key Management Service

Os dados do grupo de registros são sempre criptografados nos CloudWatch registros. Por padrão, o CloudWatch Logs usa criptografia do lado do servidor com o Galois/Counter Modo Padrão de Criptografia Avançada (AES-GCM) de 256 bits para criptografar dados de registro em repouso. Como alternativa, você pode usar o AWS Key Management Service para essa criptografia. Se você fizer isso, a criptografia será feita usando uma AWS KMS chave. O uso da criptografia AWS KMS é habilitado no nível do grupo de registros, associando uma chave KMS a um grupo de registros, seja quando você cria o grupo de registros ou depois que ele existe.

Important

CloudWatch Os registros agora oferecem suporte ao contexto de criptografia, usando `kms:EncryptionContext:aws:logs:arn` como chave e o ARN do grupo de registros como o valor dessa chave. Se você tiver grupos de logs que já criptografou com uma chave do KMS e quiser restringir a chave de modo que ela seja usada com uma única conta e

grupo de logs, atribua uma nova chave do KMS que contenha uma condição na política do IAM. Para obter mais informações, consulte AWS KMS chaves e contexto de criptografia.

Important

CloudWatch O Logs agora suporta `kms:ViaService` o que permite que os registros façam AWS KMS chamadas em seu nome. Você deve adicionar isso às suas funções que chamam CloudWatch Logs na sua Política de Chaves ou no IAM. Para maiores informações, veja kms: ViaService.

Depois que você associar uma chave do KMS a um grupo de logs, todos os novos dados ingeridos para o grupo de logs serão criptografados usando essa chave. Esses dados são armazenados em formato criptografado durante todo o período de retenção. CloudWatch O Logs descriptografa esses dados sempre que solicitados. CloudWatch Os registros devem ter permissões para a chave KMS sempre que dados criptografados forem solicitados.

Se, posteriormente, você desassociar uma chave KMS de um grupo de CloudWatch registros, o Logs criptografará os dados recém-ingeridos usando o método de criptografia padrão do CloudWatch Logs. Todos os dados ingeridos anteriormente que foram criptografados com a chave KMS permanecem criptografados com a chave KMS. CloudWatch Os registros ainda podem retornar esses dados após a desassociação da chave KMS, porque CloudWatch os registros ainda podem continuar referenciando a chave. No entanto, se a chave for desativada posteriormente, o CloudWatch Logs não conseguirá ler os registros que foram criptografados com essa chave.

Important

CloudWatch O Logs suporta somente chaves KMS simétricas. Não use uma chave assimétrica para criptografar os dados em seus grupos de logs. Para obter mais informações, consulte Usar chaves simétricas e assimétricas.

Limites

- Para executar as etapas a seguir, é necessário ter as seguintes permissões: `kms:CreateKey`, `kms:GetKeyPolicy` e `kms:PutKeyPolicy`.

- Depois de associar ou desassociar uma chave de um grupo de logs, pode levar até cinco minutos para que a operação seja efetivada.
- Se você revogar o acesso do CloudWatch Logs a uma chave associada ou excluir uma chave KMS associada, seus dados criptografados no CloudWatch Logs não poderão mais ser recuperados.
- Você não pode associar uma chave KMS a um grupo de registros existente usando o CloudWatch console.

Etapa 1: criar uma AWS KMS chave

Para criar uma chave do KMS, use o seguinte comando create-key:

```
aws kms create-key
```

A saída contém o ID de chave e o nome do recurso da Amazon (ARN) da chave. A seguir está um exemplo de saída:

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

Etapa 2: definir permissões na chave do KMS

Por padrão, todas AWS KMS as chaves são privadas. Somente o proprietário do recurso pode usá-la para criptografar e descriptografar dados. No entanto, o proprietário do recurso pode conceder permissões para acessar a chave do KMS a outros usuários e recursos. Com essa etapa, você concede permissão ao diretor do serviço de CloudWatch registros e à função de chamador para usar a chave. Esse principal de serviço deve estar na mesma AWS região em que a chave KMS está armazenada.

Como prática recomendada, recomendamos que você restrinja o uso da chave KMS somente às AWS contas ou grupos de registros que você especificar.

Primeiro, salve a política padrão para sua chave KMS `policy.json` usando o seguinte get-key-policycomando:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

Abra o arquivo `policy.json` em um editor de texto e adicione a seção em negrito de uma das instruções a seguir. Separe a instrução existente da nova instrução com uma vírgula. Essas declarações usam `Condition` seções para aumentar a segurança da AWS KMS chave. Para obter mais informações, consulte AWS KMS chaves e contexto de criptografia.

A seção `Condition` neste exemplo restringe a chave a um único ARN de grupo de logs.

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"
                }
            }
        }
    ]
}
```

A seção `Condition` deste exemplo limita o uso da chave do AWS KMS à conta especificada, mas ele pode ser usado para qualquer grupo de logs.

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:*"
                }
            }
        }
    ]
}
```

Etapa 3: definir permissões para o diretor do IAM chamador

Adicione permissões ao principal do IAM (usuário ou função) que chamará CloudWatch Logs APIs. As permissões necessárias dependem das operações que o diretor precisa realizar. Você pode adicionar essas permissões na política de AWS KMS chaves ou por meio do IAM na própria função. CloudWatch O Logs é usado `kms:ViaService` para fazer chamadas AWS KMS em nome do cliente. Para maiores informações, veja kms: ViaService.

Permissões para associar uma chave KMS a um grupo de registros

O principal do IAM que chama `CreateLogGroup` com um `kmsKeyId` parâmetro, ou chamadas`AssociateKmsKey`, deve ter `kms:DescribeKey` permissão na chave KMS especificada. Se o chamador não tiver essa permissão, a chamada da API falhará com um`AccessDeniedException`.

O exemplo de declaração de política de chaves a seguir concede as permissões mínimas necessárias para associar uma chave KMS a um grupo de registros:

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account_id:role/role_name"
  },
  "Action": [
    "kms:Describe*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "logs.region.amazonaws.com"
      ]
    }
  }
}
```

Permissões para ler e gravar dados de log criptografados

Se o diretor do IAM também precisar ler ou gravar dados de registro criptografados (por exemplo, ligar para `PutLogEvents` `GetLogEventsFilterLogEvents`, ou `StartQuery` em um grupo de registros criptografado com uma chave KMS gerenciada pelo cliente), o diretor precisará de AWS KMS permissões adicionais. O exemplo de declaração de política de chaves a seguir concede o conjunto completo de permissões necessárias para associar uma chave e ler ou gravar dados de log criptografados:

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

Como prática recomendada, defina o escopo da política somente para as funções que interagirão com grupos de registros AWS KMS criptografados.

Como alternativa, para conceder o conjunto completo de permissões para associar uma chave e ler ou gravar dados de log criptografados via IAM, você pode adicionar a política a seguir à função de chamador. Isso pode ser adicionado a uma política de função existente ou anexado a uma função como uma política separada adicional. Se você usar esse método, como prática recomendada, defina o escopo da política somente para as AWS KMS chaves que serão usadas para criptografia de log. Para obter mais informações, consulte Editar políticas do 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"
        }
    ]
```

```
}
```

Por fim, adicione a política atualizada usando o seguinte put-key-policycomando:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Etapa 4: associar uma chave KMS a um grupo de registros

É possível associar uma chave do KMS a um grupo de logs ao criá-la ou posteriormente.

Para descobrir se um grupo de registros já tem uma chave KMS associada, use o seguinte describe-log-groupscomando:

```
aws logs describe-log-groups --log-group-name-prefix "log-group-name-prefix"
```

Se a saída incluir um campo `kmsKeyId`, o grupo de logs será associado à chave exibida para o valor desse campo.

Para associar a chave do KMS a um grupo de logs ao criá-lo

Use o comando create-log-group da seguinte forma:

```
aws logs create-log-group --log-group-name my-log-group --kms-key-id "key-arn"
```

Para associar a chave do KMS a um grupo de logs existente

Use o comando associate-kms-key da seguinte forma:

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id "key-arn"
```

Etapa 5: Desassociar a chave de um grupo de registros

Para desassociar a chave KMS associada a um grupo de registros, use o seguinte comando: disassociate-kms-key

```
aws logs disassociate-kms-key --log-group-name my-log-group
```

AWS KMS chaves e contexto de criptografia

Para aumentar a segurança de suas AWS Key Management Service chaves e de seus grupos de registros criptografados, o CloudWatch Logs agora coloca o grupo de registros ARNs como parte do contexto de criptografia usado para criptografar seus dados de registro. O contexto de criptografia é um conjunto de pares de chave/valor que são usados como dados autenticados adicionais. O contexto de criptografia permite que você use as condições da política do IAM para limitar o acesso à sua AWS KMS chave por AWS conta e grupo de registros. Para obter mais informações, consulte Contexto de criptografia e Elementos de política JSON do IAM: condição.

Recomendamos usar chaves do KMS diferentes para cada grupo de logs criptografado.

Se você tem um grupo de logs criptografado anteriormente e agora deseja alterar o grupo de logs para usar uma nova chave do KMS que funcione somente para esse grupo de logs, siga estas etapas.

Como converter um grupo de logs criptografado para usar uma chave do KMS com uma política limitando-a a esse grupo de logs

1. Insira o comando a seguir para localizar o ARN da chave atual do grupo de logs:

   ```
   aws logs describe-log-groups
   ```

   A saída inclui a linha a seguir. Anote o ARN. Ele será necessário na etapa 7.

   ```
   ...
   "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/01234567-89ab-
   cdef-0123-456789abcdef"
   ...
   ```

2. Digite o comando a seguir para criar uma nova chave do KMS:

   ```
   aws kms create-key
   ```

3. Digite o comando a seguir para salvar a política da nova chave em um arquivo `policy.json`:

   ```
   aws kms get-key-policy --key-id new-key-id --policy-name default --output text > ./
   policy.json
   ```

4. Use um editor de texto para abrir `policy.json` e adicionar uma expressão `Condition` à política:

   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. Insira o comando a seguir para adicionar a política atualizada à nova chave do KMS:

```
aws kms put-key-policy --key-id new-key-ARN --policy-name default --policy file://
policy.json
```

6. Digite o comando a seguir para associar a política ao seu grupo de logs:

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id new-key-ARN
```

CloudWatch Agora, o Logs criptografa todos os novos dados usando a nova chave.

7. Depois, revogue todas as permissões, exceto `Decrypt` da chave antiga. Primeiro, digite o comando a seguir para recuperar a política antiga:

```
aws kms get-key-policy --key-id old-key-ARN --policy-name default --output text
 > ./policy.json
```

8. Use um editor de texto para abrir `policy.json` e remover todos os valores da lista `Action`, exceto `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. Insira o comando a seguir para adicionar a política atualizada à antiga chave:

```
aws kms put-key-policy --key-id old-key-ARN --policy-name default --policy file://
policy.json
```

Ajude a proteger dados de log confidenciais com mascaramento

Você pode ajudar a proteger dados confidenciais que são ingeridos pelo CloudWatch Logs usando políticas de proteção de dados de grupos de registros. Essas políticas permitem auditar e mascarar dados confidenciais que aparecem nos eventos de log consumidos pelos grupos de logs na sua conta.

Quando você cria uma política de proteção de dados, por padrão, os dados confidenciais que correspondem aos identificadores de dados selecionados são mascarados em todos os pontos de saída, incluindo CloudWatch Logs Insights, filtros métricos e filtros de assinatura. Somente usuários com a permissão do IAM `logs:Unmask` podem visualizar dados não mascarados.

Você pode criar uma política de proteção de dados para todos os grupos de logs da sua conta, além de criar políticas de proteção de dados para grupos de logs individuais. Ao criar uma política para toda a conta, ela se aplica tanto aos grupos de logs existentes quanto aos grupos de logs que forem criados no futuro.

Se você criar uma política de proteção de dados para toda a conta e também criar uma política para um único grupo de logs, as duas políticas serão aplicáveis a esse grupo de logs. Todos os identificadores de dados gerenciados especificados em qualquer política são auditados e mascarados nesse grupo de logs.

> **Note**
>
> O mascaramento de dados confidenciais é suportado para grupos de registros nas classes de registro de acesso padrão e infrequente. Para obter mais informações sobre classes de logs, consulte Classes de logs.

Cada grupo de logs pode ter somente uma política de proteção de dados em nível de grupo de logs, mas essa política pode especificar vários identificadores de dados gerenciados para auditoria e mascaramento. O limite de uma política de proteção de dados é de 30.720 caracteres.

> **Important**
>
> Os dados confidenciais são detectados e mascarados quando são ingeridos no grupo de logs. Quando você define uma política de proteção de dados, os eventos de log ingeridos no grupo de logs antes dessa hora não são mascarados.

CloudWatch O Logs oferece suporte a vários identificadores de dados gerenciados, que oferecem tipos de dados pré-configurados que você pode selecionar para proteger dados financeiros, informações pessoais de saúde (PHI) e informações de identificação pessoal (PII). CloudWatch A proteção de dados de registros permite que você aproveite a correspondência de padrões e os modelos de aprendizado de máquina para detectar dados confidenciais. Para alguns tipos de dados, a detecção depende também de certas palavras-chave serem encontradas próximas aos dados confidenciais. Também é possível usar identificadores de dados personalizados para criar identificadores de dados personalizados para um caso de uso específico.

CloudWatch Quando dados confidenciais são detectados, é emitida uma métrica que corresponde aos identificadores de dados selecionados. Essa é a LogEventsWithFindingsmétrica e é emitida no namespace AWS/Logs. Você pode usar essa métrica para criar CloudWatch alarmes e visualizá-la em gráficos e painéis. As métricas emitidas pela proteção de dados são métricas vendidas e gratuitas. Para obter mais informações sobre as métricas para as CloudWatch quais o Logs envia CloudWatch, consulteMonitoramento com CloudWatch métricas.

Cada identificador de dados gerenciados foi projetado para detectar um tipo específico de dados confidenciais, como números de cartão de crédito, chaves de acesso AWS secretas ou números de passaportes de um determinado país ou região. Ao criar uma política de proteção de dados, você pode configurá-la para usar esses identificadores para analisar logs ingeridos pelo grupo de logs e executar ações quando forem detectados.

CloudWatch A proteção de dados de registros pode detectar as seguintes categorias de dados confidenciais usando identificadores de dados gerenciados:

- Credenciais, como chaves privadas ou chaves de acesso AWS secretas
- Informações financeiras, como números de cartão de crédito

- Informações de identificação pessoal (PII), como carteiras de motorista ou números de previdência social
- Informações de saúde protegidas (PHI), como seguro de saúde ou números de identificação médica
- Identificadores de dispositivos, como endereços IP ou endereços MAC

Para obter detalhes sobre os tipos de dados que você pode proteger, consulte Tipos de dados que você pode proteger.

Sumário

- Noções básicas sobre políticas de proteção de dados
  - O que são políticas de proteção de dados?
  - Como a política de proteção de dados é estruturada?
    - Propriedades JSON para a política de proteção de dados
    - Propriedades JSON para uma declaração de política
    - Propriedades JSON para uma operação de declaração de política
- Permissões de IAM necessárias para criar ou trabalhar com uma política de proteção de dados
  - Permissões necessárias para políticas de proteção de dados no nível da conta
  - Permissões necessárias para políticas de proteção de dados para um único grupo de logs
  - Amostra da política de proteção de dados
- Criar uma política de proteção de dados para toda a conta
  - Console
  - AWS CLI
    - Sintaxe da política de proteção de dados para AWS CLI nossas operações de API
- Criar uma política de proteção de dados para um único grupo de logs
  - Console
  - AWS CLI
    - Sintaxe da política de proteção de dados para AWS CLI nossas operações de API
- Exibir dados não mascarados
- Relatórios de descobertas de auditoria
  - Política-chave necessária para enviar os resultados da auditoria para um bucket protegido por AWS KMS

- Tipos de dados que você pode proteger
  - CloudWatch Registra identificadores de dados gerenciados para tipos de dados confidenciais
    - Credenciais
      - Identificador de dados ARNs para tipos de dados de credenciais
    - Identificadores de dispositivo
      - Identificador de dados ARNs para tipos de dados de dispositivos
    - Informações financeiras
      - Identificador de dados ARNs para tipos de dados financeiros
    - Informações de saúde protegidas (PHI)
      - Identificador de dados ARNs para tipos de dados de informações de saúde protegidas (PHI)
    - Informações de identificação pessoal (PII)
      - Palavras-chave para números de identificação da carteira de habilitação
      - Palavras-chave para números de identificação nacional
      - Palavras-chave para números de passaporte
      - Palavras-chave para identificação do contribuinte e números de referência
      - Identificador de dados ARNs para informações de identificação pessoal (PII)
  - Identificadores de dados personalizados
    - O que são identificadores de dados personalizados?
    - Restrições de identificadores de dados personalizados
    - Uso de identificadores de dados personalizados no console
    - Usar identificadores de dados personalizados na política de proteção de dados

Noções básicas sobre políticas de proteção de dados

Tópicos

- O que são políticas de proteção de dados?
- Como a política de proteção de dados é estruturada?

O que são políticas de proteção de dados?

CloudWatch O Logs usa políticas de proteção de dados para selecionar os dados confidenciais que você deseja verificar e as ações que você deseja tomar para proteger esses dados. Para selecionar os dados confidenciais de interesse, você usa identificadores de dados. CloudWatch A proteção de dados de registros e, em seguida, detecta os dados confidenciais usando aprendizado de máquina e correspondência de padrões. Para agir sobre identificadores de dados encontrados, você pode definir operações de auditoria e desidentificação. Essas operações permitem registrar os dados confidenciais encontrados (ou não encontrados) e mascarar os dados sensíveis quando os eventos de log são exibidos.

Como a política de proteção de dados é estruturada?

Como mostrado na figura abaixo, um documento de política de proteção de dados inclui os seguintes elementos:

- Informações opcionais da política na parte superior do documento
- Uma declaração que define as ações de auditoria e desidentificação

Somente uma política de proteção de dados pode ser definida por grupo de CloudWatch registros de registros. A política de proteção de dados pode ter uma ou mais instruções de negação ou desidentificação, mas somente uma instrução de auditoria.

Propriedades JSON para a política de proteção de dados

Uma política de proteção de dados requer as seguintes informações básicas de política para identificação:

- Nome: o nome da política.
- Descrição (Opcional): a descrição da política.
- Versão: a versão do idioma das políticas. A versão atual é 2021-06-01.
- Declaração: uma lista de declarações que especificam as ações da política de proteção de dados.

```
{
  "Name": "CloudWatchLogs-PersonalInformation-Protection",
  "Description": "Protect basic types of sensitive data",
  "Version": "2021-06-01",
  "Statement": [
```

```
        ...
  ]
}
```

Propriedades JSON para uma declaração de política

Uma declaração de política define o contexto de detecção para a operação de proteção de dados.

- Sid (Opcional): o identificador da declaração.
- DataIdentifier— Os dados confidenciais que os CloudWatch registros devem verificar. Por exemplo, nome, endereço ou número de telefone.
- Operação — As ações subsequentes, seja Auditoria ou Desidentificação. CloudWatch O Logs executa essas ações quando encontra dados confidenciais.

```
{
  ...
  "Statement": [
    {
      "Sid": "audit-policy",
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/Address"
      ],
      "Operation": {
        "Audit": {
          "FindingsDestination": {}
        }
      }
    },
```

Propriedades JSON para uma operação de declaração de política

Uma declaração de política define uma das operações de proteção de dados a seguir.

- Auditoria — emite relatórios de métricas e descobertas sem interromper os logs. As sequências de caracteres correspondentes incrementam a LogEventsWithFindingsmétrica que o CloudWatch Logs publica no namespace AWS/Logs. CloudWatch É possível usar essas métricas para criar alarmes.

  Para obter um exemplo de um relatório de descobertas, consulte Relatórios de descobertas de auditoria.

Para obter mais informações sobre as métricas para as CloudWatch quais o Logs envia CloudWatch, consulteMonitoramento com CloudWatch métricas.

- Desidentificar — mascare os dados confidenciais sem interromper os logs.

Permissões de IAM necessárias para criar ou trabalhar com uma política de proteção de dados

Para poder trabalhar com políticas de proteção de dados para grupos de logs, você deve ter certas permissões, conforme mostrado nas tabelas a seguir. As permissões são diferentes para políticas de proteção de dados em toda a conta e para políticas de proteção de dados aplicáveis a um único grupo de logs.

Permissões necessárias para políticas de proteção de dados no nível da conta

> **Note**
>
> Se você estiver executando qualquer uma dessas operações dentro de uma função do Lambda, a função de execução do Lambda e o limite de permissões também devem incluir as seguintes permissões.

Operation	Permissão necessárias do IAM	Recurso
Criar uma política de proteção de dados sem destinos de auditoria	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
Crie uma política de proteção de dados com o CloudWatch Logs como destino de auditoria	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*

Operation	Permissão necessárias do IAM	Recurso
	`logs:CreateLogDelivery`	*
	`logs:PutResourcePolicy`	*
	`logs:DescribeResourcePolicies`	*
	`logs:DescribeLogGroups`	*
Criar uma política de proteção de dados com o Firehose como destino de auditoria	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*
	`firehose:TagDeliveryStream`	`arn:aws:logs:::deliverystream/` *YOUR_DELIVERY_STREAM*
Criar uma política de proteção de dados com o Amazon S3 como destino de auditoria	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*

Operation	Permissão necessárias do IAM	Recurso
	`s3:GetBucketPolicy`	`arn:aws:s3::: YOUR_BUCKET`
	`s3:PutBucketPolicy`	`arn:aws:s3::: YOUR_BUCKET`
Desmascarar eventos de log mascarados em um grupo de logs especificado	`logs:Unmask`	`arn:aws:logs:::log-group:*`
Ver uma política de proteção de dados existente	`logs:GetDataProtectionPolicy`	`*`
Excluir uma política de proteção de dados	`logs:DeleteAccountPolicy`	`*`
	`logs:DeleteDataProtectionPolicy`	`*`

Se algum log de auditoria de proteção de dados já estiver sendo enviado para um destino, outras políticas que enviam logs para o mesmo destino precisarão apenas das permissões `logs:PutDataProtectionPolicy` e `logs:CreateLogDelivery`.

Permissões necessárias para políticas de proteção de dados para um único grupo de logs

> **Note**
>
> Se você estiver executando qualquer uma dessas operações dentro de uma função do Lambda, a função de execução do Lambda e o limite de permissões também devem incluir as seguintes permissões.

Operation	Permissão necessárias do IAM	Recurso
Criar uma política de proteção de dados sem destinos de auditoria	`logs:PutDataProtectionPolicy`	`arn:aws:logs:::log-group:` *`YOUR_LOG_GROUP`* `:*`

Crie uma política de proteção de dados com o CloudWatch Logs como destino de auditoria	`logs:PutDataProtectionPolicy`
	`logs:CreateLogDelivery`
	`logs:PutResourcePolicy`
	`logs:DescribeResourcePolicies`
	`logs:DescribeLogGroups`
		`arn:aws:logs:::log-group:` *`YOUR_LOG_GROUP`* `:*`
		`*`
		`*`
		`*`
		`*`

Criar uma política de proteção de dados com o Firehose como destino de auditoria	`logs:PutDataProtectionPolicy`
	`logs:CreateLogDelivery`
	`firehose:TagDeliveryStream`
		`arn:aws:logs:::log-group:` *`YOUR_LOG_GROUP`* `:*`
		`*`
		`arn:aws:logs:::deliverystream/` *`YOUR_DELIVERY_STREAM`*

Criar uma política de proteção de dados com o Amazon S3 como destino de auditoria	`logs:PutDataProtectionPolicy`
	`logs:CreateLogDelivery`
	`s3:GetBucketPolicy`
		`arn:aws:logs:::log-group:` *`YOUR_LOG_GROUP`* `:*`
		`*`

Operation	Permissão necessárias do IAM	Recurso
	`s3:PutBucketPolicy`	`arn:aws:s3::: YOUR_BUCKET` `arn:aws:s3::: YOUR_BUCKET`
Desmascarar eventos de logs	`logs:Unmask`	`arn:aws:logs:::log-group: YOUR_LOG_GROUP :*`
Ver uma política de proteção de dados existente	`logs:GetDataProtectionPolicy`	`arn:aws:logs:::log-group: YOUR_LOG_GROUP :*`
Excluir uma política de proteção de dados	`logs:DeleteDataProtectionPolicy`	`arn:aws:logs:::log-group: YOUR_LOG_GROUP :*`

Se algum log de auditoria de proteção de dados já estiver sendo enviado para um destino, outras políticas que enviam logs para o mesmo destino precisarão apenas das permissões `logs:PutDataProtectionPolicy` e `logs:CreateLogDelivery`.

Amostra da política de proteção de dados

O exemplo de política a seguir permite que um usuário crie, visualize e exclua políticas de proteção de dados que podem enviar descobertas de auditoria para todos os três tipos de destinos de auditoria. Ele não permite que o usuário visualize dados não mascarados.

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:*"
            ]
        }
    ]
}
```

Criar uma política de proteção de dados para toda a conta

Você pode usar o console de CloudWatch registros ou AWS CLI os comandos para criar uma política de proteção de dados para mascarar dados confidenciais de todos os grupos de registros em sua conta. Isso afeta os grupos de logs atuais e os grupos de logs que você criar no futuro.

> **Important**
>
> Os dados confidenciais são detectados e mascarados quando são ingeridos no grupo de logs. Quando você define uma política de proteção de dados, os eventos de log ingeridos no grupo de logs antes dessa hora não são mascarados.

Tópicos

- Console
- AWS CLI

Console

Para usar o console a fim de criar uma política de proteção de dados para toda a conta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Configurações. Ele está localizado no final da lista.
3. Escolha a guia Logs.
4. Selecione Configurar.
5. Em Identificadores de dados gerenciados, selecione os tipos de dados que você deseja auditar e mascarar para todos os grupos de logs. Você pode digitar na caixa de seleção para encontrar os identificadores que deseja.

   Recomendamos que você selecione apenas os identificadores de dados relevantes para seus dados de log e sua empresa. A escolha de muitos tipos de dados pode levar a falsos positivos.

   Para obter detalhes sobre quais tipos de dados que você pode proteger, consulte Tipos de dados que você pode proteger.
6. (Opcional) Se você quiser auditar e mascarar outros tipos de dados usando identificadores de dados personalizados, escolha Adicionar identificador de dados personalizado. Em seguida, insira um nome para o tipo de dados e a expressão regular a ser usada para pesquisar esse tipo de dados nos eventos de log. Para obter mais informações, consulte Identificadores de dados personalizados.

Uma única política de proteção de dados pode incluir até 10 identificadores de dados personalizados. Cada expressão regular que define um identificador de dados personalizado deve ter no máximo 200 caracteres.

7. (Opcional) Escolha um ou mais serviços para enviar as descobertas da auditoria. Mesmo se você optar por não enviar descobertas de auditoria a nenhum desses serviços, os tipos de dados confidenciais selecionados ainda serão mascarados.
8. Escolha Activate data protection (Ativar proteção de dados).

AWS CLI

Para usar o AWS CLI para criar uma política de proteção de dados

1. Use um editor de texto para criar um arquivo de política chamado `DataProtectionPolicy.json`. Para obter informações sobre a sintaxe da política, consulte a seção a seguir.
2. Digite o comando:

```
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
```

Sintaxe da política de proteção de dados para AWS CLI nossas operações de API

Quando você cria uma política de proteção de dados JSON para usar em um AWS CLI comando ou operação de API, a política deve incluir dois blocos JSON:

- O primeiro bloco deve incluir uma matriz `DataIdentifer` e uma propriedade `Operation` com uma ação `Audit`. A matriz `DataIdentifer` faz uma lista com os tipos de dados sigilosos que você deseja mascarar. Para obter mais informações sobre as opções disponíveis, consulte Tipos de dados que você pode proteger.

  A propriedade `Operation` com uma ação `Audit` é necessária para encontrar os termos de dados confidenciais. Essa ação `Audit` deve conter um objeto `FindingsDestination`. Opcionalmente, você pode usar esse objeto `FindingsDestination` para listar um ou mais destinos para enviar

relatórios de descobertas de auditoria. Se você especificar destinos como grupos de logs, fluxos do Amazon Data Firehose e buckets do S3, eles já deverão existir. Para obter um exemplo de um relatório de constatações de auditoria, consulte Relatórios de descobertas de auditoria.

- O segundo bloco deve incluir uma matriz `DataIdentifer` e uma propriedade `Operation` com uma ação `Deidentify`. A matriz `DataIdentifer` deve corresponder exatamente à matriz `DataIdentifer` no primeiro bloco da política.

  A propriedade `Operation` com a ação `Deidentify` é o que realmente mascara os dados e deve conter o objeto `"MaskConfig": {}`. O objeto `"MaskConfig": {}` deve estar vazio.

O exemplo de política de proteção de dados a seguir só usa identificadores de dados gerenciados. Essa política mascara os endereços de e-mail e as carteiras de habilitação dos Estados Unidos.

Para obter informações sobre políticas que especificam identificadores de dados personalizados, consulte Usar identificadores de dados personalizados na política de proteção de dados.

```
{
    "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": {}
                }
            }
        }
    ]
}
```

Criar uma política de proteção de dados para um único grupo de logs

Você pode usar o console de CloudWatch registros ou AWS CLI os comandos para criar uma política de proteção de dados para mascarar dados confidenciais.

Você pode atribuir uma política de proteção de dados a cada grupo de logs. Cada política de proteção de dados pode auditar vários tipos de informações. Cada política de proteção de dados pode incluir uma declaração de auditoria.

Tópicos

- Console
- AWS CLI

Console

Para usar o console para criar uma política de proteção de dados

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Escolha o nome do grupo de logs.
4. Escolha Actions (Ações), Create data protection policy (Criar política de proteção de dados).

5. Em Identificadores de dados gerenciados, selecione os tipos de dados que você deseja auditar e mascarar nesse grupo de logs. Você pode digitar na caixa de seleção para encontrar os identificadores que deseja.

   Recomendamos que você selecione apenas os identificadores de dados relevantes para seus dados de log e sua empresa. A escolha de muitos tipos de dados pode levar a falsos positivos.

   Para obter detalhes sobre quais são os tipos de dados que você pode proteger, consulte Tipos de dados que você pode proteger.

6. (Opcional) Se você quiser auditar e mascarar outros tipos de dados usando identificadores de dados personalizados, escolha Adicionar identificador de dados personalizado. Em seguida, insira um nome para o tipo de dados e a expressão regular a ser usada para pesquisar esse tipo de dados nos eventos de log. Para obter mais informações, consulte Identificadores de dados personalizados.

   Uma única política de proteção de dados pode incluir até 10 identificadores de dados personalizados. Cada expressão regular que define um identificador de dados personalizado deve ter no máximo 200 caracteres.

7. (Opcional) Escolha um ou mais serviços para enviar as descobertas da auditoria. Mesmo se você optar por não enviar descobertas de auditoria a nenhum desses serviços, os tipos de dados confidenciais selecionados ainda serão mascarados.

8. Escolha Activate data protection (Ativar proteção de dados).

AWS CLI

Para usar o AWS CLI para criar uma política de proteção de dados

1. Use um editor de texto para criar um arquivo de política chamado `DataProtectionPolicy.json`. Para obter informações sobre a sintaxe da política, consulte a seção a seguir.

2. Digite o comando:

```
aws logs put-data-protection-policy --log-group-identifier "my-log-group" --policy-
document file:///Path/DataProtectionPolicy.json --region us-west-2
```

Sintaxe da política de proteção de dados para AWS CLI nossas operações de API

Quando você cria uma política de proteção de dados JSON para usar em um AWS CLI comando ou operação de API, a política deve incluir dois blocos JSON:

- O primeiro bloco deve incluir uma matriz `DataIdentifer` e uma propriedade `Operation` com uma ação `Audit`. A matriz `DataIdentifer` faz uma lista com os tipos de dados sigilosos que você deseja mascarar. Para obter mais informações sobre as opções disponíveis, consulte Tipos de dados que você pode proteger.

  A propriedade `Operation` com uma ação `Audit` é necessária para encontrar os termos de dados confidenciais. Essa ação `Audit` deve conter um objeto `FindingsDestination`. Opcionalmente, você pode usar esse objeto `FindingsDestination` para listar um ou mais destinos para enviar relatórios de descobertas de auditoria. Se você especificar destinos como grupos de logs, fluxos do Amazon Data Firehose e buckets do S3, eles já deverão existir. Para obter um exemplo de um relatório de constatações de auditoria, consulte Relatórios de descobertas de auditoria.
- O segundo bloco deve incluir uma matriz `DataIdentifer` e uma propriedade `Operation` com uma ação `Deidentify`. A matriz `DataIdentifer` deve corresponder exatamente à matriz `DataIdentifer` no primeiro bloco da política.

  A propriedade `Operation` com a ação `Deidentify` é o que realmente mascara os dados e deve conter o objeto `"MaskConfig": {}`. O objeto `"MaskConfig": {}` deve estar vazio.

Veja a seguir um exemplo de política de proteção de dados que mascara endereços de e-mail e carteiras de habilitação dos Estados Unidos.

```
{
    "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": {}
                }
            }
        }
    ]
}
```

Exibir dados não mascarados

Para visualizar dados não mascarados, o usuário deve ter a permissão `logs:Unmask`. Os usuários com essa permissão podem ver os dados não mascarados das seguintes maneiras:

- Ao visualizar os eventos em um fluxo de logs, escolha Display (Exibir), Unmask (Desmascarar).
- Use uma consulta do CloudWatch Logs Insights que inclua o comando unmask (@message). O exemplo de consulta a seguir exibe os 20 eventos de logs mais recentes no stream, desmascarados:

```
fields @timestamp, @message, unmask(@message)
| sort @timestamp desc
| limit 20
```

Para obter mais informações sobre CloudWatch os comandos do Logs Insights, consulteCloudWatch Sintaxe de consulta de linguagem do Logs Insights.

- Use uma FilterLogEventsoperação GetLogEventsou com o `unmask` parâmetro.

A CloudWatchLogsFullAccesspolítica inclui a `logs:Unmask` permissão. Para conceder `logs:Unmask` a um usuário que não tem CloudWatchLogsFullAccess, você pode anexar uma política personalizada do IAM a esse usuário. Para obter mais informações, consulte Incluindo permissões para um usuário (console).

Relatórios de descobertas de auditoria

Se você configurar as políticas de auditoria de proteção de dados do CloudWatch Logs para escrever relatórios de auditoria no CloudWatch Logs, no Amazon S3 ou no Firehose, esses relatórios de descobertas serão semelhantes ao exemplo a seguir. CloudWatch O Logs grava um relatório de descobertas para cada evento de registro que contém dados confidenciais.

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

Os campos do relatório são os seguintes:

- O campo `resourceArn` exibe o grupo de logs onde os dados confidenciais foram encontrados.
- O objeto `dataIdentifiers` exibe informações sobre as descobertas de um tipo de dados confidenciais que você está auditando.
- O campo `name` identifica o tipo de dados confidenciais sobre os quais esta seção está relatando.
- O campo `count` exibe o número de vezes que esse tipo de dado confidencial aparece no evento de logs.
- Os campos `end` e `start` mostram onde no evento de logs, por contagem de caracteres, cada ocorrência dos dados confidenciais aparece.

O exemplo anterior mostra um relatório de localização de dois endereços de e-mail em um evento de logs. O primeiro endereço de e-mail começa no 13º caractere do evento de logs e termina no 26º caractere. O segundo endereço de e-mail vai do 30º caractere ao 43º caractere. Mesmo que esse evento de log tenha dois endereços de e-mail, o valor da métrica `LogEventsWithFindings` é incrementado apenas em um, porque essa métrica conta o número de eventos de log que contêm dados confidenciais, não o número de ocorrências de dados confidenciais.

Política-chave necessária para enviar os resultados da auditoria para um bucket protegido por AWS KMS

É possível proteger os dados em um bucket do Amazon S3 habilitando a criptografia no lado do servidor com as chaves gerenciadas pelo Amazon S3 (SSE-S3) ou a criptografia no lado do servidor com as chaves do KMS (SSE-KMS). Para obter mais informações, consulte Proteger dados usando criptografia do lado do servidor no Guia do usuário do Amazon S3.

Se você enviar as descobertas de auditoria para um bucket protegido por SSE-S3, nenhuma configuração adicional será necessária. O Amazon S3 lida com a chave de criptografia.

Se você enviar as descobertas de auditoria para um bucket protegido por SSE-KMS, deverá atualizar a política de chaves para a chave do KMS para que a conta de entrega de logs possa gravar no bucket do S3. Para obter mais informações sobre a política de chaves necessária para uso com o SSE-KMS, consulte Amazon S3 o Guia do usuário do Amazon CloudWatch Logs.

Tipos de dados que você pode proteger

Esta seção contém informações sobre os tipos de dados que você pode proteger em uma política de proteção de dados do CloudWatch Logs. CloudWatch Os identificadores de dados gerenciados

do Logs oferecem tipos de dados pré-configurados para proteger dados financeiros, informações pessoais de saúde (PHI) e informações de identificação pessoal (PII). Também é possível usar identificadores de dados personalizados para criar identificadores de dados personalizados para um caso de uso específico.

Sumário

- CloudWatch Registra identificadores de dados gerenciados para tipos de dados confidenciais
  - Credenciais
    - Identificador de dados ARNs para tipos de dados de credenciais
  - Identificadores de dispositivo
    - Identificador de dados ARNs para tipos de dados de dispositivos
  - Informações financeiras
    - Identificador de dados ARNs para tipos de dados financeiros
  - Informações de saúde protegidas (PHI)
    - Identificador de dados ARNs para tipos de dados de informações de saúde protegidas (PHI)
  - Informações de identificação pessoal (PII)
    - Palavras-chave para números de identificação da carteira de habilitação
    - Palavras-chave para números de identificação nacional
    - Palavras-chave para números de passaporte
    - Palavras-chave para identificação do contribuinte e números de referência
    - Identificador de dados ARNs para informações de identificação pessoal (PII)
- Identificadores de dados personalizados
  - O que são identificadores de dados personalizados?
  - Restrições de identificadores de dados personalizados
  - Uso de identificadores de dados personalizados no console
  - Usar identificadores de dados personalizados na política de proteção de dados

CloudWatch Registra identificadores de dados gerenciados para tipos de dados confidenciais

Esta seção contém informações sobre os tipos de dados que você pode proteger usando identificadores de dados gerenciados e quais países e regiões são relevantes para cada tipo de dados.

Para alguns tipos de dados confidenciais, a proteção de dados do CloudWatch Logs verifica as palavras-chave nas proximidades dos dados e encontra uma correspondência somente se encontrar essa palavra-chave. Se uma palavra-chave precisar estar próxima de um tipo específico de dados, a palavra-chave normalmente precisará estar dentro de 30 caracteres (inclusive) dos dados.

Se uma palavra-chave contiver um espaço, a proteção de dados do CloudWatch Logs corresponderá automaticamente às variações de palavras-chave que não têm espaço ou que contêm um sublinhado (_) ou hífen (-) em vez do espaço. Em alguns casos, o CloudWatch Logs também expande ou abrevia uma palavra-chave para abordar variações comuns da palavra-chave.

As tabelas a seguir listam os tipos de credenciais, dispositivos, informações financeiras, médicas e de saúde protegidas (PHI) que o CloudWatch Logs pode detectar usando identificadores de dados gerenciados. Eles são uma adição a determinados tipos de dados que também podem ser qualificados como informações de identificação pessoal (PII).

Identificadores aceitos independentes de idioma e região

Identificador	Categoria
`Address`	Pessoal
`AwsSecretKey`	Credenciais
`CreditCardExpiration`	Financeiro
`CreditCardNumber`	Financeiro
`CreditCardSecurityCode`	Financeiro
`EmailAddress`	Pessoal
`IpAddress`	Pessoal
`LatLong`	Pessoal
`Name`	Pessoal
`OpenSshPrivateKey`	Credenciais
`PgpPrivateKey`	Credenciais

Identificador	Categoria
`PkcsPrivateKey`	Credenciais
`PuttyPrivateKey`	Credenciais
`VehicleIdentificationNumber`	Pessoal

Os identificadores de dados dependentes da região devem incluir o nome do identificador, um hífen e os códigos de duas letras (ISO 3166-1 alfa-2). Por exemplo, `.DriversLicense-US`

Identificadores compatíveis que devem incluir um código de país ou região de duas letras

Identificador	Categoria	Países e idiomas
BankAccountNumber	Financeiro	DE, ES, FR, GB, IT, EUA
CepCode	Pessoal	BR
Cnpj	Pessoal	BR
CpfCode	Pessoal	BR
DriversLicense	Pessoal	AT, AU, BE, BG, CA, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IT, LT, LU, LV, MT, NL, PL, PT, RO, SE, SI, SK, US
DrugEnforcementAgencyNumber	Integridade	EUA
ElectoralRollNumber	Pessoal	GB
HealthInsuranceCardNumber	Integridade	UE
HealthInsuranceClaimNumber	Integridade	EUA
HealthInsuranceNumber	Integridade	FR

Identificador	Categoria	Países e idiomas
HealthcareProcedureCode	Integridade	EUA
IndividualTaxIdentification Number	Pessoal	EUA
InseeCode	Pessoal	FR
MedicareBeneficiaryNumber	Integridade	EUA
NationalDrugCode	Integridade	EUA
NationalIdentificationNumber	Pessoal	DE, ES, IT
NationalInsuranceNumber	Pessoal	GB
NationalProviderId	Integridade	EUA
NhsNumber	Integridade	GB
NieNumber	Pessoal	ES
NifNumber	Pessoal	ES
PassportNumber	Pessoal	CA, DE, ES, FR, GB, IT, US
PermanentResidenceNumber	Pessoal	CA
PersonalHealthNumber	Integridade	CA
PhoneNumber	Pessoal	BR, DE, ES, FR, GB, IT, US
PostalCode	Pessoal	CA
RgNumber	Pessoal	BR
SocialInsuranceNumber	Pessoal	CA
Ssn	Pessoal	ES, US
TaxId	Pessoal	DE, ES, FR, GB

Identificador	Categoria	Países e idiomas
ZipCode	Pessoal	EUA

Credenciais

CloudWatch A proteção de dados de registros pode encontrar os seguintes tipos de credenciais.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões
AWS chave de acesso secreta	`AwsSecretKey`	`aws_secret_access_key` , `credentials` , `secret access key`, `secret key`, `set-awscredential`	Todos
Chave privada OpenSSH	`OpenSSHPrivateKey`	Nenhum	Todos
Chave privada PGP	`PgpPrivateKey`	Nenhum	Todos
Chave privada do Pkcs	`PkcsPrivateKey`	Nenhum	Todos
Chave privada PuTTY	`PuttyPrivateKey`	Nenhum	Todos

Identificador de dados ARNs para tipos de dados de credenciais

A seguir, são listados os nomes de recursos da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

Identificador de dados de credencial ARNs
`arn:aws:dataprotection::aws:data-identifier/AwsSecretKey`

Identificador de dados de credencial ARNs
`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`

Identificadores de dispositivo

CloudWatch A proteção de dados de registros pode encontrar os seguintes tipos de identificadores de dispositivo.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões
IP address (endereço de IP)	`IpAddress`	Nenhum	Todos

Identificador de dados ARNs para tipos de dados de dispositivos

A seguir, são listados os nomes de recursos da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

ARN de identificador de dados de dispositivos
`arn:aws:dataprotection::aws:data-identifier/IpAddress`

Informações financeiras

CloudWatch A proteção de dados de registros pode encontrar os seguintes tipos de informações financeiras.

Se você definir uma política de proteção de dados, o CloudWatch Logs verificará os identificadores de dados que você especificar, independentemente da geolocalização em que o grupo de registros

esteja localizado. As informações na coluna Países e regiões nesta tabela designam se os códigos de país de duas letras devem ser anexados ao identificador de dados para detectar as palavras-chave apropriadas para esses países e regiões.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de conta bancária	`BankAccountNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela Palavras-chave para números de contas bancárias mais adiante nesta seção.	França, Alemanh Itália, Espanha Reino Unido, Estados Unidos	Inclui números de contas bancárias internacionais (IBANs) que consistem em até 34 caracteres alfanuméricos, incluindo elementos como códigos de país.
Data de validade do cartão de crédito	`CreditCardExpiration`	`exp d`, `exp m`, `exp y`, `expiration` , `expiry`	Todos	
Números de cartão de crédito	`CreditCardNumber`	`account number`, `american express`,	Todos	A detecção

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
		`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`		exige que os dados sejam uma sequência de 13 a 19 dígitos que siga a fórmula de cheque de Luhn e use um prefixo de número de cartão padrão para qualquer um dos seguintes tipos de cartão de

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
				crédito: American Express, Dankort, Diner's Club, Discover, Electron, Japanese Card Bureau (JCB), Mastercard e Visa. UnionPay
Código de verificação do cartão de crédito	`CreditCardSecurityCode`	`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`	Todos	

Palavras-chave para números de conta bancária

Use as palavras-chave a seguir para números de contas bancárias. Isso inclui números de contas bancárias internacionais (IBANs) que consistem em até 34 caracteres alfanuméricos, incluindo elementos como códigos de país.

País	Palavras-chave
França	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
Alemanha	account code, account number, accountno# , accountnumber# , bankleitzahl , bban, customer account id, customer account number, customer bank account id, geheimzahl , iban, kartennummer , kontonummer , kreditkartennummer , sepa
Itália	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
Espanha	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
Reino Unido	account code, account number, accountno# , accountnumber# , bban, customer account ID, customer account number, customer bank account id, iban, sepa
Estados Unidos	bank account, bank acct, checking account, checking acct, deposit account, deposit acct, savings account, savings acct, chequing account, chequing acct

CloudWatch Os registros não relatam as ocorrências das seguintes sequências, que os emissores de cartão de crédito reservaram para testes públicos.

```
122000000000003, 2222405343248877, 2222990905257051, 2223007648726984,
 2223577120017656,
30569309025904, 34343434343434, 3528000700000000, 3530111333300000, 3566002020360505,
 36148900647913,
36700102000000, 371449635398431, 378282246310005, 378734493671000, 38520000023237,
 4012888888881881,
4111111111111111, 4222222222222, 4444333322221111, 4462030000000000, 4484070000000000,
 4911830000000,
4917300800000000, 4917610000000000, 4917610000000000003, 5019717010103742,
 5105105105105100,
5111010030175156, 5185540810000019, 5200828282828210, 5204230080000017,
 5204740009900014, 5420923878724339,
5454545454545454, 5455330760000018, 5506900490000436, 5506900490000444,
 5506900510000234, 5506920809243667,
5506922400634930, 5506927427317625, 5553042241984105, 5555553753048194,
 5555555555554444, 5610591081018250,
6011000990139424, 6011000400000000, 6011111111111117, 630490017740292441,
 630495060000000000,
6331101999990016, 6759649826438453, 6799990100000000019, and 76009244561.
```

Identificador de dados ARNs para tipos de dados financeiros

A seguir, são listados os nomes de recursos da Amazon (ARNs) para os identificadores de dados que você pode adicionar às suas políticas de proteção de dados.

Identificador de dados financeiros 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

Identificador de dados financeiros ARNs
`arn:aws:dataprotection::aws:data-identifier/CreditCardSecurityCode`

Informações de saúde protegidas (PHI)

CloudWatch A proteção de dados de registros pode encontrar os seguintes tipos de informações de saúde protegidas (PHI).

Se você definir uma política de proteção de dados, o CloudWatch Logs verificará os identificadores de dados que você especificar, independentemente da geolocalização em que o grupo de registros esteja localizado. As informações na coluna Países e regiões nesta tabela designam se os códigos de país de duas letras devem ser anexados ao identificador de dados para detectar as palavras-chave apropriadas para esses países e regiões.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões
Número de registro da Agência Antidrogas (Drug Enforcement Agency, DEA	`DrugEnforcementAgencyNumber`	`dea number`, `dea registration`	Estados Unidos
Número do cartão de seguro de saúde (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`	União Europeia

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões
		insurance number, insurance card number, krankenversicherungskarte , krankenversicherungsnummer , medical account number, numero conto medico, 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, sairaanhoitokortin , sairausvakuutuskortti , sairausvakuutusnumero , sjukförsäkringnummer, sjukförsäkringskort , suomi ehic-numero , tarjeta de salud, terveyskortti , tessera sanitaria assicurazione numero , versicherungsnummer	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões
Health Insurance Claim Number (HICN)	`HealthInsuranceClaimNumber`	`health insurance claim number`, `hic no`, `hic no.`, `hic number`, `hic#`, `hicn`, `hicn#`, `hicno#`	Estados Unidos
Número de identificação médica ou do seguro de saúde	`HealthInsuranceNumber`	`carte d'assuré social`, `carte vitale`, `insurance card`	França
Código do Healthcare Common Procedure Coding System (HCPCS)	`HealthcareProcedureCode`	`current procedural terminology` , `hcpcs`, `healthcare common procedure coding system`	Estados Unidos
Número do beneficiário do Medicare (MBN)	`MedicareBeneficiaryNumber`	`mbi`, `medicare beneficiary`	Estados Unidos
National Drug Code (NDC)	`NationalDrugCode`	`national drug code`, `ndc`	Estados Unidos
National Provider Identifier (NPI)	`NationalProviderId`	`hipaa`, `n.p.i.`, `national provider`, `npi`	Estados Unidos
Número do Serviço Nacional de Saúde (NHS)	`NhsNumber`	`national health service`, `NHS`	Grã-Bretanha
Número de saúde pessoal	`PersonalHealthNumber`	`canada healthcare number`, `msp number`, `care number`, `phn`, `soins de santé`	Canadá

Identificador de dados ARNs para tipos de dados de informações de saúde protegidas (PHI)

O seguinte lista o identificador de dados Amazon Resource Names (ARNs) que pode ser usado em políticas de proteção de dados de informações de saúde protegidas (PHI).

Identificador de dados PHI ARNs
`arn:aws:dataprotection::aws:data-identifier/DrugEnforcementAgencyNumber-US`
`arn:aws:dataprotection::aws:data-identifier/HealthcareProcedureCode-US`
`arn:aws:dataprotection::aws:data-identifier/HealthInsuranceCardNumber-EU`
`arn:aws:dataprotection::aws:data-identifier/HealthInsuranceClaimNumber-US`
`arn:aws:dataprotection::aws:data-identifier/HealthInsuranceNumber-FR`
`arn:aws:dataprotection::aws:data-identifier/MedicareBeneficiaryNumber-US`
`arn:aws:dataprotection::aws:data-identifier/NationalDrugCode-US`
`arn:aws:dataprotection::aws:data-identifier/NationalInsuranceNumber-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`

Informações de identificação pessoal (PII)

CloudWatch A proteção de dados de registros pode encontrar os seguintes tipos de informações de identificação pessoal (PII).

Se você definir uma política de proteção de dados, o CloudWatch Logs verificará os identificadores de dados que você especificar, independentemente da geolocalização em que o grupo de registros esteja localizado. As informações na coluna Países e regiões nesta tabela designam se os códigos de país de duas letras devem ser anexados ao identificador de dados para detectar as palavras-chave apropriadas para esses países e regiões.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Datas de nascimento	`DateOfBirth`	`dob`, `date of birth`, `birthdate` , `birth date`, `birthday`, `b-day`, `bday`	Any	O suporte inclui a maioria dos formatos de data, como todos os dígitos e combinações de dígitos e nomes de meses. Os componentes de

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
				data podem ser separados por espaços, barras (/) ou hifens (-).
Código de endereçamento postal (CEP)	`CepCode`	`cep`, `código de endereçamento postal`, `codigo de endereçamento postal`	Brasil	
Cadastro Nacional da Pessoa Jurídica (CNPJ)	`Cnpj`	`cadastro nacional da pessoa jurídica`, `cadastro nacional da pessoa juridica`, `cnpj`	Brasil	
Cadastro de Pessoa Física (CPF)	`CpfCode`	`Cadastro de pessoas fisicas`, `cadastro de pessoas físicas`, `cadastro de pessoa física`, `cadastro de pessoa fisica`, `cpf`	Brasil	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de identificação da carteira de habilitação	`DriversLicense`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira de habilitação posteriormente nesta seção.	Muitos países. Para obter detalhes, consulte a tabela de números de identificação da carteira.	
Número de registro eleitoral	`ElectoralRollNumber`	`electoral #`, `electoral number`, `electoral roll #`, `electoral roll no.`, `electoral roll number`, `electoral rollno`	Reino Unido	
Identificação do contribuinte individual	`IndividualTaxIdentificationNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	Brasil, França, Alemanh Espanha Reino Unido	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Instituto Nacional de Estatística e Estudos Econômicos (INSEE)	`InseeCode`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	França	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de identificação nacional	`NationalIdentificationNumber`	Sim. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	Alemanha Itália, Espanha	Isso inclui identificadores do Documento Nacional de Identidade (DNI) (Espanha), códigos fiscais (Itália) e números de carteira de identidade nacional (Alemanha).

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número do Seguro Nacional (NINO)	`NationalInsuranceNumber`	`insurance no.`, `insurance number`, `insurance#` , `national insurance number`, `nationalinsurance#` , `nationalinsurancenumber` , `nin`, `nino`	Reino Unido	–
Número de identidad de extranjero (NIE)	`NieNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	Espanha	
Número de Identificación Fiscal (NIF)	`NifNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	Espanha	
Número de passaporte	`PassportNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela Palavras-chave para números de passaporte mais adiante nesta seção.	Canadá, França, Alemanh Itália, Espanha Reino Unido, Estados	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de residência permanente	`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`	Canadá	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de telefone	PhoneNumber	Brasil: as palavras-chave também incluem: `cel`, `celular`, `fone`, `móvel`, `número residencial`, `numero residencial`, `telefone` Outros: `cell`, `contact`, `fax`, `fax number`, `mobile`, phone, `phone number`, `tel`, `telephone`, `telephone number`	Brasil, Canadá, França, Alemanh Itália, Espanha Reino Unido, Estados Unidos	Isso inclui números gratuitos nos Estados Unidos e números de fax. Se uma palavra-chave estiver próxima dos dados, o número não precisará incluir o código do país. Se uma palavra-chave estiver

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
				próximados dados, o número não precisará incluir o código do país.
CEP	`PostalCode`	Nenhum	Canadá	
Registro Geral (RG)	`RgNumber`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção.	Brasil	
Número do Seguro Social (SIN)	`SocialInsuranceNumber`	`canadian id`, `numéro d'assurance sociale`, `social insurance number`, `sin`	Canadá	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número da Previdência Social (SSN)	`Ssn`	Espanha — `número de la seguridad social`, `social security no.`, `social security no.` `número de la seguridad social`, `social security number`, `socialsecurityno#` , `ssn`, `ssn#` Estados Unidos – `social security`, `ss#`, `ssn`	Espanha Estados Unidos	
Identificação do contribuinte ou número de referência	`TaxId`	Sim. Palavras-chave diferentes se aplicam a diferentes países. Para obter detalhes, consulte a tabela de números de identificação da carteira nesta seção. .	França, Alemanh Espanha Reino Unido	Isso inclui TIN (França); Steueridentifikationsnummer (Alemanha); CIF (Espanha) e TRN, UTR (Reino Unido).
Código postal	`ZipCode`	`zip code`, `zip+4`	Estados Unidos	Código postal dos Estados Unidos.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Endereço postal	`Address`	Nenhum	Austrália, Canadá, França, Alemanh Itália, Espanha Reino Unido, Estados Unidos	Embora uma palavra-chave não seja necessária, a detecção exige que o endereço inclua o nome de uma cidade ou local e um CEP.
Endereço de correio eletrônico	`EmailAddress`	Nenhum	Any	

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Coordenadas do sistema de posicionamento global (GPS)	`LatLong`	`coordinate` , `coordinates` , `latlong`, `latitudelongitude` , `location`, `position`	Any	CloudWatch Os registros podem detectar coordenadas GPS se as coordenadas de latitude e longitude estiverem armazenadas como um par e estiverem no formato de graus decimais (DD), por exemplo, 41,948614, -87,65531

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
				1. O suporte não inclui coordenadas no formato graus, decimais, minutos (DDM), por exemplo, 41°56.9168'N 87°39.3187'W, ou no formato graus, minutos, segundos (DMS), por exemplo, 41°56'55.0104"N 87°39'19.1196"W.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Nome completo	`Name`	Nenhum	Any	CloudWatch Os registros podem detectar somente nomes completos. O suporte é limitado aos conjuntos de caracteres latinos.

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
Número de identificação de veículo (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`	Any	CloudWatch Os registros podem detectar VINs que consistem em uma sequência de 17 caracteres e estão em conformidade com os padrões ISO 3779 e 3780. Esses padrões foram projetados para uso em

Tipo de dados	ID do identificador de dados	Palavra-chave obrigatória	Países e regiões	Observações
				todo o mundo.

Palavras-chave para números de identificação da carteira de habilitação

Para detectar vários tipos de números de identificação da carteira de motorista, o CloudWatch Logs exige que uma palavra-chave esteja próxima dos números. A tabela a seguir lista as palavras-chave que o CloudWatch Logs reconhece para países e regiões específicos.

País ou região	Palavras-chave
Austrália	dl# 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
Áustria	führerschein, fuhrerschein, führerschein republik österreich, fuhrerschein republik osterreich
Bélgica	fuehrerschein, fuehrerschein- nr, fuehrerschheinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrerscheinnummer, führerscheinnummer, numéro permis conduire, permis de conduire, rijbewijs, rijbewijsnummer
Bulgária	превозно средство, свидетелство за управление на моторно, свидетелство за управление на мпс, сумпс, шофьорска книжка

País ou região	Palavras-chave
Canadá	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
Croácia	vozačka dozvola
Chipre	άδεια οδήγησης
República Tcheca	číslo licence, císlo licence řidiče, číslo řidičskéh o průkazu, ovladače lic., povolení k jízdě, povolení řidiče, řidiči povolení, řidičský prúkaz, řidičský průkaz
Dinamarca	kørekort, kørekortnummer
Estônia	juhi litsentsi number, juhiloa number, juhiluba, juhiluba number
Finlândia	ajokortin numero, ajokortti, förare lic., körkort, körkort nummer, kuljettaja lic., permis de conduire
França	permis de conduire
Alemanha	fuehrerschein, fuehrerschein- nr, fuehrersc heinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrersch einnummer, führerscheinnummer
Grécia	δεια οδήγησης, adeia odigisis

País ou região	Palavras-chave
Hungria	illesztőprogramok lic, jogosítvány, jogsi, licencszám, vezető engedély, vezetői engedély
Irlanda	ceadúnas tiomána
Itália	patente di guida, patente di guida numero, patente guida, patente guida numero
Letônia	autovadītāja apliecība, licences numurs, vadītāja apliecība, vadītāja apliecības numurs, vadītāja atļauja, vadītāja licences numurs, vadītāji lic.
Lituânia	vairuotojo pažymėjimas
Luxemburgo	fahrerlaubnis, führerschäin
Malta	liċenzja tas-sewqan
Holanda	permis de conduire, rijbewijs, rijbewijsnummer
Polônia	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
Romênia	numărul permisului de conducere, permis de conducere
Eslováquia	číslo licencie, číslo vodičského preukazu, ovládače lic., povolenia vodičov, povolenie jazdu, povolenie na jazdu, povolenie vodiča, vodičský preukaz

País ou região	Palavras-chave
Eslovênia	vozniško dovoljenje
Espanha	carnet conducer, el carnet de conducer, licencia conducer, licencia de manejo, número carnet conducer, número de carnet de conducer, número de permiso conducer, número de permiso de conducer, número licencia conducer, número permiso conducer, permiso conducción, permiso conducer, permiso de conducción
Suécia	ajokortin numero, dlno# ajokortti, drivere lic., förare lic., körkort, körkort nummer, körkortsn ummer, kuljettajat lic.
Reino Unido	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
Estados Unidos	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

Palavras-chave para números de identificação nacional

Para detectar vários tipos de números de identificação nacional, o CloudWatch Logs exige que uma palavra-chave esteja próxima dos números. Isso inclui identificadores do Documento Nacional de Identidad (DNI) (Espanha), códigos do Instituto Nacional de Estatística e Estudos Econômicos (INSEE) da França, números da carteira de identidade nacional alemã e números do Registro Geral (RG) (Brasil).

A tabela a seguir lista as palavras-chave que o CloudWatch Logs reconhece para países e regiões específicos.

País ou região	Palavras-chave
Brasil	registro geral, rg
França	assurance sociale, carte nationale d’identit é, cni, code sécurité sociale, French social security number, fssn#, insee, insurance 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#
Alemanha	ausweisnummer, id number, identification number, identity number, insurance number, personal id, personalausweis
Itália	codice fiscal, dati anagrafici, ehic, health card, health insurance card, p. iva, partita i.v.a., personal data, tax code, tessera sanitaria
Espanha	dni, dni#, dninúmero#, documento nacional de identidad, identidad único, identidadúnico#, insurance number, national identification number, national identity, nationalid#, nationali dno#, número nacional identidad, personal

País ou região	Palavras-chave
	identification number, personal identity no, unique identity number, uniqueid#

Palavras-chave para números de passaporte

Para detectar vários tipos de números de passaporte, o CloudWatch Logs exige que uma palavra-chave esteja próxima dos números. A tabela a seguir lista as palavras-chave que o CloudWatch Logs reconhece para países e regiões específicos.

País ou região	Palavras-chave
Canadá	passeport, passeport#, passport, passport#, passportno, passportno#
França	numéro de passeport, passeport, passeport #, passeport #, passeportn °, passeport n °, passeportNon, passeport non
Alemanha	ausstellungsdatum, ausstellungsort, geburtsdatum, passport, passports, reisepass, reisepass–nr, reisepassnummer
Itália	italian passport number, numéro passeport , numéro passeport italien, passaporto, passaporto italiana, passaporto numero, passport number, repubblica italiana passaporto
Espanha	españa pasaporte, libreta pasaporte, número pasaporte, pasaporte, passport, passport book, passport no, passport number, spain passport
Reino Unido	passeport #, passeport n °, passeportNon, passeport non, passeportn °, passport #, passport no, passport number, passport#, passportid

País ou região	Palavras-chave
Estados Unidos	passport, travel document

Palavras-chave para identificação do contribuinte e números de referência

Para detectar vários tipos de identificação de contribuintes e números de referência, o CloudWatch Logs exige que uma palavra-chave esteja próxima dos números. A tabela a seguir lista as palavras-chave que o CloudWatch Logs reconhece para países e regiões específicos.

País ou região	Palavras-chave
Brasil	cadastro de pessoa física, cadastro de pessoa fisica, cadastro de pessoas físicas, cadastro de pessoas fisicas, cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj, cpf
França	numéro d'identification fiscale, tax id, tax identification number, tax number, tin, tin#
Alemanha	identifikationsnummer, steuer id, steueride ntifikationsnummer, steuernummer, tax id, tax identification number, tax number
Espanha	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#
Reino Unido	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

País ou região	Palavras-chave
Estados Unidos	número de identificação de contribuinte individual, itin, i.t.i.n.

Identificador de dados ARNs para informações de identificação pessoal (PII)

A tabela a seguir lista os nomes de recursos da Amazon (ARNs) para os identificadores de dados de informações de identificação pessoal (PII) que você pode adicionar às suas políticas de proteção de dados.

Identificador de dados 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`
`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`

Identificador de dados PII ARNs
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
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-RO
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SE
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SI
arn:aws:dataprotection::aws:data-identifier/DriversLicense-SK

Identificador de dados PII ARNs
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
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

Identificador de dados PII ARNs
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
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

Identificador de dados PII ARNs
`arn:aws:dataprotection::aws:data-identifier/VehicleIdentificationNumber`
`arn:aws:dataprotection::aws:data-identifier/ZipCode-US`

Identificadores de dados personalizados

Tópicos

- O que são identificadores de dados personalizados?
- Restrições de identificadores de dados personalizados
- Uso de identificadores de dados personalizados no console
- Usar identificadores de dados personalizados na política de proteção de dados

O que são identificadores de dados personalizados?

Os identificadores de dados personalizados (CDIs) permitem que você defina suas próprias expressões regulares personalizadas que podem ser usadas em sua política de proteção de dados. Usando identificadores de dados personalizados, é possível segmentar casos de uso de informações de identificação pessoal (PII) específicos da empresa que os identificadores de dados gerenciados não podem fornecer. Por exemplo, você pode usar um identificador de dados personalizado para procurar funcionários específicos da empresa. IDs Identificadores de dados personalizados podem ser usados em conjunto com identificadores de dados gerenciados.

Restrições de identificadores de dados personalizados

CloudWatch Os identificadores de dados personalizados de registros têm as seguintes limitações:

- As políticas de proteção de dados são compatíveis com dez identificadores de dados personalizados, no máximo.
- O tamanho máximo dos nomes de identificadores de dados personalizados é 128 caracteres. Os seguintes caracteres são aceitos:
  - Alfanuméricos: (a – z; A – Z; 0 – 9)
  - Símbolos: (“_” | “-”)
- O RegEx tem um tamanho máximo de 200 caracteres. Os seguintes caracteres são aceitos:

  - Alfanuméricos: (a – z; A – Z; 0 – 9)
  - Símbolos: (“_” | “#” | “=” | “@” | “/” | “;” | “,” | “-” | “ ”)
  - Caracteres reservados RegEx: (“^” | “$” | “?” | “[“ | “]” | “{“ | “}” | “|” | “\\” | “*” | “+” | “.”)
- Os identificadores de dados personalizados não podem compartilhar o mesmo nome de um identificador de dados gerenciados.
- Identificadores de dados personalizados podem ser especificados em uma política de proteção de dados ao nível da conta ou em políticas de proteção de dados ao nível do grupo de logs. Como ocorre com identificadores de dados gerenciados, os identificadores de dados personalizados definidos em uma política ao nível da conta funcionam em combinação com identificadores de dados personalizados definidos em uma política ao nível do grupo de logs.

Uso de identificadores de dados personalizados no console

Ao usar o CloudWatch console para criar ou editar uma política de proteção de dados, para especificar um identificador de dados personalizado, basta digitar um nome e uma expressão regular para o identificador de dados. Por exemplo, você pode inserir **`Employee_ID`** para o nome e **`EmployeeID-\d{9}`** para a expressão regular. Essa expressão regular detectará e mascarará eventos de log com nove números depois do `EmployeeID-`. Por exemplo, `EmployeeID-123456789`.

Usar identificadores de dados personalizados na política de proteção de dados

Se você estiver usando a AWS API AWS CLI ou para especificar um identificador de dados personalizado, precisará incluir o nome do identificador de dados e a expressão regular na política JSON usada para definir a política de proteção de dados. A política de proteção de dados a seguir detecta e mascara eventos de registro que transportam funcionários específicos da empresa. IDs

1. Crie um bloco `Configuration` na política de proteção de dados.
2. Insira um `Name` para o identificador de dados personalizado. Por exemplo, .**`EmployeeId`**
3. Insira um `Regex` para o identificador de dados personalizado. Por exemplo, .**`EmployeeID-\d{9}`** Essa expressão regular corresponderá a eventos de log contendo o `EmployeeID-` com nove dígitos depois do `EmployeeID-`. Por exemplo, `EmployeeID-123456789`.
4. Consulte o identificador de dados personalizado a seguir em uma declaração de política.

```
{
    "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. (Opcional) Continue adicionando identificadores de dados personalizados ao bloco `Configuration`, conforme necessário. No momento, as políticas de proteção de dados são compatíveis com dez identificadores de dados personalizados, no máximo.

Transformar logs durante a ingestão

Com a transformação e o enriquecimento dos registros, você pode normalizar todos os seus registros em um formato consistente e contextual no momento da ingestão no Logs. CloudWatch Você pode adicionar estrutura aos seus registros usando modelos pré-configurados para AWS serviços comuns, como AWS WAF o Amazon Route 53, ou criar transformadores personalizados com analisadores nativos, como o Grok. Você também pode renomear atributos existentes e adicionar outros metadados aos seus logs, como ID da conta e região.

A transformação de logs ajuda a simplificar e encurtar suas consultas de logs nos aplicativos e ajuda a simplificar a criação de alertas em seus logs. Esse recurso fornece transformação para tipos de log comuns com modelos de out-of-the-box transformação para as principais fontes de AWS log, como registros de fluxo de VPC, Route 53 e. Amazon RDS for PostgreSQL Você pode usar modelos de transformação pré-configurados ou criar transformadores personalizados para atender às suas necessidades.

A transformação de logs ajuda você a gerenciar logs emitidos de múltiplas fontes que variam muito em formato e nomes de atributos.

Depois de criar um transformador, os eventos de log ingeridos são convertidos e armazenados em um formato padrão. Você pode aproveitar esses logs transformados para acelerar sua experiência de analytics com os seguintes recursos:

- Índices de campo
- CloudWatch Campos descobertos do Logs Insights
- Flexibilidade em alarmes usando filtros métricos
- Encaminhamento por meio de filtros de assinatura
- Criação de dados métricos a partir de eventos de logs com o Contributor Insights, onde você pode optar por fazer com que a regra do Contributor Insights avalie os eventos de logs antes ou depois de serem transformados.

As transformações acontecem somente durante a ingestão de logs. Você não pode transformar eventos de logs que já foram ingeridos. As transformações não são reversíveis. Tanto os registros originais quanto os transformados são armazenados em CloudWatch registros com a mesma política de retenção. A capacidade de transformação e enriquecimento de logs está incluída no preço de

ingestão da classe de log padrão existente. Os custos de armazenamento de logs serão baseados no tamanho do log após a transformação, o que pode exceder o volume original do log.

> **Important**
>
> Depois que os eventos de registro forem transformados, você deverá usar as consultas do CloudWatch Logs Insights para visualizar as versões transformadas dos registros. As FilterLogEventsações GetLogEventse retornam somente as versões originais dos eventos de log, antes de serem transformados.

> **Important**
>
> Apesar de um único evento de log PutLogEvents permitir até 1 MB, a transformação de registros só pode lidar com eventos de log de tamanho menor que 512 kb. Qualquer evento de registro maior que 512 kb falhará na transformação e emitirá um erro. O tamanho total de ainda PutLogEvents pode ultrapassar 512 kb.

Além de se transformar em formatos diferentes, você também pode enriquecer seus logs com contexto adicional, como ID da conta, região e palavra-chave. Eles são extraídos do nome do grupo de logs e de palavras-chave estáticas.

A transformação de logs ajuda você com logs emitidos de múltiplas fontes que variam muito em formato e nomes de atributos.

A transformação e o enriquecimento de logs só são compatíveis com grupos de logs da classe de logs Padrão.

Você pode criar transformadores para grupos de logs individuais, além de criar transformadores no nível da conta que se aplicam a todos ou a muitos grupos de logs da sua conta. Se um grupo de logs tiver um transformador em nível de grupo de logs, esse transformador substituirá qualquer transformador em nível de conta que, de outra forma, se aplicaria a esse grupo de logs.

Tópicos

- Crie e gerencie transformadores de log
- Processadores configuráveis do tipo analisador
- Processadores integrados para AWS registros vendidos

- Processadores de mutação de string
- Processadores de mutação do JSON
- Processadores de conversão de tipos de dados
- Métricas e erros de transformação

Crie e gerencie transformadores de log

Um transformador de log inclui um ou mais processadores que estão juntos em um pipeline lógico. Cada processador é aplicado a um evento de logs, um após o outro, na ordem em que estão listados na configuração do transformador.

Alguns processadores são do tipo analisador. Cada transformador deve ter pelo menos um analisador, e o primeiro processador em um transformador deve ser um analisador.

Alguns dos analisadores são do tipo integrado que são configurados para um determinado tipo de log AWS fornecido.

Outros tipos de processadores são mutadores de string, mutadores JSON e processadores de dados.

Você pode criar transformadores para grupos de logs individuais, além de criar transformadores no nível da conta que se aplicam a todos ou a muitos grupos de logs da sua conta. Se um grupo de logs tiver um transformador em nível de grupo de logs, esse transformador substituirá qualquer transformador em nível de conta que, de outra forma, se aplicaria a esse grupo de logs. Você pode ter até 20 transformadores no nível da conta em uma região na sua conta.

Ao criar um transformador, você deve seguir estas diretrizes:

- Se você incluir um analisador pré-configurado para um tipo de registros AWS vendidos, ele deverá ser o primeiro processador listado no transformador. Você pode incluir apenas um desses processadores em um transformador.
- Você pode incluir apenas um processador `grok` em um transformador.
- Você deve ter pelo menos um processador do tipo analisador em um transformador. Você pode incluir até cinco processadores do tipo analisador. Esse limite de cinco inclui analisadores integrados e analisadores configuráveis.
- Você pode ter até 20 processadores em um transformador.

- Você pode incluir somente um processador addKeys em um transformador.
- Você pode incluir somente um processador copyValue em um transformador.
- Cada transformador pode extrair até 200 campos de um evento de logs.
- Cada evento de registro DEVE estar abaixo de 512 KB. O tamanho total dos eventos de log ainda pode ultrapassar 512 KB.

Tópicos

- Crie uma política de transformador no nível da conta
- Editar ou excluir uma política de transformador em nível de conta
- Crie um transformador de log-group-level toras do zero
- Crie um log-group-level transformador copiando um existente
- Editar um log-group-level transformador
- Excluir um log-group-level transformador

Crie uma política de transformador no nível da conta

Use as etapas desta seção para criar uma política de transformador que se aplique a todos os grupos de logs na conta ou a vários grupos de logs que tenham nomes que comecem com a mesma string (prefixo). Você pode ter até 20 políticas de transformador no nível da conta em uma região.

Você não pode criar duas políticas de transformador na mesma região que usem o mesmo prefixo ou tenham um prefixo contido em outra. Por exemplo, se você criar uma política de transformador para o prefixo de string `/aws/lambda`, não poderá criar outra com o prefixo `/aws`. Mas você poderia ter um transformador para `/aws/lambda` e outro para `/aws/waf`

Para criar uma política de transformador no nível da conta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação à esquerda, escolha Configurações e, em seguida, escolha Logs.
3. Na seção Política de transformador para conta, escolha Criar política de transformador.
4. Em Nome da política do transformador, insira um nome para a sua nova política.
5. Em Selecionar grupos de logs, execute uma das seguintes ações:
    - Escolha Todos os grupos de logs padrão para que a política do transformador se aplique a todos os grupos de logs da Classe Padrão na conta.

- escolha Grupos de logs por correspondência de prefixo para aplicar a política a um subconjunto de grupos de logs, todos com nomes que começam com a mesma string. Em seguida, insira o prefixo desses grupos de logs em Critérios de seleção.

6. Na área Selecionar analisadores, use Analisadores para selecionar um analisador a ser incluído em seu transformador.

   Se for um analisador pré-configurado para um tipo de registro AWS vendido, você não precisa especificar nenhuma configuração para ele.

   Se for um analisador diferente, é necessário especificar sua configuração. Para obter mais informações, consulte as informações para esse processador em Processadores configuráveis do tipo analisador.

7. Para adicionar outro processador, escolha Selecionar processador. Em seguida, selecione o processador desejado na caixa Processador e preencha os parâmetros de configuração.

   Lembre-se de que os processadores operam nos eventos de logs na ordem em que você os adiciona ao transformador.

8. (Opcional) Para adicionar mais processadores, escolha + Processador e repita a etapa anterior.

9. (Opcional) A qualquer momento, você pode testar o transformador construído até agora em um evento de logs de amostra. Para fazer isso, siga um dos seguintes procedimentos na seção Visualização do transformador:

   - Selecione até cinco grupos de logs em Selecionar grupos de logs e, em seguida, escolha Carregar eventos de logs mais recentes. Em seguida, escolha Testar transformador.
   - Copie eventos de logs diretamente em Eventos de logs de amostra e escolha Testar transformador.

   Em seguida, a versão transformada do log é exibida.

10. Quando terminar de adicionar processadores e estiver satisfeito com os testes nos logs de amostra, escolha Salvar.

11. Quando terminar, selecione Create (Criar).

Editar ou excluir uma política de transformador em nível de conta

Use as etapas nesta seção para editar ou excluir uma política de transformação no nível da conta.

Para editar ou excluir uma política de transformador em nível de conta

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação à esquerda, escolha Configurações e, em seguida, escolha Logs.
3. Na seção Polícia da conta do transformador, selecione Gerenciar.
4. Selecione o botão ao lado da política de transformador que você deseja gerenciar e, em seguida, escolha Editar ou Excluir.

   Se você estiver editando a política, consulte as etapas 5 a 11 em Processadores configuráveis do tipo analisador para ver suas opções.

Crie um transformador de log-group-level toras do zero

Use essas etapas para criar um log-group-level transformador do zero.

Para usar o console para criar um transformador de logs para um grupo de logs

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Escolha o grupo de logs para o qual você deseja criar o transformador.
4. Escolha a guia Transformador. Talvez seja necessário rolar a lista de guias para a direita para vê-la.
5. Escolha Criar transformador.
6. Na caixa Escolha um analisador, selecione um analisador para incluir em seu transformador.

   Se for um analisador pré-configurado para um tipo de registro AWS vendido, você não precisa especificar nenhuma configuração para ele.

   Se for um analisador diferente, é necessário especificar sua configuração. Para obter mais informações, consulte as informações para esse processador em Processadores configuráveis do tipo analisador.
7. Para adicionar outro processador, escolha + Adicionar processador. Em seguida, selecione o processador desejado na caixa Escolher processadores e preencha os parâmetros de configuração.

   Lembre-se de que os processadores operam nos eventos de logs na ordem em que você os adiciona ao transformador.

8. (Opcional) A qualquer momento, você pode testar o transformador construído até agora em um evento de logs de amostra. Para isso, faça o seguinte:

   - Na seção Visualização da transformação, escolha Carregar log de amostra para carregar um evento de logs de amostra do grupo de logs ao qual esse transformador se destina ou cole um evento de logs na caixa de texto.

     Escolha Testar transformador. A versão transformada do log é exibida

9. Quando terminar de adicionar processadores e estiver satisfeito com os testes nos logs de amostra, escolha Salvar.

Para usar o AWS CLI para criar um transformador de log do zero

- Use o comando `aws logs put-transformer`. Ao usar o `parseJSON` como primeiro processador, você deve analisar todo o evento de logs usando `@message` como campo de origem. Após a análise inicial do JSON, você pode manipular campos específicos nos processadores subsequentes. Veja a seguir um exemplo que cria um transformador que inclui os processadores `parseJSON` e `addKeys`:

```
 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
```

Crie um log-group-level transformador copiando um existente

Você pode usar o console para copiar a configuração JSON de um transformador existente. Em seguida, você pode usar esse código para criar um transformador idêntico usando o AWS CLI, ou você pode modificar a configuração primeiro.

Para criar um transformador de logs copiando um existente

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Escolha o grupo de logs que contém o transformador que você deseja copiar.

4. Escolha a guia Transformações. Talvez seja necessário rolar a lista de guias para a direita para vê-la.
5. Escolha Gerenciar transformador.
6. Escolha Copiar transformador. Isso copia o transformador JSON para a área de transferência.
7. Crie um arquivo e cole na configuração do transformador. Neste exemplo, chamamos o arquivo `CopiedTransformer.json`
8. Use o AWS CLI para criar um novo transformador com essa configuração.

```
aws logs put-transformer --log-group-identifier my-log-group-name \
--transformer-config file://CopiedTransformer.json
```

Editar um log-group-level transformador

Use essas etapas para editar um transformador de logs existente.

Para editar um transformador de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Escolha o grupo de logs que tem o transformador que você deseja editar.
4. Escolha a guia Transformações. Talvez seja necessário rolar a lista de guias para a direita para vê-la.
5. Escolha Gerenciar transformador.
6. Faça suas alterações nas seções Analisadores e Processadores.
7. Para adicionar outro processador, escolha + Adicionar processador. Em seguida, selecione o processador desejado na caixa Processador e preencha os parâmetros de configuração.

   Lembre-se de que os processadores operam nos eventos de logs na ordem em que você os adiciona ao transformador.
8. (Opcional) A qualquer momento, você pode testar o transformador construído até agora em um evento de logs de amostra. Para isso, faça o seguinte:

   - Na seção Visualização da transformação, escolha Carregar log de amostra para carregar um evento de logs de amostra do grupo de logs ao qual esse transformador se destina ou cole um evento de logs na caixa de texto.

Escolha Testar transformação. A versão transformada do log é exibida

9. Quando terminar de adicionar processadores e estiver satisfeito com os testes nos logs de amostra, escolha Salvar.

Excluir um log-group-level transformador

Use essas etapas para excluir um transformador de log.

Para excluir um transformador de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Logs, Grupos de log.
3. Escolha o grupo de logs que tem o transformador que você deseja editar.
4. Escolha a guia Transformações. Talvez seja necessário rolar a lista de guias para a direita para vê-la.
5. Escolha Excluir.
6. Na caixa de confirmação, escolha Excluir política.

Processadores configuráveis do tipo analisador

Esta seção contém informações sobre os processadores de analisadores de dados configuráveis que você pode usar em um transformador de eventos de log.

Sumário

- parseJSON
- grok
  - Exemplos do Grok
    - Exemplo 1: use o Grok para extrair um campo de logs não estruturados
    - Exemplo 2: use o Grok em combinação com parseJSON para extrair campos de um evento de logs JSON
    - Exemplo 3: padrão Grok com anotação pontilhada em FIELD_NAME
  - Padrões Grok compatíveis
    - Exemplos comuns de formato de log

    - Exemplo de log do Apache
    - Exemplo de log do NGINX
    - Exemplo de log do protocolo Syslog (RFC 5424)
- csv
- parseKeyValue

parseJSON

O processador parseJSON analisa os eventos de logs JSON e insere os pares de valores-chave JSON extraídos no destino. Se você não especificar um destino, o processador colocará o par de valores-chave sob o nó raiz. Ao usar o `parseJSON` como primeiro processador, você deve analisar todo o evento de logs usando `@message` como campo de origem. Após a análise inicial do JSON, você pode manipular campos específicos nos processadores subsequentes.

O conteúdo `@message` original não é alterado, as novas chaves são adicionadas à mensagem.

Campo	Description	Obrigatório?	Padrão	Limites
origem	Caminho para o campo no evento de logs que será analisado. Use a notação de pontos para acessar os campos secundários. Por exemplo, `store.book` .	Não	`@message`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
destination	O campo de destino do JSON analisado	Não	`Parent JSON node`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3

Exemplo

Suponha que um evento de logs ingerido seja semelhante a:

```
{
```

```
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

Então, se tivermos esse processador parseJSON:

```
[
   {
        "parseJSON": {
            "destination": "new_key"
        }
   }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "new_key": {
        "outer_key": {
            "inner_key": "inner_value"
        }
    }
}
```

grok

Use o processador Grok para analisar e estruturar dados não estruturados usando correspondência de padrões. Esse processador também pode extrair campos de mensagens de log.

Campo	Description	Obrigatório?	Padrão	Limites	Observações
origem	Caminho do campo no qual aplicar a correspondência Grok	Não	`@messag`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3	

Campo	Description	Obrigatório?	Padrão	Limites	Observações
match	O padrão Grok para combinar com o evento de logs	Sim		Tamanho máximo: 512 Máximo de padrões Grok: 20 Alguns tipos de padrões Grok têm limites de uso individuais. Qualquer combinação dos seguintes padrões pode ser usada até cinco vezes: {URI, URIPARAM, URIPATHPARAM, SPACE, DATA, GREEDYDATA, GREEDYDATA_MULTILINE} Os padrões Grok não oferecem suporte a conversões de tipos. Para padrões de formato de log comuns (APACHE_ACCESS_LOG, NGINX_ACCESS_LOG,	Veja todos os padrões Grok compatíveis

Campo	Description	Obrigatório?	Padrão	Limites	Observações
				SYSLOG5424), somente os padrões DATA, GREEDYDATA ou GREEDYDATA_MULTILINE podem ser incluídos após o padrão de log comum.	

Estrutura de um padrão Grok

Esta é a estrutura do padrão Grok compatível:

```
%{PATTERN_NAME:FIELD_NAME}
```

- PATTERN_NAME: refere-se a uma expressão regular predefinida para corresponder a um tipo específico de dados. Somente padrões grok predefinidos são suportados. Não é permitido criar padrões personalizados.
- FIELD_NAME: atribui um nome ao valor extraído. `FIELD_NAME` é opcional, mas se você não especificar esse valor, os dados extraídos serão eliminados do evento de logs transformado. Se `FIELD_NAME` usa notação pontilhada (por exemplo, “parent.child”), ele é considerado um caminho JSON.
- Conversão de tipo: conversões de tipo explícitas não são suportadas. Use o TypeConverter processador para converter o tipo de dados de qualquer valor extraído pelo grok.

Para criar expressões de correspondência mais complexas, você pode combinar vários padrões Grok. Até 20 padrões Grok podem ser combinados para corresponder a um evento de logs. Por exemplo, essa combinação de padrões `%{NUMBER:timestamp} [%{NUMBER:db} %{IP:client_ip}:%{NUMBER:client_port}] %{GREEDYDATA:data}` pode ser usada para extrair campos de uma entrada de log lenta do Redis, como esta:

```
1629860738.123456 [0 127.0.0.1:6379] "SET" "key1" "value1"
```

Exemplos do Grok

Exemplo 1: use o Grok para extrair um campo de logs não estruturados

Log de amostra:

```
293750 server-01.internal-network.local OK "[Thread-000] token generated"
```

Transformador usado:

```
[
     {
         "grok": {
             "match": "%{NUMBER:version} %{HOSTNAME:hostname} %{NOTSPACE:status}
 %{QUOTEDSTRING:logMsg}"
         }
    }
]
```

Saída:

```
{
  "version": "293750",
  "hostname": "server-01.internal-network.local",
  "status": "OK",
  "logMsg": "[Thread-000] token generated"
}
```

Log de amostra:

```
23/Nov/2024:10:25:15 -0900 172.16.0.1 200
```

Transformador usado:

```
[
    {
        "grok": {
            "match": "%{HTTPDATE:timestamp} %{IPORHOST:clientip}
 %{NUMBER:response_status}"
        }
```

```
    }
]
```

Saída:

```
{
  "timestamp": "23/Nov/2024:10:25:15 -0900",
  "clientip": "172.16.0.1",
  "response_status": "200"
}
```

Exemplo 2: use o Grok em combinação com parseJSON para extrair campos de um evento de logs JSON

Log de amostra:

```
{
    "timestamp": "2024-11-23T16:03:12Z",
    "level": "ERROR",
    "logMsg": "GET /page.html HTTP/1.1"
}
```

Transformador usado:

```
[
     {
        "parseJSON": {}
    },
    {
         "grok": {
            "source": "logMsg",
             "match": "%{WORD:http_method} %{NOTSPACE:request} HTTP/
%{NUMBER:http_version}"
         }
    }
]
```

Saída:

```
{
  "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"
}
```

Exemplo 3: padrão Grok com anotação pontilhada em FIELD_NAME

Log de amostra:

```
192.168.1.1 GET /index.html?param=value 200 1234
```

Transformador usado:

```
[
    {
        "grok": {
            "match": "%{IP:client.ip} %{WORD:method} %{URIPATHPARAM:request.uri}
 %{NUMBER:response.status} %{NUMBER:response.bytes}"
        }
    }
]
```

Saída:

```
{
  "client": {
    "ip": "192.168.1.1"
  },
  "method": "GET",
  "request": {
    "uri": "/index.html?param=value"
  },
  "response": {
    "status": "200",
    "bytes": "1234"
  }
}
```

Padrões Grok compatíveis

As tabelas a seguir listam os padrões que são compatíveis com o processador `grok`.

Padrões Grok gerais

Padrão Grok	Description	Limite máximo de padrões	Exemplo
USERNAME ou USER	Corresponde a um ou mais caracteres que podem incluir letras minúsculas (a-z), letras maiúsculas (A-Z), dígitos (0-9), pontos (.), sublinhados (_) ou hifens (-).	20	Entrada: `user123.name-TEST` Padrão: `%{USERNAME:name}` Saída: `{"name": "user123.name-TEST"}`
INT	Corresponde a um sinal opcional de mais ou menos seguido por um ou mais dígitos.	20	Entrada: `-456` Padrão: `%{INT:num}` Saída: `{"num": "-456"}`
BASE10NUM	Corresponde a um número inteiro ou de ponto flutuante com sinal e ponto decimal opcionais	20	Entrada: `-0.67` Padrão: `%{BASE10NUM:num}` Saída: `{"num": "-0.67"}`
BASE16NUM	Combina números decimais e hexadecimais com um sinal opcional (+ ou -) e um prefixo 0x opcional	20	Entrada: `+0xA1B2` Padrão: `%{BASE16NUM:num}` Saída: `{"num": "+0xA1B2"}`
POSINT	Corresponde a números inteiros positivos sem zeros à esquerda, consistindo em um	20	Entrada: `123` Padrão: `%{POSINT:num}` Saída: `{"num": "123"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
	ou mais dígitos (1-9 seguido por 0-9)		
NONNEGINT	Corresponde a qualquer número inteiro (consistindo em um ou mais dígitos de 0 a 9), incluindo zero e números com zeros à esquerda.	20	Entrada: `007` Padrão: `%{NONNEGINT:num}` Saída: `{"num": "007"}`
WORD	Corresponde a palavras inteiras compostas por um ou mais caracteres (\w), incluindo letras, dígitos e sublinhados	20	Entrada: `user_123` Padrão: `%{WORD:user}` Saída: `{"user": "user_123"}`
NOTSPACE	Corresponde a um ou mais caracteres que não sejam espaços em branco.	5	Entrada: `hello_world123` Padrão: `%{NOTSPACE:msg}` Saída: `{"msg": "hello_world123"}`
SPACE	Corresponde a zero ou mais caracteres de espaço em branco.	5	Entrada: `" "` Padrão: `%{SPACE:extra}` Saída: `{"extra": " "}`
DATA	Corresponde a qualquer caractere (exceto nova linha) zero ou mais vezes, de forma não gananciosa.	5	Entrada: `abc def ghi` Padrão: `%{DATA:x} %{DATA:y}` Saída: `{"x": "abc", "y": "def ghi"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
GREEDYDATA	Corresponde a qualquer caractere (exceto nova linha) zero ou mais vezes, de forma gananciosa.	5	Entrada: `abc def ghi` Padrão: `%{GREEDYDATA:x} %{GREEDYDATA:y}` Saída: `{"x": "abc def", "y": "ghi"}`

GREEDYDATA_MULTILINE	Corresponde a qualquer caractere (incluindo nova linha) zero ou mais vezes, de forma gananciosa.	1	Entrada:
			
			`abc`
			
			`def`
			
			`ghi`
			
			Padrão: `%{GREEDYDATA_MULTILINE:data}`
			
			Saída: `{"data": "abc\ndef\nghi"}`

QUOTEDSTRING	Combina cadeias de caracteres entre aspas (aspas simples ou duplas) com caracteres de escape.	20	Entrada: `"Hello, world!"` Padrão: `%{QUOTEDSTRING:msg}` Saída: `{"msg": "Hello, world!"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
UUID	Corresponde a um formato UUID padrão: 8 caracteres hexadecimais, seguidos por três grupos de 4 caracteres hexadecimais e terminando com 12 caracteres hexadecimais, todos separados por hifens.	20	Entrada: `550e8400-e29b-41d4-a716-446655440000` Padrão: `%{UUID:id}` Saída: `{"id": "550e8400-e29b-41d4-a716-446655440000"}`
URN	Corresponde à sintaxe URN (Uniform Resource Name).	20	Entrada: `urn:isbn:0451450523` Padrão: `%{URN:urn}` Saída: `{"urn": "urn:isbn:0451450523"}`

AWS padrões grok

Padrão	Description	Limite máximo de padrões	Exemplo
ARN	Corresponde aos nomes de recursos da AWS Amazon (ARNs), capturando a partição (`aws`,`aws-cn`, ou`aws-us-gov`), o serviço, a região, o ID da conta e até 5 identificadores hierárquicos de recursos separados por barras. Não corresponderá às	5	Entrada: `arn:aws:iam:us-east-1:123456789012:user/johndoe` Padrão: `%{ARN:arn}` Saída: `{"arn": "arn:aws:iam:us-east-1:1234`

Padrão	Description	Limite máximo de padrões	Exemplo
	informações ARNs que faltam entre dois pontos.		`56789012:user/john doe"}`

Padrões Grok de redes

Padrão Grok	Description	Limite máximo de padrões	Exemplo
CISCOMAC	Corresponde a um endereço MAC no formato hexadecimal 4-4-4.	20	Entrada: `0123.4567.89AB` Padrão: `%{CISCOMAC:MacAddress}` Saída: `{"MacAddress": "0123.4567.89AB"}`
WINDOWSMAC	Corresponde a um endereço MAC em formato hexadecimal com hifens	20	Entrada: `01-23-45-67-89-AB` Padrão: `%{WINDOWSMAC:MacAddress}` Saída: `{"MacAddress": "01-23-45-67-89-AB"}`
COMMONMAC	Corresponde a um endereço MAC em formato hexadecimal com dois pontos.	20	Entrada: `01:23:45:67:89:AB` Padrão: `%{COMMONMAC:MacAddress}` Saída: `{"MacAddress": "01:23:45:67:89:AB"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
Mac	Corresponde a um dos padrões Grok CISCOMAC, WINDOWSMAC ou COMMONMAC	20	Entrada: `01:23:45:67:89:AB` Padrão: `%{MAC:m1}` Saída: `{"m1":"01:23:45:67:89:AB"}`
IPV6	Corresponde a IPv6 endereços, incluindo formulários compactados e endereços IPv4 mapeados IPv6 .	5	Entrada: `2001:db8:3333:4444:5555:6666:7777:8888` Padrão: `%{IPV6:ip}` Saída: `{"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}`
IPV4	Corresponde a um IPv4 endereço.	20	Entrada: `192.168.0.1` Padrão: `%{IPV4:ip}` Saída: `{"ip": "192.168.0.1"}`
IP	Corresponde aos IPv6 endereços suportados por% {IPv6} ou aos IPv4 endereços compatíveis com% {IPv4}	5	Entrada: `192.168.0.1` Padrão: `%{IP:ip}` Saída: `{"ip": "192.168.0.1"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
HOSTNAME ou HOST	Corresponde a nomes de domínio, incluindo subdomínios	5	Entrada: `server-01.internal-network.local` Padrão: `%{HOST:host}` Saída: `{"host": "server-01.internal-network.local"}`
IPORHOST	Corresponde a um nome de host ou endereço IP	5	Entrada: `2001:db8:3333:4444:5555:6666:7777:8888` Padrão: `%{IPORHOST:ip}` Saída: `{"ip": "2001:db8:3333:4444:5555:6666:7777:8888"}`
HOSTPORT	Corresponde a um endereço IP ou nome de host, conforme suportado pelo padrão %{IPORHOST} seguido por dois pontos e um número de porta, capturando a porta como “PORT” na saída.	5	Entrada: `192.168.1.1:8080` Padrão: `%{HOSTPORT:ip}` Saída: `{"ip":"192.168.1.1:8080","PORT":"8080"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
URIHOST	Corresponde a um endereço IP ou nome de host, conforme suportado pelo padrão %{IPORHOST}, opcionalm ente seguido por dois pontos e um número de porta, capturando a porta como “port”, se presente.	5	Entrada: `example.com:443 10.0.0.1` Padrão: `%{URIHOST:host} %{URIHOST:ip}` Saída: `{"host":" example.com:443","port":"443","ip":" 10.0.0.1"}`

Padrões Grok de caminho

Padrão Grok	Description	Limite máximo de padrões	Exemplo
UNIXPATH	Corresponde aos caminhos de URL, incluindo potencialmente parâmetros de consulta.	20	Entrada: `/search?q=regex` Padrão: `%{UNIXPAT H:path}` Saída: `{"path":"/ search?q=regex"}`
WINPATH	Corresponde aos caminhos dos arquivos do Windows.	5	Entrada: `C:\Users\John \Documents\file.txt` Padrão: `%{WINPATH:path}` Saída: `{"path": "C: \\Users\\John\\ Documents\\file.tx t"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
PATH	Corresponde aos caminhos de arquivo do URL ou do Windows	5	Entrada: `/search?q=regex` Padrão: `%{PATH:path}` Saída: `{"path":"/search?q=regex"}`
TTY	Combina caminhos de dispositivos Unix para terminais e pseudoterminais.	20	Entrada: `/dev/tty1` Padrão: `%{TTY:path}` Saída: `{"path":"/dev/tty1"}`
URIPROTO	Corresponde a letras, opcionalmente seguidas por um caractere de adição (+) e letras adicionais ou caracteres de adição (+)	20	Entrada: `web+transformer` Padrão: `%{URIPROTO:protocol}` Saída: `{"protocol":"web+transformer"}`
URIPATH	Corresponde ao componente de caminho de um URI	20	Entrada: `/category/sub-category/product_name` Padrão: `%{URIPATH:path}` Saída: `{"path":"/category/sub-category/product_name"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
URIPARAM	Corresponde aos parâmetros de consulta de URL	5	Entrada: `?param1=value1&param2=value2` Padrão: `%{URIPARAM:url}` Saída: `{"url":"?param1=value1&param2=value2"}`
URIPATHPARAM	Corresponde a um caminho de URI, opcionalmente seguido por parâmetros de consulta	5	Entrada: `/category/sub-category/product?id=12345&color=red` Padrão: `%{URIPATHPARAM:path}` Saída: `{"path":"/category/sub-category/product?id=12345&color=red"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
URI	Corresponde a um URI completo	5	Entrada: `https://user:password@example.com/path/to/resource?param1=value1&param2=value2` Padrão: `%{URI:uri}` Saída: `{"path":"https://user:password@example.com/path/to/resource?param1=value1&param2=value2"}`

Padrões Grok de data e hora

Padrão Grok	Description	Limite máximo de padrões	Exemplo
MONTH	Combina nomes completos ou abreviados dos meses em inglês como palavras inteiras	20	Entrada: `Jan` Padrão: `%{MONTH:month}` Saída: `{"month": "Jan"}` Entrada: `January` Padrão: `%{MONTH:month}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
			Saída: `{"month": "January"}`
MONTHNUM	Corresponde aos números dos meses de 1 a 12, com zero inicial opcional para meses de um dígito.	20	Entrada: `5` Padrão: `%{MONTHNUM:month}` Saída: `{"month":"5"}` Entrada: `05` Padrão: `%{MONTHNUM:month}` Saída: `{"month": "05"}`
MONTHNUM	Corresponde a números de meses de dois dígitos de 01 a 12.	20	Entrada: `05` Padrão: `%{MONTHNUM2:month}` Saída: `{"month": "05"}`
MONTHDAY	Corresponde ao dia do mês de 1 a 31, com zero inicial opcional.	20	Entrada: `31` Padrão: `%{MONTHDAY:monthDay}` Saída: `{"monthDay":"31"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
YEAR	Corresponde ao ano em dois ou quatro dígitos	20	Entrada: `2024` Padrão: `%{YEAR:year}` Saída: `{"year":"2024"}` Entrada: `24` Padrão: `%{YEAR:year}` Saída: `{"year":"24"}`
DAY	Corresponde aos nomes dos dias completos ou abreviados.	20	Entrada: `Tuesday` Padrão: `%{DAY:day}` Saída: `{"day":"Tuesday"}`
HOUR	Corresponde à hora no formato de 24 horas com um zero (0)0-23 à esquerda opcional.	20	Entrada: `22` Padrão: `%{HOUR:hour}` Saída: `{"hour":"22"}`
MINUTE	Corresponde aos minutos (00-59).	20	Entrada: `59` Padrão: `%{MINUTE:min}` Saída: `{"min":"59"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
SECOND	Corresponde a um número que represent a os segundos (0)0-60, opcionalmente seguido por um ponto decimal ou dois pontos e um ou mais dígitos para minutos fracionários	20	Entrada: `3` Padrão: `%{SECOND: second}` Saída: `{"second" :"3"}` Entrada: `30.5` Padrão: `%{SECOND: minSec}` Saída: `{"minSec" :"30.5"}` Entrada: `30:5` Padrão: `%{SECOND: minSec}` Saída: `{"minSec" :"30:5"}`
TIME	Corresponde a um formato de tempo com horas, minutos e segundos no formato (H)H:mm:(s)s. Os segundos incluem o segundo bissexto (0)0-60.	20	Entrada: `09:45:32` Padrão: `%{TIME:ti me}` Saída: `{"time":" 09:45:32"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
DATE_US	Corresponde a uma data no formato (M)M/(d)d /(aa)aa ou (M)M-(d)d-(aa)aa.	20	Entrada: `11/23/2024` Padrão: `%{DATE_US :date}` Saída: `{"date":" 11/23/2024"}` Entrada: `1-01-24` Padrão: `%{DATE_US :date}` Saída: `{"date":" 1-01-24"}`
DATE_EU	Corresponde à data no formato (d)d/(M)M/(aa)aa, (d)d-(M)M-(aa)aa ou (d)d. (M)M.(aa)aa.	20	Entrada: `23/11/2024` Padrão: `%{DATE_EU :date}` Saída: `{"date":" 23/11/2024"}` Entrada: `1.01.24` Padrão: `%{DATE_EU :date}` Saída: `{"date":" 1.01.24"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
ISO8601_FUSO HORÁRIO	Corresponde ao deslocamento UTC 'Z' ou ao deslocamento de fuso horário com dois pontos opcionais no formato [+-](H)H(:)mm.	20	Entrada: `+05:30` Padrão: `%{ISO8601_TIMEZONE:tz}` Saída: `{"tz":"+05:30"}` Entrada: `-530` Padrão: `%{ISO8601_TIMEZONE:tz}` Saída: `{"tz":"-530"}` Entrada: `Z` Padrão: `%{ISO8601_TIMEZONE:tz}` Saída: `{"tz":"Z"}`
ISO8601_SEGUNDO	Corresponde a um número que represent a os segundos (0)0-60, opcionalmente seguido por um ponto decimal ou dois pontos e um ou mais dígitos por segundos fracionários	20	Entrada: `60` Padrão: `%{ISO8601_SECOND:second}` Saída: `{"second":"60"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
CARIMBO DE DATA/ HORA _ ISO8601	Corresponde ao formato de ISO8601 data e hora (yy) yy- (M) M- (d) dT (H) H:mm: ((s) s) (Z| [+-] (H) H:mm) com segundos e fuso horário opcionais.	20	Entrada: `2023-05-15T14:30:00+05:30` Padrão: `%{TIMESTAMP_ISO8601:timestamp}` Saída: `{"timestamp":"2023-05-15T14:30:00+05:30"}` Entrada: `23-5-1T1:25+5:30` Padrão: `%{TIMESTAMP_ISO8601:timestamp}` Saída: `{"timestamp":"23-5-1T1:25+5:30"}` Entrada: `23-5-1T1:25Z` Padrão: `%{TIMESTAMP_ISO8601:timestamp}` Saída: `{"timestamp":"23-5-1T1:25Z"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
DATE	Corresponde a uma data no formato dos EUA usando %{DATE_US} ou no formato da UE usando %{DATE_EU}	20	Entrada: `11/29/2024` Padrão: `%{DATE:date}` Saída: `{"date":"11/29/2024"}` Entrada: `29.11.2024` Padrão: `%{DATE:date}` Saída: `{"date":"29.11.2024"}`
DATESTAMP	Corresponde a %{DATE} seguido pelo padrão %{TIME}, separado por espaço ou hífen.	20	Entrada: `29-11-2024 14:30:00` Padrão: `%{DATESTAMP:dateTime}` Saída: `{"dateTime":"29-11-2024 14:30:00"}`
TZ	Corresponde às abreviações de fuso horário comuns (PST, PDT, MST, MDT, CST CDT, EST, EDT, UTC).	20	Entrada: `PDT` Padrão: `%{TZ:tz}` Saída: `{"tz":"PDT"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
CARIMBO DE DATA_ RFC822	Corresponde à data e hora no formato: Dia MonthName (D) D (YY) YY (H) H:mm :(s) s Fuso horário	20	Entrada: `Monday Jan 5 23 1:30:00 CDT` Padrão: `%{DATESTA MP_RFC822 :dateTime}` Saída: `{"dateTim e":"Monday Jan 5 23 1:30:00 CDT"}` Entrada: `Mon January 15 2023 14:30:00 PST` Padrão: `%{DATESTA MP_RFC822 :dateTime}` Saída: `{"dateTim e":"Mon January 15 2023 14:30:00 PST"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
CARIMBO DE DATA_ RFC2822	Corresponde ao formato RFC2822 de data e hora: Dia, (d) d MonthName (yy) yy (H) H:mm :(s) s Z|[+-] (H) H:mm	20	Entrada: `Mon, 15 May 2023 14:30:00 +0530` Padrão: `%{DATESTAMP_RFC2822:dateTime}` Saída: `{"dateTime":"Mon, 15 May 2023 14:30:00 +0530"}` Entrada: `Monday, 15 Jan 23 14:30:00 Z` Padrão: `%{DATESTAMP_RFC2822:dateTime}` Saída: `{"dateTime":"Monday, 15 Jan 23 14:30:00 Z"}`
DATESTAMP_OTHER	Corresponde à data e hora no formato: Dia MonthName (d) d (H) H:mm :(s) s Fuso horário (yy) yy	20	Entrada: `Mon May 15 14:30:00 PST 2023` Padrão: `%{DATESTAMP_OTHER:dateTime}` Saída: `{"dateTime":"Mon May 15 14:30:00 PST 2023"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
DATESTAMP_EVENTLOG	Corresponde ao formato compacto de data e hora sem separadores: (aa)aaMM(d)d(H)Hmm(s)s	20	Entrada: `20230515143000` Padrão: `%{DATESTAMP_EVENTLOG:dateTime}` Saída: `{"dateTime":"20230515143000"}`

Padrões Grok de log

Padrão Grok	Description	Limite máximo de padrões	Exemplo
LOGLEVEL	Corresponde aos níveis de log padrão em diferentes letras maiúsculas e abreviações, incluindo as seguintes: `Alert/ALERT`, `Trace/TRACE`, `Debug/DEBUG`, `Notice/NOTICE`, `Info/INFO`, `Warn/Warning/WARN/WARNING`, `Err/Error/ERR/ERROR`, `Crit/Critical/CRIT/CRITICAL`, `Fatal/FATAL`, `Severe/SEVERE`, `Emerg/Emergency/EMERG/EMERGENCY`	20	Entrada: `INFO` Padrão: `%{LOGLEVEL:logLevel}` Saída: `{"logLevel":"INFO"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
HTTPDATE	Corresponde ao formato de data e hora frequente mente usado em arquivos de log. Formato: (d) MonthName d/ (yy) yy : (H) H:mm :(s) s Fuso horário: corresponde aos nomes completos ou abreviados dos meses em inglês (exemplo MonthName: “janeiro” ou “janeiro”) Fuso horário: corresponde ao padrão% {INT} grok	20	Entrada: `23/Nov/20 24:14:30:00 +0640` Padrão: `%{HTTPDAT E:date}` Saída: `{"date":"23/ Nov/2024:14:30:00 +0640"}`
SYSLOGTIM ESTAMP	Corresponde ao formato de data com MonthName (d) d (H) H:mm :(s) s MonthName: Correspon de aos nomes completos ou abreviados dos meses em inglês (exemplo: “janeiro” ou “janeiro”)	20	Entrada: `Nov 29 14:30:00` Padrão: `%{SYSLOGT IMESTAMP: dateTime}` Saída: `{"dateTim e":"Nov 29 14:30:00"}`
PROG	Corresponde a um nome de programa que consiste em uma sequência de letras, dígitos, ponto, sublinhado, barra, sinal de porcentagem e caracteres de hífen.	20	Entrada: `user.prof ile/settings-page` Padrão: `%{PROG:pr ogram}` Saída: `{"program ":"user.profile/ settings-page"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
SYSLOGPRO G	Corresponde ao padrão Grok PROG opcionalm ente seguido por um ID de processo entre colchetes.	20	Entrada: `user.prof ile/settings- page[1234]` Padrão: `%{SYSLOGP ROG:progr amWithId}` Saída: `{"program WithId":" user.prof ile/settings- page[1234]","p rogram":" user.profile/ settings-page" ,"pid":"1234"}`
SYSLOGHOS T	Corresponde a um padrão %{HOST} ou %{IP}	5	Entrada: `2001:db8: 3333:4444 :5555:666 6:7777:8888` Padrão: `%{SYSLOGH OST:ip}` Saída: `{"ip": "2001:db8:3333:444 4:5555:66 66:7777:8888"}`

Padrão Grok	Description	Limite máximo de padrões	Exemplo
SYSLOGFAC ILITY	Corresponde à prioridad e do syslog no formato decimal. O valor deve estar entre colchetes angulares (**<>**).	20	Entrada: `<13.6>` Padrão: `%{SYSLOGF ACILITY:syslog}` Saída: `{"syslog" :"<13.6>" ,"facilit y":"13"," priority":"6"}`

Padrões Grok de log comum

Você pode usar padrões Grok personalizados predefinidos para corresponder aos formatos de log Apache, NGINX e protocolo Syslog (RFC 5424). Quando você usa esses padrões específicos, eles devem ser os primeiros em sua configuração correspondente, e nenhum outro padrão pode precedê-los. Além disso, você pode segui-los somente com exatamente um DADO. Padrão GREEDYDATA ou GREEDYDATA_MULTILINE,

Padrão grok	Description	Limite máximo de padrões
APACHE ACCESS_LOG	Correspon de aos logs de acesso do Apache	1
NGINX_ACCESS_LOG	Correspon de aos logs de acesso do NGINX	1
SYSLOG5424	Correspon de aos logs do protocolo	1

Padrão grok	Description	Limite máximo de padrões
	Syslog (RFC 5424)	

Veja a seguir exemplos válidos e inválidos do uso desses padrões comuns de formato de log.

```
"%{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)
```

Exemplos comuns de formato de log

Exemplo de log do Apache

Log de amostra:

```
127.0.0.1 - - [03/Aug/2023:12:34:56 +0000] "GET /page.html HTTP/1.1" 200 1234
```

Transformador:

```
[
     {
        "grok": {
            "match": "%{APACHE_ACCESS_LOG}"
        }
    }
]
```

Saída:

```
{
    "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"
}
```

Exemplo de log do NGINX

Log de amostra:

```
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"
```

Transformador:

```
[
     {
        "grok": {
            "match": "%{NGINX_ACCESS_LOG}"
        }
    }
]
```

Saída:

```
{
    "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"
}
```

Exemplo de log do protocolo Syslog (RFC 5424)

Log de amostra:

```
<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"]
```

Transformador:

```
[
     {
        "grok": {
            "match": "%{SYSLOG5424}"
        }
    }
]
```

Saída:

```
{
  "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

O processador csv analisa valores separados por vírgulas (CSV) dos eventos de logs em colunas.

Campo	Description	Obrigatório?	Padrão	Limites
origem	Caminho para o campo no evento de logs que será analisado	Não	`@message`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3

Campo	Description	Obrigatório?	Padrão	Limites
delimitador	O caractere usado para separar cada coluna no evento de logs original com valores separados por vírgula	Não	`,`	Comprimento máximo: 1, a menos que o valor seja `\t` ou `\s`
quoteCharacter	Caractere usado como qualificador de texto para uma única coluna de dados	Não	`"`	Tamanho máximo: 1
colunas	Lista de nomes a serem usados para as colunas no evento de logs transformado.	Não	`[column_1, column_2`	Máximo de colunas CSV: 100 Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
destination	O campo pai em que colocar os pares de valores-chave transformados	Não	`Root node`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3

Definir `delimiter` como `\t` separará cada coluna em um caractere de tabulação e `\t` separará cada coluna em um único caractere de espaço.

Exemplo

Suponha que parte de um evento de logs ingerido seja semelhante a:

```
'Akua Mansa':28:'New York: USA'
```

Suponha que usemos apenas o processador csv:

```
[
```

```
     {
        "csv": {
            "delimiter": ":",
            "quoteCharacter": "'"
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "column_1": "Akua Mansa",
  "column_2": "28",
  "column_3": "New York: USA"
}
```

Exemplo 2

Suponha que um evento de logs ingerido seja semelhante a:

```
{
    "timestamp": "2024-11-23T16:03:12Z",
    "type": "user_data",
    "logMsg": "'Akua Mansa':28:'New York: USA'"
}
```

Suponha que analisemos o evento como JSON, eles analisam um campo JSON com o processador csv, especificando os nomes e o destino das colunas:

```
[
    {
        "parseJSON": {}
    },
    {
        "csv": {
            "source": "logMsg",
            "delimiter": ":",
            "quoteCharacter": "'",
            "columns":["name","age","location"],
            "destination": "msg"
        }
```

```
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "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

Use o parseKeyValueprocessador para analisar um campo especificado em pares de valores-chave. Você pode personalizar o processador para analisar as informações do campo com as seguintes opções.

Campo	Description	Obrigatório?	Padrão	Limites
origem	Caminho para o campo no evento de logs que será analisado	Não	`@message`	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
destination	O campo de destino no qual colocar os pares de valores-chave extraídos	Não		Tamanho máximo: 128
fieldDelimiter	A string delimitadora de campo que é usada entre pares de valores-chave nos eventos de logs originais	Não	&	Tamanho máximo: 128

Campo	Description	Obrigatório?	Padrão	Limites
keyValueDelimiter	A string delimitadora a ser usada entre a chave e o valor em cada par no evento de logs transformado	Não	=	Tamanho máximo: 128
nonMatchValue	Um valor a ser inserido no campo de valor no resultado, quando um par de valores-chave não é dividido com sucesso.	Não		Tamanho máximo: 128
keyPrefix	Se você quiser adicionar um prefixo a todas as chaves transformadas, especifique-o aqui.	Não		Tamanho máximo: 128
overwriteIfExists	Se o valor deve ser substituído caso a chave de destino já exista	Não	`false`	

Exemplo

Observe o seguinte exemplo de evento de logs:

```
key1:value1!key2:value2!key3:value3!key4
```

Suponha que usemos a seguinte configuração de processador:

```
[
    {
        "parseKeyValue": {
            "destination": "new_key",
            "fieldDelimiter": "!",
            "keyValueDelimiter": ":",
            "nonMatchValue": "defaultValue",
            "keyPrefix": "parsed_"
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "new_key": {
    "parsed_key1": "value1",
    "parsed_key2": "value2",
    "parsed_key3": "value3",
    "parsed_key4": "defaultValue"
  }
}
```

Processadores integrados para AWS registros vendidos

Esta seção contém informações sobre os processadores integrados que você pode usar com AWS serviços que vendem registros.

Sumário

- parseWAF
- parsePostgres
- parseCloudfront
- parseRoute53
- parseVPC
- parseToOCSF

parseWAF

Use esse processador para analisar registros vendidos. Ele pega o conteúdo `httpRequest.headers` e cria chaves JSON a partir de cada nome de cabeçalho, com o AWS WAF valor correspondente. Também faz o mesmo para `labels`. Essas transformações podem facilitar muito a consulta aos AWS WAF registros. Para obter mais informações sobre o formato de AWS WAF log, consulte Exemplos de log para tráfego de ACL da web.

Esse processador aceita somente `@message` como entrada.

> **Important**
>
> Se você usa esse processador, ele deve ser o primeiro em seu transformador.

Exemplo

Observe o seguinte exemplo de evento de logs:

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

A configuração do processador é a seguinte:

```
[
    {
        "parseWAF": {}
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "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

Use esse processador para analisar Amazon RDS for PostgreSQL registros vendidos, extrair campos e convertê-los para o formato JSON. Para obter mais informações sobre o formato de log do RDS para PostgreSQL, consulte RDS para arquivos de log do banco de dados PostgreSQL.

Esse processador aceita somente `@message` como entrada.

> **Important**
>
> Se você usa esse processador, ele deve ser o primeiro em seu transformador.

Exemplo

Observe o seguinte exemplo de evento de logs:

```
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
```

A configuração do processador é a seguinte:

```
[
    {
        "parsePostgres": {}
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "logTime": "2019-03-10 03:54:59 UTC",
  "srcIp": "10.0.0.123(52834)",
  "userName": "postgres",
  "dbName": "logtestdb",
```

```
  "processId": "20175",
  "logLevel": "ERROR"
}
```

parseCloudfront

Use esse processador para analisar Amazon CloudFront registros vendidos, extrair campos e convertê-los em formato JSON. Os valores de campo codificados são decodificados. Valores que são inteiros e de ponto flutuante são tratados como tal. Para obter mais informações sobre o formato do Amazon CloudFront registro, consulte Configurar e usar registros padrão (registros de acesso).

Esse processador aceita somente `@message` como entrada.

> **Important**
>
> Se você usa esse processador, ele deve ser o primeiro em seu transformador.

Exemplo

Observe o seguinte exemplo de evento de logs:

```
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
    SOX4xwn4XV6Q4rgb7XiVGOHms_BGlTAC4KyHmureZmBNrjGdRLiNIQ==
 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 -  -
```

A configuração do processador é a seguinte:

```
[
    {
        "parseCloudfront": {}
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
```

```
  "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": "SOX4xwn4XV6Q4rgb7XiVGOHms_BGlTAC4KyHmureZmBNrjGdRLiNIQ==",
  "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

Use esse processador para analisar Amazon Route 53 Public Data Plane registros vendidos, extrair campos e convertê-los em formato JSON. Os valores de campo codificados são decodificados. Esse processador não oferece suporte a Amazon Route 53 Resolver registros.

Esse processador aceita somente `@message` como entrada.

> **Important**
>
> Se você usa esse processador, ele deve ser o primeiro em seu transformador.

Exemplo

Observe o seguinte exemplo de evento de logs:

```
1.0 2017-12-13T08:15:50.235Z Z123412341234 example.com AAAA NOERROR TCP IAD12 192.0.2.0
 198.51.100.0/24
```

A configuração do processador é a seguinte:

```
[
    {
        "parseRoute53": {}
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "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

Use esse processador para analisar logs fornecidos pelo Amazon VPC, extrair campos e convertê-los no formato JSON. Os valores de campo codificados são decodificados.

Esse processador aceita somente `@message` como entrada.

> **Important**
>
> Se você usa esse processador, ele deve ser o primeiro em seu transformador.

Exemplo

Observe o seguinte exemplo de evento de logs:

```
2 123456789010 eni-abc123de 192.0.2.0 192.0.2.24 20641 22 6 20 4249 1418530010
 1418530070 ACCEPT OK
```

A configuração do processador é a seguinte:

```
[
    {
        "parseVPC": {}
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "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

O processador `parseTo0CSF` converte logs em eventos do Open Cybersecurity Schema Framework (OCSF). O OCSF é um padrão aberto que fornece um esquema comum para dados de segurança, permitindo melhor interoperabilidade e análise em diferentes ferramentas e plataformas de segurança.

Esse processador é particularmente útil para fluxos de trabalho de análise de segurança em que você precisa padronizar formatos de log de vários AWS serviços em um esquema consistente para análise posterior.

Parâmetros

`eventSource`(obrigatório)

Especifica o AWS serviço ou processo que produz os eventos de log a serem convertidos. Os valores válidos são:

- `CloudTrail`- CloudTrail troncos
- `Route53Resolver` - Logs do Route 53 Resolver
- `VPCFlow` - Logs do Amazon VPC Flow
- `EKSAudit` - Logs de auditoria do Amazon EKS
- `AWSWAF`- AWS WAF troncos

`ocsfVersion`(obrigatório)

Especifica qual versão do esquema OCSF usar para os eventos de logs transformados. Versões atualmente suportadas: `V1.1, V1.5`

`mappingVersion` (opcional)

Especifica a versão do mapeamento de transformação do OCSF. Controla qual lógica de transformação é aplicada ao converter registros para o formato OCSF. Se não for especificado, usa a versão mais recente disponível no momento da criação da política. As políticas existentes não são atualizadas automaticamente quando novas versões de mapeamento são lançadas. Versão mais recente atual:`v1.5.0`.

Nota: Não é suportado quando `ocsfVersion` é`V1.1`.

`source` (opcional)

O caminho para o campo no evento de logs que você deseja analisar. Se for omitida, toda a mensagem do log será analisada.

Exemplo

O seguinte exemplo mostra como usar `parseToOCSF` para converter logs do VPC Flow para o formato OCSF:

```
{
  "parseToOCSF": {
    "eventSource": "VPCFlow",
    "ocsfVersion": "V1.1"
  }
}
```

O exemplo a seguir mostra como especificar uma versão de mapeamento específica para um comportamento de transformação consistente:

```
{
  "parseToOCSF": {
    "eventSource": "CloudTrail",
    "ocsfVersion": "V1.5",
    "mappingVersion": "v1.5.0"
  }
}
```

Processadores de mutação de string

Esta seção contém informações sobre os processadores de mutação de string que você pode usar com um transformador de eventos de log.

Sumário

- lowerCaseString
- upperCaseString
- splitString
- substituteString

- trimString

lowerCaseString

O processador `lowerCaseString` converte uma string em sua versão em minúsculas.

Campo	Description	Obrigatório?	Padrão	Limites
withKeys	Uma lista de chaves a serem convertidas em letras minúsculas	Sim		Máximo de entradas: 10

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "INNER_VALUE"
    }
}
```

A configuração do transformador é esta, usando `lowerCaseString` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "lowerCaseString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
```

```
    "inner_key": "inner_value"
  }
}
```

upperCaseString

O processador `upperCaseString` converte uma string em sua versão em maiúsculas.

Campo	Description	Obrigatório?	Padrão	Limites
withKeys	Uma lista de chaves a serem convertidas em maiúsculas	Sim		Máximo de entradas: 10

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `upperCaseString` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "upperCaseString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
```

```
  "outer_key": {
    "inner_key": "INNER_VALUE"
  }
}
```

splitString

O processador `splitString` é um tipo de processador de mutação de string que divide um campo em uma matriz usando um caractere delimitador.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz deve conter os campos `source` e `delimiter` .	Sim		Máximo de entradas: 10
origem	A chave do valor do campo a ser dividido	Sim		Tamanho máximo: 128
delimitador	A string delimitadora onde dividir o valor do campo	Sim		Tamanho máximo: 128

Exemplo 1

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `splitString` com `parseJSON`:

```
[
     {
        "parseJSON": {}
    },
    {
```

```
         "splitString": {
            "entries": [
                {
                    "source": "outer_key.inner_key",
                    "delimiter": "_"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key": [
      "inner",
      "value"
    ]
  }
}
```

Exemplo 2

O delimitador para dividir a string pode ter vários caracteres.

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "item1, item2, item3"
    }
}
```

A configuração do transformador é a seguinte:

```
[
     {
        "parseJSON": {}
    },
    {
         "splitString": {
            "entries": [
```

```
                {
                    "source": "outer_key.inner_key",
                    "delimiter": ", "
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key": [
      "item1",
      "item2",
      "item3"
    ]
  }
}
```

substituteString

O processador `substituteString` é um tipo de processador de mutação de string que combina o valor de uma chave com uma expressão regular e substitui todas as correspondências por uma string de substituição.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz deve conter os campos `source`, `from` e `to`.	Sim		Máximo de entradas: 10
origem	A chave do campo a ser modificado	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3

Campo	Description	Obrigatório?	Padrão	Limites
from	A string de expressão regular a ser substituída. Caracteres regex especiais, como [e], devem ser escapados usando \\ ao usar aspas duplas e com \ ao usar aspas simples ou quando configurados a partir do Console de gerenciamento da AWS. Para obter mais informações, consulte o Padrão de classe no site da Oracle. Você pode agrupar um padrão em `(...)` para criar um grupo de captura numerado e criar grupos de captura denominados `(?P<group_name>...)` que possam ser referenciados no campo `to`.	Sim		Tamanho máximo: 128
para	A string a ser substituída para cada correspondência de `from`. Podem ser usadas referências anteriores para os grupos de captura. Use o formulário $n para grupos numerados como `$1`, e use `${group_name}` para grupos nomeados, como `${my_group} .>`	Sim		Tamanho máximo: 128 Número máximo de referências anteriores: 10 Número máximo de referências anteriores duplicadas: 2

Exemplo 1

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key1": "[]",
        "inner_key2": "123-345-567",
        "inner_key3": "A cat takes a catnap."
    }
}
```

A configuração do transformador é esta, usando `substituteString` com `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"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key1": "value1",
```

```
    "inner_key2": "xxx-xxx-xxx",
    "inner_key3": "A dog takes a dognap."
  }
}
```

Exemplo 2

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key1": "Tom, Dick, and Harry",
        "inner_key2": "arn:aws:sts::123456789012:assumed-role/MyImportantRole/
MySession"
    }
}
```

A configuração do transformador é esta, usando `substituteString` com `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}"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key1": "Tom and Harry",
    "inner_key2": "123456789012:MyImportantRole:MySession"
  }
}
```

trimString

O processador `trimString` remove o espaço em branco do início e do fim de uma chave.

Campo	Description	Obrigatório?	Padrão	Limites
withKeys	Uma lista de chaves a serem tratadas	Sim		Máximo de entradas: 10

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "   inner_value  "
    }
}
```

A configuração do transformador é esta, usando `trimString` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "trimString": {
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key": "inner_value"
  }
}
```

Processadores de mutação do JSON

Esta seção contém informações sobre os processadores de mutação JSON que você pode usar com um transformador de eventos de log.

Sumário

- addKeys
- deleteKeys
- moveKeys
- renameKeys
- copyValue
- listToMap

addKeys

Use o processador `addKeys` para adicionar novos pares de chave-valor ao evento de logs.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz pode conter os campos `key`, `value` e `overwriteIfExists` .	Sim		Máximo de entradas: 5
key	A chave da nova entrada a ser adicionada	Sim		Tamanho máximo: 128

Campo	Description	Obrigatório?	Padrão	Limites
				Profundidade máxima de chave aninhada: 3
valor	O valor da nova entrada a ser adicionada	Sim		Tamanho máximo = 256
overwriteIfExists	Se você definir isso como `true`, o valor existente será substituído se `key` já existir no evento. O valor padrão é `false`.	Não	false	Sem limite

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `addKeys` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "addKeys": {
            "entries": [
                {
                    "key": "outer_key.new_key",
                    "value": "new_value"
                }
            ]
        }
    }
```

```
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {
    "inner_key": "inner_value",
    "new_key": "new_value"
  }
}
```

deleteKeys

Use o processador `deleteKeys` para excluir campos de um evento de logs. Esses campos podem incluir pares de chave-valor.

Campo	Description	Obrigatório?	Padrão	Limites
withKeys	Uma lista de chaves a serem excluídas.	Sim	Sem limite	Máximo de entradas: 5

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `deleteKeys` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "deleteKeys": {
```

```
            "withKeys":["outer_key.inner_key"]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key": {}
}
```

moveKeys

Use o processador `moveKeys` para mover uma chave de um campo para outro.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz pode conter os campos `source`, `target` e `overwriteIfExists` .	Sim		Máximo de entradas: 5
origem	A chave a ser movida	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
target	A chave a ser movida para	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
overwriteIfExists	Se você definir isso como `true`, o valor existente será substituído se `key` já existir no evento. O valor padrão é `false`.	Não	false	Sem limite

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key1": {
        "inner_key1": "inner_value1"
    },
    "outer_key2": {
        "inner_key2": "inner_value2"
    }
}
```

A configuração do transformador é esta, usando `moveKeys` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "moveKeys": {
            "entries": [
                {
                    "source": "outer_key1.inner_key1",
                    "target": "outer_key2"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "outer_key1": {},
  "outer_key2": {
    "inner_key2": "inner_value2",
    "inner_key1": "inner_value1"
  }
}
```

renameKeys

Use o processador `renameKeys` para renomear as chaves em um evento de logs.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz pode conter os campos `key`, `target` e `overwriteIfExists` .	Sim	Sem limite	Máximo de entradas: 5
key	A chave a ser renomeada	Sim	Sem limite	Tamanho máximo: 128
target	O novo nome da chave	Sim	Sem limite	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
overwriteIfExists	Se você definir isso como `true`, o valor existente será substituído se `key` já existir no evento. O valor padrão é `false`.	Não	false	Sem limite

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `renameKeys` com `parseJSON`:

```
[
```

```
    {
        "parseJSON": {}
    },
    {
        "renameKeys": {
            "entries": [
                {
                    "key": "outer_key",
                    "target": "new_key"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "new_key": {
    "inner_key": "inner_value"
  }
}
```

copyValue

Use o processador `copyValue` para copiar valores em um evento de logs. Você também pode usar esse processador para adicionar metadados aos eventos de logs, copiando os valores das seguintes chaves de metadados nos eventos de logs:`@logGroupName`, `@logGroupStream`, `@accountId`, `@regionName`. Isso é ilustrado no exemplo a seguir.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz pode conter os campos `source`, `target` e `overwriteIfExists` .	Sim		Máximo de entradas: 5
origem	A chave a ser copiada	Sim		Tamanho máximo: 128

Campo	Description	Obrigatório?	Padrão	Limites
				Profundidade máxima de chave aninhada: 3
target	A chave cujo valor será copiado para	Sim	Sem limite	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
overwriteIfExists	Se você definir isso como `true`, o valor existente será substituído se `key` já existir no evento. O valor padrão é `false`.	Não	false	Sem limite

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "outer_key": {
        "inner_key": "inner_value"
    }
}
```

A configuração do transformador é esta, usando `copyValue` com `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"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
  "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

O processador `listToMap` pega uma lista de objetos que contêm campos-chave e os converte em um mapa das chaves de destino.

Campo	Description	Obrigatório?	Padrão	Limites
origem	A chave no ProcessingEvent com uma lista de objetos que serão convertidos em um mapa	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
key	A chave dos campos a serem extraídos como chaves no mapa gerado	Sim		Tamanho máximo: 128
valueKey	Se isso for especificado, os valores especificados nesse parâmetro serão extraídos dos objetos `source` e colocados nos valores do mapa gerado. Caso contrário, os objetos originais na lista da origem serão colocados nos valores do mapa gerado.	Não		Tamanho máximo: 128
target	A chave do campo que conterá o mapa gerado	Não	Nó raiz	Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
nivelamento	Um valor booleano para indicar se a lista será nivelada em itens únicos ou se os valores no mapa gerado serão listas. Por padrão, os valores das chaves correspondentes serão representados em uma matriz. Defina `flatten` como `true` para converter a matriz em um	Não	false	

Campo	Description	Obrigatório?	Padrão	Limites
	único valor com base no valor de `flattenedElement` .			
flattenedElement	Se você definir `flatten` como `true`, use `flattenedElement` para especificar qual elemento manter, `first` ou `last`.	Obrigatório quando `flatten` estiver definido como `true`		O valor pode ser somente `first` ou `last`

Exemplo

Observe o seguinte exemplo de evento de logs:

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

Transformador para caso de uso 1: `flatten` é `false`

```
[
    {
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
            "valueKey": "inner_value",
            "flatten": false
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "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"
    ]
}
```

Transformador para o caso de uso 2: `flatten` é `true` e `flattenedElement` é `first`

```
[
    {
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
            "valueKey": "inner_value",
            "flatten": true,
            "flattenedElement": "first"
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "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"
}
```

Transformador para o caso de uso 3: `flatten` é `true` e `flattenedElement` é `last`

```
[
    {
        "parseJSON": {}
    },
    {
        "listToMap": {
            "source": "outer_key"
            "key": "inner_key",
            "valueKey": "inner_value",
            "flatten": true,
            "flattenedElement": "last"
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "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"
}
```

Processadores de conversão de tipos de dados

Esta seção contém informações sobre os processadores conversores de tipos de dados que você pode usar com um transformador de eventos de log.

Sumário

- typeConverter
- datetimeConverter

typeConverter

Use o processador `typeConverter` para converter um tipo de valor associado à chave especificada para o tipo especificado. É um processador de conversão que altera os tipos dos campos especificados. Os valores podem ser convertidos em um dos seguintes tipos de dados: `integer`, `double`, `string` e `boolean`.

Campo	Description	Obrigatório?	Padrão	Limites
entries	Matriz de entradas. Cada item na matriz deve conter os campos `key` e `type`.	Sim		Máximo de entradas: 10
key	A chave com o valor que deve ser convertido em um tipo diferente	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3

Campo	Description	Obrigatório?	Padrão	Limites
type	O tipo para o qual converter. Os valores válidos são `integer`, `double`, `string` e `boolean`.	Sim		

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{
    "name": "value",
    "status": "200"
}
```

A configuração do transformador é esta, usando `typeConverter` com `parseJSON`:

```
[
    {
        "parseJSON": {}
    },
    {
        "typeConverter": {
            "entries": [
                {
                    "key": "status",
                    "type": "integer"
                }
            ]
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "name": "value",
    "status": 200
}
```

datetimeConverter

Use o processador `datetimeConverter` para converter uma string de data e hora em um formato que você especificar.

Campo	Description	Obrigatório?	Padrão	Limites
origem	A chave na qual aplicar a conversão de data.	Sim		Máximo de entradas: 10
matchPatterns	Uma lista de padrões para combinar com o campo `source`	Sim		Máximo de entradas: 5
target	O campo JSON no qual armazenar o resultado.	Sim		Tamanho máximo: 128 Profundidade máxima de chave aninhada: 3
targetFormat	O formato de data e hora a ser usado para os dados convertidos no campo de destino.	Não	`yyyy-MM-dd'T'HH:mm:ss.SSS'Z`	Tamanho máximo: 64
sourceTimezone	O fuso horário do campo de origem. Para obter uma lista de valores possíveis, consulte Ids e deslocamentos de zona aceitos pelo Java.	Não	UTC	Tamanho mínimo: 1
targetTimezone	O fuso horário do campo de destino. Para obter uma lista de valores possíveis, consulte Ids e	Não	UTC	Tamanho mínimo: 1

Campo	Description	Obrigatório?	Padrão	Limites
	deslocamentos de zona aceitos pelo Java.			
locale	O locale do campo de origem. Para obter uma lista de valores possíveis, consulte o Método Locale getAvailableLocales () em Java com exemplos.	Sim		Tamanho mínimo: 1

Exemplo

Observe o seguinte exemplo de evento de logs:

```
{"german_datetime": "Samstag 05. Dezember 1998 11:00:00"}
```

A configuração do transformador é esta, usando `dateTimeConverter` com `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"
        }
    }
]
```

O evento de logs transformado seria o seguinte.

```
{
    "german_datetime": "Samstag 05. Dezember 1998 11:00:00",
    "target_1": "1998-12-05T17:00:00 MEZ"
}
```

Métricas e erros de transformação

CloudWatch O Logs publica métricas de transformação em CloudWatch. Essas métricas são `TransformedLogEvents`, `TransformedBytes` e `TransformationErrors`. Para obter mais informações, consulte Métricas e dimensões do transformador de log.

Sempre que o CloudWatch Logs tenta e não consegue transformar um evento de registro, ele adiciona um campo `@transformationError` do sistema a esse evento de registro. Ao executar uma consulta do CloudWatch Logs Insights, você verá esse campo em todos os eventos de log que tiveram uma falha na transformação. Você pode consultar esse campo com uma consulta como `filter ispresent(@transformationError)` para encontrar todos os eventos de transformação com falha.

Registros de acesso com integração de tabelas do S3

A integração de tabelas do S3 CloudWatch permite que você acesse dados de log ingeridos CloudWatch usando mecanismos de análise como Amazon Athena, Amazon Redshift e ferramentas de terceiros que oferecem suporte à conexão com lojas Apache. Iceberg-compatible Essa integração permite que você realize uma análise abrangente de registros usando ferramentas de sua preferência e correlacione dados em CloudWatch registros com dados que não CloudWatch são dados. Esse recurso está disponível sem custo adicional.

Entendendo a integração de tabelas do S3

A integração de tabelas do Amazon S3 é uma solução totalmente gerenciada que disponibiliza seus registros em CloudWatch registros como tabelas gerenciadas do Amazon S3. Com essa integração, você ganha maior flexibilidade na forma como analisa seus registros, além dos recursos de CloudWatch registros.

A integração funciona criando um bucket de tabelas gerenciado do Amazon S3 (`aws-cloudwatch`) e associando fontes de log específicas às tabelas do Amazon S3 com base no nome e no tipo da fonte de dados (que podem ser gerenciados na guia Gerenciamento de registros > Fontes de dados no console de registros). CloudWatch Uma vez associados, CloudWatch os dados de registros se tornam acessíveis por meio de tabelas do Amazon S3 usando o formato Apache Iceberg. Esse formato fornece uma maneira padronizada para vários mecanismos de análise consultarem os dados com eficiência.

Componentes principais

Tabelas do Apache Iceberg

: O formato de tabela subjacente usado pelo S3 Tables que fornece armazenamento estruturado de dados e permite compatibilidade com vários mecanismos de análise.

Associação de fontes de dados

: O processo de vincular fontes de CloudWatch registros específicas à integração de tabelas do S3 com base na fonte de dados e nos critérios de tipo.

Fluxo de dados para tabelas do S3

Entender como os dados fluem entre CloudWatch os registros e as tabelas do S3 ajuda você a planejar sua integração e gerenciar seus dados de registro de forma eficaz.

Quando você cria uma associação, o CloudWatch Logs entrega automaticamente novos eventos de log correspondentes ao nome e tipo da fonte de dados associada a um bucket de tabela do CloudWatch-managed S3. Esses eventos aparecem no namespace de registros abaixo da tabela correspondente para essa fonte de dados. Os processos de integração registram somente os eventos recebidos após a criação da associação. Os dados de registro existentes não são preenchidos. Ao criar uma integração do S3 Tables, se você deixar a caixa de seleção Habilitar todas as fontes e tipos de log disponíveis na tabela do S3 marcada, todas as fontes de dados em sua conta serão automaticamente associadas e entregues às tabelas do S3 por padrão, incluindo quaisquer fontes de dados adicionadas no futuro. Para fornecer somente fontes de dados específicas às tabelas do S3, desmarque essa caixa de seleção durante a criação da integração e associe individualmente as fontes de dados que você deseja incluir.

A retenção de dados no bucket de tabelas do S3 corresponde à política de retenção definida para o grupo de registros. Por exemplo, se você definir um grupo de registros para retenção de 1 dia, o CloudWatch Logs removerá os dados dos CloudWatch Logs e da Tabela S3 após um dia. Quando você exclui um grupo de registros ou um stream de CloudWatch registros, o Logs também remove os dados do bucket da tabela do S3.

Quando usar a integração de tabelas do S3

Considere usar a integração de tabelas do S3 para correlacionar dados de log com outros dados externos ou não, ou quando preferir usar outras ferramentas de análise, como o Amazon Athena, para realizar análises CloudWatch em CloudWatch dados de registros. Use essa integração quando precisar de recursos que vão além do que está disponível no CloudWatch Logs. Essa integração é particularmente valiosa quando:

- Você precisa executar SQL-like consultas complexas em grandes volumes de dados de log
- Você deseja integrar a análise de registros aos fluxos de trabalho e ferramentas de análise existentes
- Você precisa de recursos abrangentes de análise de registros que abranjam várias fontes de dados

Não há cobranças adicionais de armazenamento ou manutenção de tabelas para tabelas do S3 criadas por meio dessa integração, além dos preços existentes CloudWatch de ingestão e armazenamento.

Pré-requisitos

Antes de implementar a integração, verifique se você tem o seguinte:

- Dados CloudWatch de registros existentes
- Permissões apropriadas do IAM para acesso entre serviços entre CloudWatch registros e tabelas do S3, conforme descrito na seção a seguir

Permissões do IAM

Para integrar CloudWatch os registros às tabelas do S3, você precisa configurar as permissões do IAM para duas entidades separadas: o usuário ou a função que configura a integração e a função de serviço que o CloudWatch Logs assume para gravar dados nas tabelas do S3.

Para a função ou usuário que está criando a integração

O usuário ou a função que configura a integração exige as seguintes permissões:

- `observabilityadmin:CreateS3TableIntegration`para criar a integração e `logs:AssociateSourceToS3TableIntegration` adicionar fontes
- `s3tables:CreateTableBucket`,`s3tables:PutTableBucketEncryption`, e `s3tables:PutTableBucketPolicy` para configurar o bucket de tabela S3

Para o perfil de serviço

Anexe a seguinte política do IAM à função de serviço do IAM que o CloudWatch Logs usa para gravar dados no bucket da tabela. Essa política concede permissão para gravar nas tabelas. Substitua *`aws-region`*`123456789012`, e *`log-group-name`* por sua AWS região, ID da conta e nome do grupo de registros.

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

Anexe a seguinte política de confiança à função de serviço do IAM que a CloudWatch Logs assumirá para gravar dados de log nas tabelas do S3. Você cria ou seleciona essa função durante a configuração da integração. As condições restringem a função para que o CloudWatch Logs só possa assumi-la para a conta e o grupo de registros especificados. Substitua *aws-region123456789012*, e *log-group-name* por sua AWS região, ID da conta e nome do grupo de registros.

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

```
}
```

Política de chaves do KMS (para dados criptografados)

Se você usar uma chave gerenciada pelo cliente para criptografar seus dados de log, deverá conceder acesso à chave ao responsável pelo CloudWatch serviço e ao responsável pelo serviço de manutenção do S3 Tables. Adicione as seguintes declarações à sua política de chaves do KMS. Substitua os valores do espaço reservado por seu Conta da AWS ID, região, ID da chave KMS e ARN da tabela ou do bucket de tabela do 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>/*"
                }
            }
        }
    ]
}
```

Conceitos básicos

Para começar com a integração de tabelas do S3, você precisa configurar a integração entre seus CloudWatch registros e tabelas do S3. Esse processo envolve a configuração de associações de fontes de dados e a configuração de permissões apropriadas do IAM.

Para criar uma integração de tabelas do S3

1. Abra o console de CloudWatch registros em https://console.aws.amazon.com/cloudwatch/".
2. Escolha Configurações, Global, Criar integração de tabelas do S3.
3. Personalize como os registros serão criptografados nas tabelas do S3 e a função que CloudWatch os registros usarão para gravar seus registros nas tabelas do S3.
4. Se você quiser que todas as fontes de dados sejam associadas automaticamente à integração, deixe marcada a caixa de seleção Habilitar que todas as fontes e tipos de log estejam disponíveis na tabela do S3 (ela é selecionada por padrão). Se você quiser associar somente fontes de dados específicas, desmarque essa caixa de seleção.
5. Escolha Criar integração de tabela do S3.

> **Note**
>
> Se você selecionou Habilitar que todas as fontes e tipos de log estejam disponíveis na tabela do S3 durante a criação, todas as fontes de dados serão associadas automaticamente, incluindo todas as fontes de dados adicionadas no futuro. Você pode parar aqui. As etapas a seguir só serão necessárias se você tiver desmarcado a caixa de seleção e quiser associar fontes de dados específicas.

Para associar fontes a uma integração de tabelas do S3

1. Abra o console de CloudWatch registros em https://console.aws.amazon.com/cloudwatch/".
2. Escolha Configurações, Global, Gerenciar integração de tabelas do S3.
3. Escolha Associar fonte de dados.
4. Selecione o nome e o tipo da fonte de dados para os quais você deseja habilitar a integração.
5. Escolha Associar fonte de dados.

Para associar fontes a uma integração de tabela do S3 a partir da página de gerenciamento de registros

1. Abra o console de CloudWatch registros em https://console.aws.amazon.com/cloudwatch/".
2. Escolha Gerenciamento de registros no painel de navegação.
3. Selecione a guia Fontes de dados.
4. Escolha o nome e o tipo da fonte de dados que você deseja integrar.
5. Escolha Ações da fonte de dados.
6. Selecione Associar à integração de tabelas do S3.
7. Revise as fontes de dados e escolha Associar fonte de dados.

Antes de usar os dados, você precisa seguir as 3 etapas a seguir:

1. Integrando tabelas do Amazon Amazon S3 AWS com serviços de análise - Usando o console do Amazon S3
2. Configurar as permissões do Lake Formation
3. Conecte-se com as ferramentas de análise

Integrando tabelas do Amazon Amazon S3 com AWS serviços de análise - Usando o console Amazon S3 (Link)

Para habilitar a integração de tabelas do S3 usando o console do S3

1. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. No painel de navegação à esquerda, escolha Buckets de tabela.
3. Escolha a opção Ativar integração na parte superior.

4. Na primeira vez que você integra buckets de tabela em qualquer região, o Amazon Amazon S3 cria uma nova função de serviço do IAM em seu nome. Essa função permite que o Lake Formation acesse todos os compartimentos de mesa em sua conta e federe o acesso às suas tabelas no AWS Glue Data Catalog.

Configurar as permissões do Lake Formation

Embora o CloudWatch Logs tenha permissão para gravar na tabela (configurada nas etapas anteriores), os usuários e as funções de análise não têm permissão automática para ler os dados. Você deve conceder acesso explicitamente usando o AWS Lake Formation. Você precisa fazer isso para cada diretor do IAM que você deseja fornecer acesso à tabela.

Para conceder acesso de consulta a usuários ou funções

Você deve conceder as permissões SELECT e DESCRIBE aos diretores do IAM (usuários ou funções) que executarão consultas no Athena ou no Redshift.

1. Abra o console do AWS Lake Formation.
2. No painel de navegação, em Permissões, escolha Permissões do data lake.
3. Selecione Conceder.
4. Diretores: selecione os usuários ou funções do IAM que exigem acesso (por exemplo, seus analistas de dados ou a função de administrador que você está usando atualmente).
5. LF-Tags ou Recursos do catálogo: selecione Recursos do catálogo de dados nomeado.
6. Bancos de dados e tabelas:
   - Selecione o bucket do S3 Table criado pela CloudWatch integração (`aws-cloudwatch`).
   - Selecione a tabela específica associada à sua fonte de dados (opcional).
7. Permissões da tabela: selecione Selecionar e descrever.
8. Selecione Conceder.

> **Note**
>
> Se você encontrar erros de “Acesso negado” ao consultar registros no Athena, certifique-se de que o usuário que está executando a consulta tenha as permissões apropriadas do IAM para o Athena e as permissões do Lake Formation definidas acima.

Saiba mais sobre as permissões do Lake Formation em https://docs.aws.amazon.com/lake-formation/latest/dg/granting-catalog-permissions.html.

Conecte-se com as ferramentas de análise

Depois que as permissões forem concedidas, você poderá configurar seu serviço de análise preferido para consultar as tabelas do S3. As tabelas S3 usam o formato Apache Iceberg, que é suportado nativamente pelo Amazon Athena, Amazon Redshift e Amazon EMR.

Para consultar dados de log no Amazon Athena

O Amazon Athena interage com as tabelas do S3 por meio do catálogo de tabelas do Amazon S3.

Para configurar o Athena para consultar seus dados de registro

1. Abra o console do Amazon Athena em. https://console.aws.amazon.com/athena/
2. No editor de consultas, selecione o catálogo de Tabelas do Amazon S3 no menu suspenso da fonte de dados.
3. Se você não vê o catálogo, certifique-se de ter concluído as etapas de permissão do Lake Formation acima para sua função de usuário específica.
4. Depois que o catálogo for selecionado, suas tabelas de log aparecerão na lista do banco de dados. Agora você pode executar consultas SQL padrão em seus dados de log.

Exemplo de consulta: `SELECT * FROM "amazon_vpc__flow" LIMIT 100;`

Saiba mais sobre como conectar os serviços do Analytics às tabelas do S3 em https://docs.aws.amazon.com/AmazonS3/latest/userguide/s3-tables-integrating-aws.html.

Criar métricas de eventos de log usando filtros

Você pode pesquisar e filtrar os dados de registro que entram no CloudWatch Logs criando um ou mais filtros métricos. Os filtros métricos definem os termos e padrões a serem procurados nos dados de registro à medida que são enviados para CloudWatch os registros. CloudWatch O Logs usa esses filtros métricos para transformar dados de registro em CloudWatch métricas numéricas que você pode representar graficamente ou ativar um alarme.

Ao criar uma métrica de um filtro de log, você também pode atribuir dimensões e uma unidade à métrica. Se você especificar uma unidade, certifique-se de especificar a correta ao criar o filtro. Alterar a unidade do filtro posteriormente não terá efeito.

Se você configurou AWS Organizations e está trabalhando com contas de membros, pode usar a centralização de registros para coletar dados de registro das contas de origem em uma conta de monitoramento central.

Ao trabalhar com grupos de logs centralizados, você pode usar essas dimensões de campos do sistema ao criar filtros métricos.

- `@aws.account`- Essa dimensão representa a ID da AWS conta da qual o evento de registro se originou.
- `@aws.region`- Essa dimensão representa a AWS região em que o evento de log foi gerado.

Essas dimensões ajudam a identificar a origem dos dados de log, permitindo uma filtragem e análise mais granulares das métricas derivadas de logs centralizados. Para obter mais informações, consulte Cross-account Centralização de registros entre regiões.

O padrão de filtro será aplicado às versões transformadas dos eventos de logs caso um grupo de logs com uma assinatura utilizar a transformação de log. Para obter mais informações, consulte Transformar logs durante a ingestão.

> **Note**
>
> Os filtros de métrica só são compatíveis com grupos de logs da classe de logs Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

Você pode usar qualquer tipo de CloudWatch estatística, incluindo estatísticas percentuais, ao visualizar essas métricas ou definir alarmes.

> **Note**
>
> As estatísticas de percentual serão compatíveis com uma métrica somente se nenhum dos valores da métrica for negativo. Se você configurar o filtro de métrica para que ele possa relatar números negativos, as estatísticas de percentual não estarão disponíveis para essa métrica quando ela tiver números negativos como valores. Para obter mais informações, consulte Percentuais.

Os filtros não funcionam de forma retroativa nos dados. Eles só publicam os pontos de dados de métrica para eventos que ocorrem após a criação do filtro. Ao testar um padrão de filtro, a visualização dos Resultados do filtro mostra até as primeiras 50 linhas de log correspondentes para fins de validação. Se o timestamp nos resultados filtrados for anterior ao horário de criação da métrica, nenhum log será exibido.

> **Note**
>
> Os filtros métricos garantem a entrega, pelo menos uma vez, de eventos de log para a métrica especificada, enquanto entregas duplicadas podem ocorrer ocasionalmente.

Conteúdo

- Conceitos
- Sintaxe de padrões de filtros para filtros de métricas
- Criar filtros de métrica
- Listagem de filtros de métrica
- Excluir um filtro de métrica

Conceitos

Cada filtro de métrica é composto dos seguintes elementos-chave:

valor padrão

O valor indicado para o filtro de métrica durante um período em que logs são ingeridos, mas não se encontram logs correspondentes. Ao definir esse valor como 0, você garante que os dados sejam relatados durante cada período, impedindo métricas irregulares em períodos sem dados correspondentes. Se nenhum log for ingerido durante um período de um minuto, nenhum valor será relatado.

Se você atribuir dimensões a uma métrica criada por um filtro de métrica, não será possível atribuir um valor padrão a essa métrica.

dimensões

Dimensões são os pares chave-valor que definem melhor uma métrica. Também é possível atribuir dimensões à métrica criada de um filtro de métrica. Como as dimensões fazem parte do identificador exclusivo de uma métrica, sempre que um name/value par exclusivo é extraído dos seus registros, você cria uma nova variação dessa métrica.

padrão de filtro

Uma descrição simbólica de como CloudWatch os registros devem interpretar os dados em cada evento de registro. Por exemplo, uma entrada de log pode conter time stamps, endereços IP, sequências de caracteres, etc. Você pode usar o padrão para especificar o que procurar no arquivo de log.

metric name (nome da métrica)

O nome da CloudWatch métrica na qual as informações de registro monitoradas devem ser publicadas. Por exemplo, você pode publicar em uma métrica chamada ErrorCount.

namespace de métrica

O namespace de destino da nova CloudWatch métrica.

valor da métrica

O valor numérico para publicar a métrica cada vez que um log correspondente for encontrado. Por exemplo, se você contabilizar as ocorrências de um determinado termo, como "Erro", o valor será "1" para cada ocorrência. Se você estiver contando os bytes transferidos, poderá incrementar pelo número real de bytes encontrado no evento de log.

Sintaxe de padrões de filtros para filtros de métricas

> **Note**
>
> Como os filtros de métricas diferem das consultas do CloudWatch Logs Insights
> Os filtros métricos diferem das consultas do CloudWatch Logs Insights porque um valor numérico especificado é adicionado a um filtro métrico sempre que um registro correspondente é encontrado. Para obter mais informações, consulte Configurando valores métricos para um filtro de métrica.
> Para obter informações sobre como consultar seus grupos de CloudWatch logs com a linguagem de consulta do Amazon Logs Insights, consulteCloudWatch Sintaxe de consulta de linguagem do Logs Insights.
> Exemplos genéricos de padrões de filtros
> Para obter mais informações sobre a sintaxe de padrões de filtros genéricos aplicável a filtros de métricas, bem como filtros de assinatura e filtros de eventos de log, consulte Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura e filtros de eventos de log, que inclui os seguintes exemplos:
>
> - Sintaxe de expressões regulares (regex) compatíveis
> - Como fazer a correspondência de termos em eventos de log não estruturados
> - Correspondência de termos em eventos de log JSON
> - Como fazer a correspondência de termos em eventos de log delimitados por espaços

Os filtros métricos permitem pesquisar e filtrar dados de registro que chegam aos CloudWatch registros, extrair observações métricas dos dados de registro filtrados e transformar os pontos de dados em uma métrica de CloudWatch registros. Você define os termos e padrões a serem procurados nos dados de registro quando eles são enviados para o CloudWatch Logs. Filtros de métrica são atribuídos a grupos de logs, e todos os filtros atribuídas a um grupo de logs são aplicados a seus streams de log.

Quando um filtro de métrica corresponde a um termo, ele incrementa a contagem da métrica em um valor numérico especificado. Por exemplo, você pode criar um filtro de métrica que conte a ocorrência da palavra ERRO em seus eventos de log.

É possível atribuir unidades de medida e dimensões a métricas. Por exemplo, se você criar um filtro de métrica que conte as vezes que a palavra ERROR ocorre em seus eventos de logs, poderá

especificar uma dimensão chamada `ErrorCode` para mostrar o número total de eventos de logs que contenham a palavra ERROR e filtrar os dados por códigos de erro relatados.

> **Tip**
>
> Ao atribuir uma unidade de medida a uma métrica, certifique-se de especificar a correta. Se você alterar a unidade depois, sua alteração pode não entrar em vigor. Para ver a lista completa das unidades que oferecem CloudWatch suporte, consulte MetricDatum a Amazon CloudWatch API Reference.

Tópicos

- Configurando valores métricos para um filtro de métrica
- Publicar dimensões com métricas de valores em eventos de log JSON ou delimitados por espaços
- Usando valores em eventos de log para incrementar o valor de uma métrica

Configurando valores métricos para um filtro de métrica

Ao criar um filtro de métrica, você define seu padrão de filtro e especifica o valor da métrica e o valor padrão. Você pode definir valores de métrica para números, identificadores nomeados ou identificadores numéricos. Se você não especificar um valor padrão, CloudWatch não reportará dados quando seu filtro de métrica não encontrar uma correspondência. Recomendamos que você especifique um valor padrão, mesmo que o valor seja 0. Definir um valor padrão ajuda a CloudWatch relatar dados com mais precisão e CloudWatch evita a agregação de métricas irregulares. CloudWatch agrega e relata valores métricos a cada minuto.

Quando o filtro de métrica encontra uma correspondência em seus eventos de log, ele incrementa a contagem da métrica de acordo com o valor da métrica. Se seu filtro de métrica não encontrar uma correspondência, CloudWatch informa o valor padrão da métrica. Por exemplo, seu grupo de logs publica dois registros a cada minuto, o valor de métrica é 1 e o valor padrão é 0. Se o filtro de métrica encontrar correspondências nos dois registros de log no primeiro minuto, o valor de métrica para aquele minuto será 2. Se o filtro de métrica não encontrar correspondências em nenhum dos dois registros durante o segundo minuto, o valor padrão para aquele minuto será 0. Se você atribuir dimensões a métricas geradas por filtros de métrica, não será possível especificar valores padrão para essas métricas.

Você também pode configurar um filtro de métrica para incrementar uma métrica com um valor extraído de um evento de log, em vez de um valor estático. Para obter mais informações, consulte Usando valores em eventos de log para incrementar o valor de uma métrica.

Publicar dimensões com métricas de valores em eventos de log JSON ou delimitados por espaços

Você pode usar o CloudWatch console ou a AWS CLI para criar filtros métricos que publicam dimensões com métricas geradas por JSON e eventos de log delimitados por espaço. As dimensões são pares de name/value valores e estão disponíveis somente para padrões de filtro JSON e delimitados por espaço. Você pode criar filtros de métrica JSON e delimitados por espaço com até três dimensões. Para obter mais informações sobre dimensões e informações sobre como atribuir dimensões a métricas, consulte as seguintes seções:

- Dimensões no guia do CloudWatch usuário da Amazon
- Exemplo: extrair campos de um log do Apache e atribuir dimensões no Guia do usuário do Amazon CloudWatch Logs

> **Important**
>
> As dimensões contêm valores que coletam cobranças iguais às métricas personalizadas. Para evitar cobranças inesperadas, não especifique campos de alta cardinalidade, por exemplo, `IPAddress` ou `requestID`, como dimensões.
> Se você extrair métricas de eventos de log, você será cobrado por métricas personalizadas. Para evitar que você colete altas cobranças acidentais, a Amazon pode desativar seu filtro métrico se ele gerar 1000 name/value pares diferentes para dimensões específicas durante um determinado período de tempo.
> Você pode criar alarmes de faturamento que o notificam sobre suas cobranças estimadas. Para mais informações, consulte Criar um alarme de faturamento para monitorar suas cobranças estimadas da AWS.

Publicar dimensões com métricas de eventos de log JSON

Os exemplos a seguir contêm trechos de código que descrevem como especificar dimensões em um filtro de métrica 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**
>
> Se você testar o filtro de métrica de exemplo com o exemplo de evento de log JSON, deverá inserir o exemplo de log JSON em uma única linha.

Example: Metric filter

O filtro de métrica incrementa a métrica sempre que um evento de log JSON contiver as propriedades `eventType` e `"sourceIPAddress"`.

```
{ $.eventType = "*" && $.sourceIPAddress != 123.123.* }
```

Ao criar um filtro de métrica JSON, é possível especificar qualquer uma das propriedades no filtro de métrica como dimensão. Por exemplo, para definir `eventType` como uma dimensão, use o seguinte:

```
"eventType" : $.eventType
```

A métrica de exemplo contém uma dimensão nomeada `"eventType"`, e o valor da dimensão no evento de log de exemplo é `"UpdateTrail"`.

Publicar dimensões com métricas de eventos de log delimitados por espaços

Os exemplos a seguir contêm trechos de código que descrevem como especificar dimensões em um filtro de métrica delimitado por espaço.

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]
```

O filtro de métrica incrementa a métrica quando um evento de log delimitado por espaço inclui qualquer um dos campos especificados no filtro. Por exemplo, o filtro de métrica encontra os seguintes campos e valores no exemplo de evento de log delimitado por espaço.

```
{
   "$bytes": "1534",
   "$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"
}
```

Ao criar um filtro de métrica delimitado por espaço, é possível especificar qualquer um dos campos no filtro de métrica como uma dimensão. Por exemplo, para definir `server` como uma dimensão, use o seguinte:

```
"server" : $server
```

O filtro de métrica de exemplo contém uma dimensão nomeada `server`, e o valor da dimensão no evento de log de exemplo é `"Prod"`.

Example: Match terms with AND (&&) and OR (||)

Você pode usar os operadores lógicos AND ("&&") e OR ("||") para criar filtros de métrica delimitados por espaço que contenham condições. O filtro de métrica a seguir retorna eventos de log em que a primeira palavra nos eventos é ERROR ou qualquer string que contenha WARN.

```
[w1=ERROR || w1=%WARN%, w2]
```

Usando valores em eventos de log para incrementar o valor de uma métrica

Você pode criar filtros de métrica que publicam valores numéricos encontrados em eventos de log. O procedimento nesta seção usa o seguinte exemplo de filtro de métrica para mostrar como você pode publicar um valor numérico em um evento de log JSON em uma métrica.

```
{ $.latency = * } metricValue: $.latency
```

Criar um filtro de métrica que publique um valor em um evento de log

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs e, em seguida, escolha Grupos de log.
3. Selecione ou crie um grupo de logs.

   Para obter informações sobre como criar um grupo de logs, consulte Criar um grupo de CloudWatch logs em Logs no Guia do usuário do Amazon CloudWatch Logs.
4. Escolha Actions (Ações) e Create metric filter (Criar filtro de métrica).
5. Em Padrão de filtro, digite **`{ $.latency = * }`** e escolha Próximo.

6. Em Nome da métrica, insira myMetric.
7. Em Valor da métrica, insira **`$.latency`**.
8. (Opcional) Em Valor padrão, insira 0 e escolha Próximo.

   Recomendamos que você especifique um valor padrão, mesmo que o valor seja 0. Definir um valor padrão ajuda a CloudWatch relatar dados com mais precisão e CloudWatch evita a agregação de métricas irregulares. CloudWatch agrega e relata valores métricos a cada minuto.
9. Escolha Criar filtro de métrica.

O filtro de métrica de exemplo corresponde ao termo `"latency"` no exemplo de evento de log JSON e publica um valor numérico de 50 na métrica MyMetric.

```
{
"latency": 50,
"requestType": "GET"
}
```

Criar filtros de métrica

Os procedimentos e exemplos a seguir mostram como criar filtros de métrica.

Exemplos

- Criar um filtro de métrica para um grupo de logs
- Exemplo: contar eventos de log
- Exemplo: contar as ocorrências de um termo
- Exemplo: contar códigos HTTP 404
- Exemplo: contar códigos HTTP 4xx
- Exemplo: Extrair campos de um log Apache e atribuir dimensões

Criar um filtro de métrica para um grupo de logs

Para criar um filtro de métrica para um grupo de logs, siga estas etapas. A métrica não será visível até que haja alguns pontos de dados para ela.

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs e, em seguida, escolha Grupos de logs.
3. Escolha o nome do grupo de logs.
4. Escolha `Actions` e Create metric filter (Criar filtro de métrica).
5. Em Padrão de filtro, insira o padrão de filtro. Para obter mais informações, consulte Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura, filtros de eventos de log e Live Tail..
6. (Opcional) Se estiver usando grupos de logs centralizados, em Critérios de seleção de filtros, você poderá especificar filtros com base na conta de origem (`@aws.account`), na região de origem (`@aws.region`) ou em ambas as condições.
7. (Opcional) Para testar seu padrão de filtro, em Test Pattern (Testar padrão), insira um ou mais eventos de logs para testar o padrão. Cada evento de logs deve ser formatado em uma linha. Quebras de linha são usadas para separar eventos de logs na caixa Log event messages (Mensagens de log de eventos).
8. Escolha Next (Próximo) e digite um nome para o filtro de métrica.
9. Em Detalhes da métrica, em Namespace métrico, insira um nome para o CloudWatch namespace em que a métrica será publicada. Se esse namespace ainda não existir, certifique-se de que a opção Create new (Criar novo) esteja selecionada.
10. Em Metric name (Nome da métrica), insira um nome para a nova métrica.
11. Em Metric value (Valor da métrica), se o filtro de métrica estiver contando ocorrências das palavras-chave no filtro, digite 1. Isso incrementa a métrica em 1 para cada evento de log que contém uma das palavras-chave.

    Se preferir, insira um token, como **`$size`**. Isso incrementa a métrica pelo valor do número no campo `size` para cada evento de log que contém um campo `size`.
12. (Opcional) Em Unit (Unidade), selecione uma unidade para atribuir à métrica. Se você não especificar uma unidade, a unidade será definida como `None`.
13. (Opcional) Insira os nomes e tokens de até três dimensões para a métrica. Se você atribuir dimensões a métricas criadas por filtros de métrica, não poderá atribuir valores padrões para essas métricas.

> **Note**
>
> As dimensões são compatíveis apenas com filtros de métrica JSON ou delimitados por espaço.

14. Escolha Criar filtro de métrica. Você pode encontrar o filtro de métrica que criou no painel de navegação. Escolha Logs e depois escolha Log groups (Grupo de logs). Escolha o nome do grupo de logs para o qual você criou o filtro de métrica e, em seguida, selecione a guia Metric filters (Filtros de métrica).

Exemplo: contar eventos de log

O tipo mais simples de monitoramento de eventos de log é contar o número de eventos de log ocorridos. É possível fazer isso para manter uma contagem de todos os eventos, para criar um monitor no estilo de "pulsação" ou apenas para praticar a criação de filtros de métrica.

No exemplo de CLI a seguir, um filtro de métrica chamado MyAppAccessCount é aplicado ao grupo de logs MyApp/access.log para criar a métrica EventCount no CloudWatch MyNamespace namespace. O filtro é configurado para fazer a correspondência de qualquer conteúdo de eventos de log e incrementar a métrica por "1".

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Escolha o nome de um grupo de logs.
4. Escolha a `Actions` Criar filtro de métrica.
5. Deixe Padrão de filtro e Selecionar dados de log para testar em branco.
6. Escolha Próximo e, em Nome do filtro, digite **`EventCount`**.
7. Em Detalhes da métrica, em Namespace da métrica, digite **`MyNameSpace`**.
8. Para Metric Name (Nome da métrica), digite **`MyAppEventCount`**.
9. Confirme se o Valor da métrica é 1. Isso especifica que a contagem é aumentada em 1 para cada evento de log.

10. Para Valor padrão, insira 0 e escolha Próximo. Especificar um valor padrão garante que os dados são relatados mesmo durante períodos em que não ocorrem eventos de log. Isso evita métricas irregulares nas quais os dados, às vezes, não existem.
11. Escolha Criar filtro de métrica.

Para criar um filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
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
```

Você pode testar essa nova política postando quaisquer dados de eventos. Você deve ver os pontos de dados publicados na métrica MyAppAccessEventCount.

Para publicar dados de eventos usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
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"
```

Exemplo: contar as ocorrências de um termo

Os eventos de log frequentemente incluem mensagens importantes que você deseja contar, talvez sobre o êxito ou a falha de operações. Por exemplo, poderá ocorrer um erro e ele ser registrado em um arquivo de log se ocorrer uma falha em uma determinada operação. É possível monitorar essas entradas para entender a tendência dos erros.

No exemplo abaixo, um filtro de métrica é criado para monitorar o termo Erro. A política foi criada e adicionada ao grupo de registros MyApp/message.log. CloudWatch O Logs publica um ponto

de dados ErrorCount na métrica CloudWatch personalizada no MyApp/message.log namespace com um valor de “1" para cada evento contendo erro. Se não houver um evento com a palavra Erro, será publicado um valor de 0. Ao representar graficamente esses dados no CloudWatch console, certifique-se de usar a estatística de soma.

Depois de criar um filtro métrico, você pode ver a métrica no CloudWatch console. Ao selecionar a métrica a ser exibida, selecione o namespace da métrica que corresponde ao nome do grupo de logs. Para obter mais informações, consulte Visualizar métricas disponíveis.

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Escolha o nome do grupo de logs.
4. Escolha Ações, Criar filtro de métrica.
5. Em Padrão de filtro, insira **`Error`**.

   > **Note**
   >
   > Todas as entradas em Filtrar padrão fazem distinção de maiúsculas e minúsculas.

6. (Opcional) Para testar seu padrão de filtro, em Test Pattern (Testar padrão), insira um ou mais eventos de log a serem usados para testar o padrão. Cada evento de log deve estar dentro de uma linha, porque as quebras de linha são usadas para separar eventos de log na caixa Log event messages (Mensagens do evento de log).
7. Escolha Próximo e, na página Atribuir métrica, em Nome do filtro, digite **`MyAppErrorCount`**.
8. Em Detalhes da métrica, em Namespace da métrica, digite MyNameSpace.
9. Para Metric Name (Nome da métrica), digite ErrorCount.
10. Confirme se o Valor da métrica é 1. Isso especifica que a contagem é aumentada em 1 para cada evento de log que contém "Erro".
11. Para Valor padrão, digite 0 e escolha Próximo.
12. Escolha Criar filtro de métrica.

Para criar um filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
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
```

Você pode testar essa nova política postando eventos que contenham a palavra "Erro" na mensagem.

Para publicar eventos usando o AWS CLI

Em um prompt de comando, execute o seguinte comando da . Os padrões fazem distinções de maiúsculas e minúsculas.

```
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"
```

Exemplo: contar códigos HTTP 404

Usando o CloudWatch Logs, você pode monitorar quantas vezes seus servidores Apache retornam uma resposta HTTP 404, que é o código de resposta da página não encontrada. Você pode monitorar isso para entender a frequência com que os visitantes de seu site não encontram o recurso que procuram. Suponha que seus registros de log estejam estruturados para incluir as seguintes informações para cada evento de log (visita ao site):

- Endereço IP do solicitante
- Identidade RFC 1413
- Nome de usuário
- Timestamp
- Método de solicitação com o recurso solicitado e o protocolo
- Código de resposta HTTP para a solicitação
- Bytes transferidos na solicitação

Um exemplo disso pode ter a seguinte aparência:

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 404 2326
```

Você pode especificar uma regra que tente fazer a correspondência de eventos dessa estrutura para erros HTTP 404, como mostrado no exemplo a seguir:

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Escolha a `Actions` Criar filtro de métrica.
4. Em Padrão de filtro, digite **`[IP, UserInfo, User, Timestamp, RequestInfo, StatusCode=404, Bytes]`**.
5. (Opcional) Para testar seu padrão de filtro, em Test Pattern (Testar padrão), insira um ou mais eventos de log a serem usados para testar o padrão. Cada evento de log deve estar dentro de uma linha, porque as quebras de linha são usadas para separar eventos de log na caixa Log event messages (Mensagens do evento de log).
6. Escolha Próximo e, para Nome do filtro, digite HTTP404Errors.
7. Em Detalhes da métrica, para Namespace da métrica, insira **`MyNameSpace`**.
8. Em Nome da métrica, insira **`ApacheNotFoundErrorCount`**.
9. Confirme se o Valor da métrica é 1. Isso especifica que a contagem é aumentada em 1 para cada evento 404 Error.
10. Para Valor padrão, insira 0 e escolha Próximo.
11. Escolha Criar filtro de métrica.

Para criar um filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
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=ApacheNotFoundErrorCount,metricNamespace=MyNamespace,metricValue=1
```

Neste exemplo, os caracteres literais, como os colchetes à direita e à esquerda, aspas duplas e string de caracteres 404 foram usados. O padrão precisa fazer a correspondência com toda a mensagem de evento de log para o evento de log a ser considerado para monitoramento.

Você pode verificar a criação do filtro de métrica usando o comando describe-metric-filters. Você deve ver uma saída semelhante a:

```
aws logs describe-metric-filters --log-group-name MyApp/access.log

{
    "metricFilters": [
        {
            "filterName": "HTTP404Errors",
            "metricTransformations": [
                {
                    "metricValue": "1",
                    "metricNamespace": "MyNamespace",
                    "metricName": "ApacheNotFoundErrorCount"
                }
            ],
            "creationTime": 1399277571078,
            "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
 size]"
        }
    ]
}
```

Agora você pode publicar alguns eventos manualmente:

```
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"
```

Logo após colocar esses exemplos de eventos de log, você pode recuperar a métrica nomeada no CloudWatch console como ApacheNotFoundErrorCount.

Exemplo: contar códigos HTTP 4xx

Como no exemplo anterior, você pode monitorar seus logs de acesso do serviço da Web e monitorar os níveis de código de resposta HTTP. Por exemplo, você pode monitorar todos os erros no nível de HTTP 400 erros. No entanto, é possível especificar um novo filtro de métrica para cada código de retorno.

O exemplo a seguir demonstra como criar uma métrica que inclua todas as 400 respostas de código HTTP a partir de um log de acesso usando o formato de log de acesso do Apache do exemplo Exemplo: contar códigos HTTP 404.

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Escolha o nome do grupo de logs para o servidor Apache.
4. Escolha a `Actions` Criar filtro de métrica.
5. Em Padrão de filtro, insira **`[ip, id, user, timestamp, request, status_code=4*, size]`**.
6. (Opcional) Para testar seu padrão de filtro, em Test Pattern (Testar padrão), insira um ou mais eventos de log a serem usados para testar o padrão. Cada evento de log deve estar dentro de uma linha, porque as quebras de linha são usadas para separar eventos de log na caixa Log event messages (Mensagens do evento de log).
7. Escolha Próximo e, em Nome do filtro, digite **`HTTP4xxErrors`**.
8. Em Detalhes da métrica, em Namespace da métrica, insira **`MyNameSpace`**.
9. Em Metric name (Nome da métrica), insira HTTP4xxErrors.
10. Em Metric value (Valor da métrica), digite 1. Isso especifica que a contagem é aumentada em 1 para cada log que contém um erro 4xx.
11. Em Valor padrão, insira 0 e escolha Próximo.
12. Escolha Criar filtro de métrica.

Para criar um filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
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
```

Você pode usar os seguintes dados em chamadas put-event para testar essa regra. Se você não removeu a regra de monitoramento no exemplo anterior, gerará duas métricas diferentes.

```
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
```

Exemplo: Extrair campos de um log Apache e atribuir dimensões

Às vezes, em vez de contar, é útil usar os valores em eventos de log individuais para valores de métrica. Este exemplo mostra como criar uma regra de extração para criar uma métrica que meça os bytes transferidos por um servidor web Apache.

O exemplo também mostra como atribuir dimensões à métrica que você está criando.

Para criar um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Escolha o nome do grupo de logs para o servidor Apache.
4. Escolha a `Actions` Criar filtro de métrica.
5. Em Padrão de filtro, insira **`[ip, id, user, timestamp, request, status_code, size]`**.
6. (Opcional) Para testar seu padrão de filtro, em Test Pattern (Testar padrão), insira um ou mais eventos de log a serem usados para testar o padrão. Cada evento de log deve estar dentro de uma linha, porque as quebras de linha são usadas para separar eventos de log na caixa Log event messages (Mensagens do evento de log).
7. Escolha Próximo e, em Nome do filtro, digite **`size`**.

8. Em Metric details (Detalhes da métrica), em Metric namespace (Namespace da métrica), insira **`MyNameSpace`**. Como se trata de um novo namespace, verifique se a opção Create new (Criar novo) está selecionada.
9. Em Metric name (Nome da métrica), insira **`BytesTransferred`**
10. Em Valor de métrica, insira **`$size`**.
11. Em Unit (Unidade), selecione Bytes.
12. Para Dimension Name, digite **`IP`**.
13. Em Dimension Value (Valor da dimensão), digite **`$ip`** e escolha Next (Próximo).
14. Escolha Criar filtro de métrica.

Para criar esse filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando

```
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,dimensio
$ip}}'
```

> **Note**
>
> Nesse comando, use esse formato para especificar várias dimensões.
>
> ```
> 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}'
```

Você pode usar os seguintes dados em chamadas log-event para testar essa regra. Isso gerará duas métricas diferentes se você não removeu a regra de monitoramento no exemplo anterior.

```
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
```

Listagem de filtros de métrica

Você pode listar todos os filtros de métrica em um grupo de logs.

Para listar filtros métricos usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. No painel de conteúdo, na lista de grupos de log, na coluna Filtros de métrica, escolha o número de filtros.

   A tela Grupos de logs > Filtros para lista todos os filtros de métrica associados ao grupo de logs.

Para listar filtros métricos usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
aws logs describe-metric-filters --log-group-name MyApp/access.log
```

A seguir está um exemplo de saída:

```
{
    "metricFilters": [
        {
```

```
            "filterName": "HTTP404Errors",
            "metricTransformations": [
                {
                    "metricValue": "1",
                    "metricNamespace": "MyNamespace",
                    "metricName": "ApacheNotFoundErrorCount"
                }
            ],
            "creationTime": 1399277571078,
            "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
 size]"
        }
    ]
}
```

Excluir um filtro de métrica

Uma política é identificada por seu nome e o grupo de logs ao qual ela pertence.

Para excluir um filtro métrico usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. No painel de conteúdo, na lista de grupos de log, na coluna Metric Filter (Filtros de métrica), escolha o número de filtros de métrica para o grupo de logs.
4. Em Metric Filters (Filtros de métrica), selecione a caixa de seleção à direita do nome do filtro que deseja excluir. Escolha Excluir.
5. Quando a confirmação for solicitada, escolha Excluir.

Para excluir um filtro métrico usando o AWS CLI

Em um prompt de comando, execute o seguinte comando:

```
aws logs delete-metric-filter --log-group-name MyApp/access.log \
 --filter-name MyFilterName
```

Real-time processamento de dados de registro com assinaturas

Você pode usar assinaturas para obter acesso a um feed em tempo real de eventos de log do CloudWatch Logs e entregá-lo a outros serviços, como um stream do Amazon Kinesis, um stream do Amazon Data Firehose ou AWS Lambda para processamento, análise ou carregamento personalizados em outros sistemas. Quando os eventos de log são enviados ao serviço de recebimento, eles são codificados em base64 e compactados no formato gzip.

Você também pode usar a centralização de CloudWatch registros para replicar dados de registro de várias contas e regiões em um local central. Para obter mais informações, consulte Cross-account Centralização de registros entre regiões.

Para começar a se inscrever para registrar eventos, crie o recurso de recebimento, como um stream do Amazon Kinesis Data Streams, onde os eventos serão entregues. Um filtro de assinatura define o padrão de filtro a ser usado para filtrar quais eventos de registro são entregues ao seu AWS recurso, bem como informações sobre para onde enviar eventos de registro correspondentes. Os eventos de logs são enviados para o recurso receptor logo após serem ingeridos, geralmente em menos de três minutos.

> **Note**
>
> O padrão de filtro será comparado às versões transformadas dos eventos de logs caso um grupo de logs com uma assinatura utilizar a transformação de log. Para obter mais informações, consulte Transformar logs durante a ingestão.

Você pode criar assinaturas ao nível da conta e ao nível do grupo de logs. Cada conta pode ter um filtro de assinatura ao nível da conta por Região. Cada grupo de registros pode ter até cinco filtros de assinatura associados a ele.

> **Note**
>
> Se o serviço de destino retornar um erro que pode ser repetido, como uma exceção de limitação ou uma exceção de serviço que pode ser repetida (HTTP 5xx, por exemplo), o CloudWatch Logs continuará tentando entregar novamente por até 24 horas. CloudWatch

Os registros não tentarão ser reenviados se o erro for um erro que não possa ser repetido, como ou. AccessDeniedException ResourceNotFoundException Nesses casos, o filtro de assinatura é desativado por até 10 minutos e, em seguida, o CloudWatch Logs tenta enviar os registros novamente para o destino. Enquanto estão desabilitados, os logs são ignorados.

CloudWatch Os registros também produzem CloudWatch métricas sobre o encaminhamento de eventos de registro para assinaturas. Para obter mais informações, consulte Monitoramento com CloudWatch métricas.

Você também pode usar uma assinatura do CloudWatch Logs para transmitir dados de log quase em tempo real para um cluster do Amazon OpenSearch Service. Para obter mais informações, consulte Streaming de dados de CloudWatch registros para o Amazon OpenSearch Service.

Assinaturas só são compatíveis com grupos de logs da classe de logs Padrão. Para obter mais informações sobre classes de logs, consulte Classes de logs.

Note

Os filtros de assinatura podem registrar os eventos em log em lotes para otimizar a transmissão e reduzir a quantidade de chamadas feitas para o destino. O agrupamento em lotes não é garantido, mas é usado quando possível.

Para processamento em lote e análise de dados de registro em um cronograma, considere usarAutomatizando a análise de registros com consultas agendadas. As consultas agendadas executam consultas do CloudWatch Logs Insights automaticamente e entregam resultados para destinos como buckets do Amazon S3 ou ônibus de eventos da Amazon. EventBridge

Note

Os filtros de assinatura garantem a entrega de eventos pelo menos uma vez, enquanto eventos duplicados podem ocorrer ocasionalmente.

Conteúdo

- Conceitos

- Filtros de assinatura ao nível do grupo de logs
- Account-level filtros de assinatura
- Cross-account Assinaturas entre regiões
- Prevenção de 'confused deputy'
- Prevenção de repetição de logs

Conceitos

Cada filtro de assinatura é composto dos seguintes elementos-chave:

padrão de filtro

Uma descrição simbólica de como CloudWatch os registros devem interpretar os dados em cada evento de registro, junto com expressões de filtragem que restringem o que é entregue ao AWS recurso de destino. Para obter mais informações sobre a sintaxe de padrões de filtros, consulte Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura, filtros de eventos de log e Live Tail..

arn de destino

O Amazon Resource Name (ARN) do stream do Amazon Kinesis Data Streams, do Firehose Streams ou da função Lambda que você deseja usar como destino do feed de assinatura.

arn de função

Uma função do IAM que concede aos CloudWatch Logs as permissões necessárias para colocar dados no destino escolhido. Essa função não é necessária para destinos do Lambda porque CloudWatch os registros podem obter as permissões necessárias nas configurações de controle de acesso na própria função do Lambda.

> **Note**
>
> Se o gerente de funções estiver habilitado em sua conta, CloudWatch anexará a função para você e as opções de função descritas aqui (por exemplo, o botão Criar nova função) serão substituídas pela opção Personalizar. Para usar um perfil diferente, escolha Personalizar. Para obter mais informações sobre a criação de funções do IAM, consulte Criação de funções do IAM no Guia do usuário do IAM.

distribuição

O método usado para distribuir dados de log ao destino, quando o destino é um fluxo do Amazon Kinesis Data Streams. Por padrão, os dados de log são agrupados por stream de log. Para obter uma distribuição uniforme, você pode agrupar os dados de log aleatoriamente.

Para assinaturas ao nível do grupo de logs, o seguinte elemento-chave também é incluído:

nome do grupo de logs

O grupo de logs ao qual associar o filtro de assinatura. Todos os eventos de log carregados para esse grupo de logs estariam sujeitos ao filtro de assinatura, e aqueles que correspondem ao filtro são entregues ao serviço de destino que está recebendo os eventos de log correspondentes.

Para assinaturas ao nível da conta, o seguinte elemento-chave também é incluído:

critérios de seleção

Os critérios usados para selecionar a quais grupos de log o filtro de assinatura ao nível da conta será aplicado. Se você não especificar isso, o filtro de assinatura ao nível da conta será aplicado a todos os grupos de logs da conta. Esse campo é usado para evitar loops de log infinitos. Para obter mais informações sobre o problema de loops de log infinitos, consulte Prevenção de repetição de logs.

Os critérios de seleção têm um limite de tamanho de 25 KB.

Para grupos de logs centralizados, os seguintes elementos-chave também estão incluídos. Esses elementos podem ser usados como critérios de seleção de campo para ajudar a identificar a origem dos dados de log, permitindo uma filtragem e análise mais granulares das métricas derivadas de logs centralizados.

@aws.account

Esse campo identifica a ID da AWS conta da qual o evento de registro se originou.

@aws.region

Esse campo identifica a AWS região em que o evento de registro foi gerado.

@source .log

Esse campo identifica o grupo de registros de origem do qual o evento de registro se originou.

Filtros de assinatura ao nível do grupo de logs

Você pode usar um filtro de assinatura com o Amazon Kinesis Data Streams AWS Lambda, Amazon Data Firehose ou Amazon Service. OpenSearch Os logs enviados a um serviço por meio de um filtro de assinatura são codificados na base64 e compactados no formato gzip. Se você estiver usando registros centralizados com o seu AWS Organizations, poderá optar por emitir os campos`@aws.account`,`@aws.region`, e `@source.log` do sistema para identificar quais dados vêm de quais contas, regiões e grupos de registros de origem em sua organização. Esta seção fornece exemplos que você pode seguir para criar um filtro de assinatura de CloudWatch registros que envia dados de log para Firehose, Lambda, Amazon Kinesis Data Streams e Service. OpenSearch

> **Note**
>
> Se você quiser pesquisar dados de log, consulte Sintaxe de filtros e padrões.

Exemplos

- Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams
- Exemplo 2: filtros de assinatura com AWS Lambda
- Exemplo 3: filtros de assinatura com o Amazon Data Firehose
- Exemplo 4: filtros de assinatura com o Amazon OpenSearch Service

Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams

O exemplo a seguir associa um filtro de assinatura a um grupo de logs que contenha eventos do AWS CloudTrail . O filtro de assinatura entrega todas as atividades registradas feitas pelas AWS credenciais “Root” para um stream no Amazon Kinesis Data Streams chamado "". RootAccess Para obter mais informações sobre como enviar AWS CloudTrail eventos para CloudWatch registros, consulte Enviando CloudTrail eventos para CloudWatch registros no Guia do AWS CloudTrail usuário.

> **Note**
>
> Antes de criar o fluxo do , calcule o volume de dados de log que será gerado. Certifique-se de criar um fluxo do com estilhaços suficientes para suportar esse volume. Se o fluxo não tiver um número suficiente de estilhaços, o fluxo de logs será limitado. Para obter mais informações sobre os limites de volume de fluxo, consulte Cotas e limites.
> São feitas novas tentativas de entregáveis com controle de utilização por até 24 horas. Após 24 horas, os entregáveis com falha são descartados.
> Para reduzir o risco de controle de utilização, você pode seguir as seguintes etapas:
>
> - Especifique `random` para `distribution` quando você criar o filtro de assinatura com PutSubscriptionFilter ou https://awscli.amazonaws.com/v2/documentation/api/2.4.18/reference/logs/put-subscription-filter.html put-subscription-filter. Por padrão, a distribuição de filtro de fluxo é feita por fluxo de log e isso pode causar a aplicação de um controle de utilização.
> - Monitore seu stream usando CloudWatch métricas. Isso ajuda você a identificar qualquer controle de utilização e ajustar sua configuração adequadamente. Por exemplo, a `DeliveryThrottling` métrica pode ser usada para rastrear o número de eventos de registro para os quais CloudWatch os registros foram limitados ao encaminhar dados para o destino da assinatura. Para obter mais informações sobre o monitoramento, consulte Monitoramento com CloudWatch métricas.
> - Use o modo de capacidade sob demanda para seu stream no Amazon Kinesis Data Streams. On-demandO modo acomoda instantaneamente suas cargas de trabalho à medida que elas aumentam ou diminuem. Mais informações sobre o modo de capacidade sob demanda, consulte On-demand modo.
> - Restrinja seu padrão de filtro de CloudWatch assinatura para corresponder à capacidade do seu stream no Amazon Kinesis Data Streams. Se você estiver enviando muitos dados para o fluxo, talvez seja necessário reduzir o tamanho do filtro ou ajustar os critérios dele.

Para criar um filtro de assinatura para o Amazon Kinesis Data Streams

1. Crie um stream do de destino usando o seguinte comando:

```
$ C:\>  aws kinesis create-stream --stream-name "RootAccess" --shard-count 1
```

2. Aguarde até que o stream do se torne ativo (isso pode levar um ou dois minutos). Você pode usar o seguinte comando describe-stream do Amazon Kinesis Data Streams para verificar a propriedade. StreamDescription.StreamStatus Além disso, observe o valor StreamDescription.StreamARN, pois ele será necessário em uma etapa posterior:

```
aws kinesis describe-stream --stream-name "RootAccess"
```

A seguir está um exemplo de saída:

```
{
    "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. Crie a função do IAM que concederá permissão ao CloudWatch Logs para colocar dados em seu stream. Primeiro, você precisará criar uma política de confiança em um arquivo (por exemplo, `~/TrustPolicyForCWL-Kinesis.json`). Use um editor de texto para criar esta política. Não use o console do IAM para criá-la.

Esta política inclui uma chave de contexto de condição global `aws:SourceArn` para ajudar a evitar o problema de segurança 'confused deputy'. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
  "Statement": {
```

```
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "ArnLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Observe o valor Role.Arn retornado, pois ele também será necessário em uma etapa posterior:

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

A seguir, veja um exemplo de saída.

```
{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": { "arn:aws:logs:region:123456789012:*" }
                    }
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisRole"
    }
}
```

5. Crie uma política de permissões para definir quais ações CloudWatch os registros podem fazer em sua conta. Primeiro, você criará uma política de permissões em um arquivo (por exemplo, `~/PermissionsForCWL-Kinesis.json`). Use um editor de texto para criar esta política. Não use o console do IAM para criá-la.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/RootAccess"
    }
  ]
}
```

6. Associe a política de permissões com a função usando o seguinte comando put-role-policy:

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. Depois que o stream estiver no estado Ativo e você tiver criado a função do IAM, você poderá criar o filtro de assinatura de CloudWatch registros. O filtro de assinatura inicia imediatamente o fluxo de dados de log em tempo real a partir do grupo de logs escolhido para o fluxo do :

```
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. Depois de configurar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro recebidos que correspondem ao padrão do filtro para o seu stream. Você pode verificar se isso está acontecendo pegando um iterador de fragmentos do Amazon Kinesis Data Streams e usando o comando get-records do Amazon Kinesis Data Streams para buscar alguns registros do Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator --stream-name RootAccess --shard-id
 shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
```

Observe que talvez seja necessário fazer essa chamada algumas vezes antes que o Amazon Kinesis Data Streams comece a retornar dados.

É necessário esperar para ver uma resposta com um conjunto de registros. O atributo Data em um registro do Amazon Kinesis Data Streams é codificado em base64 e compactado com o formato gzip. Você pode examinar os dados brutos na linha de comando usando os seguintes comandos Unix:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "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\"}"
        }
    ]
}
```

Os principais elementos na estrutura de dados acima são os seguintes:

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

messageType

As mensagens de dados usarão o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros do Amazon Kinesis Data Streams do tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade "id" é um identificador exclusivo de cada evento de log.

Exemplo 2: filtros de assinatura com AWS Lambda

Neste exemplo, você criará um filtro de assinatura de CloudWatch registros que envia dados de registro para sua AWS Lambda função.

> **Note**
>
> Antes de criar a função Lambda, calcule o volume de dados de log que será gerado. Lembre-se de criar uma função que suporte esse volume. Se a função não tiver um volume suficiente, o fluxo de logs será limitado. Para obter mais informações sobre limites do Lambda, consulte Limites do AWS Lambda.

Para criar um filtro de assinatura para o Lambda

1. Crie a AWS Lambda função.

   Certifique-se de ter configurado a função de execução do Lambda. Para obter mais informações, consulte Etapa 2.2: Criar uma função do IAM (função de execução) no Guia do desenvolvedor do AWS Lambda .

2. Abra um editor de texto e crie um arquivo chamado `helloWorld.js` com o seguinte conteúdo:

```
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. Compacte o arquivo hello World.js e salve-o com o nome`helloWorld.zip`.
4. Use o comando a seguir, no qual a função é a função de execução do Lambda configurada na primeira etapa:

```
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. Conceda ao CloudWatch Logs a permissão para executar sua função. Use o comando a seguir, substituindo o espaço reservado conta pela sua própria conta e o espaço reservado grupo de logs pelo grupo de logs a ser processado:

```
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. Crie um filtro de assinatura usando o seguinte comando, substituindo o espaço reservado conta pela sua própria conta e o espaço reservado grupo de logs pelo grupo de logs a ser processado:

```
aws logs put-subscription-filter \
    --log-group-name myLogGroup \
    --filter-name demo \
    --filter-pattern "" \
    --destination-arn arn:aws:lambda:region:123456789123:function:helloworld
```

7. (Opcional) Teste usando um exemplo de evento de log. Em um prompt de comando, execute o seguinte comando, que colocará uma mensagem de log simples no stream assinado.

   Para ver a saída da sua função Lambda, navegue até a função Lambda, onde você verá a saída em/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\"}]"
```

É necessário esperar para ver uma resposta com uma matriz do Lambda. O atributo Data (Dados) em um registro do Lambda é codificado em base64 e compactado com o formato gzip. A carga útil real recebida pelo Lambda está no seguinte formato `{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }`. Você pode examinar os dados brutos na linha de comando usando os seguintes comandos Unix:

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "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\"}"
        }
    ]
}
```

Os principais elementos na estrutura de dados acima são os seguintes:

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

messageType

As mensagens de dados usarão o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros Lambda com o tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade "id" é um identificador exclusivo de cada evento de log.

Exemplo 3: filtros de assinatura com o Amazon Data Firehose

Neste exemplo, você criará uma assinatura do CloudWatch Logs que enviará quaisquer eventos de log recebidos que correspondam aos filtros definidos para o seu stream de entrega do Amazon Data Firehose. Os dados enviados do CloudWatch Logs para o Amazon Data Firehose já estão compactados com a compactação gzip de nível 6, portanto, você não precisa usar a compactação no seu stream de entrega do Firehose. Você pode usar o atributo de descompactação no Firehose para

descompactar automaticamente os logs. Para obter mais informações, consulte Enviar CloudWatch registros para o Firehose.

> **Note**
>
> Antes de criar o fluxo do Firehose, calcule o volume de dados de log que será gerado. Certifique-se de criar um fluxo do Firehose que possa lidar com esse volume. Se o fluxo não puder suportar o volume, o fluxo de logs será limitado. Para obter mais informações sobre os limites de volume de fluxo do Firehose, consulte Amazon Data Firehose Data Limits.

Para criar um filtro de assinatura para o Firehose

1. Crie um bucket do Amazon Simple Storage Service (Amazon S3). Recomendamos que você use um bucket criado especificamente para CloudWatch registros. No entanto, se você desejar usar um bucket existente, vá para a etapa 2.

   Execute o comando a seguir, substituindo o espaço reservado Region (Região) pela região que você deseja usar:

   ```
   aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration
    LocationConstraint=region
   ```

   A seguir está um exemplo de saída:

   ```
   {
       "Location": "/amzn-s3-demo-bucket2"
   }
   ```

2. Crie o perfil do IAM que concede ao Amazon Data Firehose permissão para colocar dados no bucket do Amazon S3.

   Para obter mais informações, consulte Controlling Access with Amazon Data Firehose no Amazon Data Firehose Developer Guide.

   Primeiro, use um editor de texto para criar uma política de confiança em um arquivo `~/TrustPolicyForFirehose.json`, da seguinte maneira:

   ```
   {
     "Statement": {
   ```

```
    "Effect": "Allow",
    "Principal": { "Service": "firehose.amazonaws.com" },
    "Action": "sts:AssumeRole"
    }
}
```

3. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Observe o valor Role.Arn retornado, pois ele será necessário em uma etapa posterior:

```
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": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "FirehosetoS3Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/FirehosetoS3Role"
    }
}
```

4. Crie uma política de permissões para definir quais ações o Firehose pode fazer em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/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. Associe a política de permissões com a função usando o seguinte comando put-role-policy:

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-
Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. Crie um fluxo de entrega do Firehose de destino como se segue, substituindo os valores de espaço reservado para RoleARN e BucketARN pelos ARNs do perfil e do bucket que você criou:

```
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"}'
```

Observe que o Firehose usa automaticamente um prefixo no formato de horário YYYY/MM/DD/HH UTC para objetos do Amazon S3 entregues. Você pode especificar um prefixo extra a ser incluído na frente do prefixo de formato de tempo. Se o prefixo terminar com uma barra (/), ele aparecerá como uma pasta no bucket do Amazon S3.

7. Aguarde até que o stream fique ativo (isso pode levar alguns minutos). Você pode usar o comando Firehose describe-delivery-stream para verificar a propriedade. DeliveryStreamDescription.DeliveryStreamStatus Além disso, observe o valor DeliveryStreamDescription.DeliveryStreamARN, pois ele será necessário em uma etapa posterior:

```
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. Crie a função do IAM que concede permissão ao CloudWatch Logs para colocar dados em seu stream de entrega do Firehose. Primeiro, use um editor de texto para criar uma política de confiança em um arquivo `~/TrustPolicyForCWL.json`:

   Esta política inclui uma chave de contexto de condição global `aws:SourceArn` para ajudar a evitar o problema de segurança 'confused deputy'. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
         "ArnLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
```

```
     }
  }
}
```

9. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Observe o valor Role.Arn retornado, pois ele será necessário em uma etapa posterior:

```
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": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
    }
}
```

10. Crie uma política de permissões para definir quais ações CloudWatch os registros podem fazer em sua conta. Primeiro, use um editor de texto para criar um arquivo de política de permissões (por exemplo, ~/PermissionsForCWL.json):

```
{
    "Statement":[
      {
```

```
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
      }
    ]
}
```

11. Associe a política de permissões com a função usando o comando put-role-policy:

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. Depois que o stream de entrega do Amazon Data Firehose estiver ativo e você tiver criado a função IAM, você poderá criar o filtro de assinatura do CloudWatch Logs. O filtro de assinatura inicia imediatamente o fluxo de dados de log em tempo real do grupo de logs escolhido para o fluxo de entrega do Amazon Data Firehose:

```
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. Depois de configurar o filtro de assinatura, o CloudWatch Logs encaminhará todos os eventos de log recebidos que correspondam ao padrão do filtro para seu stream de entrega do Amazon Data Firehose. Seus dados começarão a aparecer no Amazon S3 com base no intervalo de buffer de tempo definido no fluxo de entrega do Amazon Data Firehose. Quando tiver passado tempo suficiente, você poderá conferir seus dados verificando o bucket do Amazon S3.

```
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": {}
}
```

Os dados no objeto do Amazon S3 são compactados com o formato gzip. Você pode examinar os dados brutos na linha de comando usando o seguinte comando Unix:

```
zcat testfile.gz
```

Exemplo 4: filtros de assinatura com o Amazon OpenSearch Service

Neste exemplo, você criará uma assinatura do CloudWatch Logs que envia eventos de registro de entrada que correspondam aos filtros definidos para o seu domínio OpenSearch de serviço.

Para criar um filtro de assinatura para o OpenSearch Service

1. Crie um domínio OpenSearch de serviço. Para obter mais informações, consulte Criação de domínios OpenSearch de serviço
2. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
3. No painel de navegação, escolha Grupos de logs.
4. Selecione o nome do grupo de logs.
5. Escolha Ações, Filtros de assinatura, Criar filtro OpenSearch de assinatura do Amazon Service.
6. Escolha se deseja fazer streaming para um cluster nessa conta ou em outra conta.
   - Se você escolheu Esta conta, selecione o domínio que criou na etapa 1.
   - Se você escolher Outra conta, insira o ARN e o endpoint desse domínio.
7. Caso tenha escolhido outra conta, forneça o ARN do domínio e o endpoint.
8. Para o cluster do Amazon OpenSearch Service, escolha o nome do cluster em que os dados do grupo de logs serão entregues.
9. Escolha um formato de log.
10. Em Padrão de filtro de assinatura, digite os termos ou o padrão a ser localizado nos eventos de logs. Isso garante que você envie somente os dados que lhe interessam para seu cluster OpenSearch de serviços. Para obter mais informações, consulte Sintaxe de padrões de filtros para filtros de métricas.
11. (Opcional) Em Select log data to test (Selecionar dados de log para testar), selecione um fluxo de logs e escolha Test pattern (Testar padrão) para verificar se o filtro de pesquisa está retornando os resultados esperados.
12. Selecione Start streaming (Iniciar transmissão).

Account-level filtros de assinatura

> **Important**
>
> Existe o risco de causar um loop recursivo infinito com filtros de assinatura, que pode levar a um grande aumento de custos de ingestão se não for resolvido. Para reduzir esse risco, recomendamos que você use critérios de seleção nos filtros de assinatura ao nível da conta para excluir os grupos de logs que ingerem dados de log dos recursos que fazem parte do fluxo de trabalho de entrega de assinatura. Para obter mais informações sobre esse problema e determinar quais grupos de logs devem ser excluídos, consulte Prevenção de repetição de logs.

Você pode definir uma política de assinatura ao nível da conta que inclua um subconjunto de grupos de logs na conta. A política de assinatura da conta pode funcionar com o Amazon Kinesis Data Streams ou o AWS Lambda Amazon Data Firehose. Os logs enviados a um serviço por meio de uma política de assinatura ao nível da conta são codificados na base64 e compactados com o formato gzip. Esta seção fornece exemplos que você pode seguir para criar uma assinatura em nível de conta para Amazon Kinesis Data Streams, Lambda e Firehose.

> **Note**
>
> Para ver uma lista de todas as políticas de filtro de assinatura de sua conta, use o comando `describe-account-policies` com um valor de `SUBSCRIPTION_FILTER_POLICY` para o parâmetro `--policy-type`. Para obter mais informações, consulte describe-account-policies.

Exemplos

- Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams
- Exemplo 2: filtros de assinatura com AWS Lambda
- Exemplo 3: filtros de assinatura com o Amazon Data Firehose

Exemplo 1: filtros de assinatura com o Amazon Kinesis Data Streams

Antes de criar um stream de dados do Amazon Kinesis Data Streams para usar com uma política de assinatura em nível de conta, calcule o volume de dados de log que serão gerados. Certifique-se de criar um fluxo do com estilhaços suficientes para suportar esse volume. Se um fluxo não tiver fragmentos suficientes, será submetido ao controle de utilização. Para obter mais informações sobre limites de volume de streams, consulte Cotas e limites na documentação do Amazon Kinesis Data Streams.

Warning

Como os eventos de log de vários grupos de log são encaminhados para o destino, existe o risco da aplicação de controle de utilização. São feitas novas tentativas de entregáveis com controle de utilização por até 24 horas. Após 24 horas, os entregáveis com falha são descartados.
Para reduzir o risco de controle de utilização, você pode seguir as seguintes etapas:

- Monitore seu stream do Amazon Kinesis Data Streams com CloudWatch métricas. Isso ajuda a identificar o controle de utilização e ajustar a configuração adequadamente. Por exemplo, a `DeliveryThrottling` métrica rastreia o número de eventos de registro para CloudWatch os quais os registros foram limitados ao encaminhar dados para o destino da assinatura. Para obter mais informações, consulte Monitoramento com CloudWatch métricas.
- Use o modo de capacidade sob demanda para seu stream no Amazon Kinesis Data Streams. On-demandO modo acomoda instantaneamente suas cargas de trabalho à medida que elas aumentam ou diminuem. Para obter mais informações, consulte On-demand o modo.
- Restrinja seu padrão de filtro de assinatura do CloudWatch Logs para corresponder à capacidade do seu stream no Amazon Kinesis Data Streams. Se você estiver enviando muitos dados para o fluxo, talvez seja necessário reduzir o tamanho do filtro ou ajustar os critérios dele.

O exemplo a seguir usa uma política de assinatura em nível de conta para encaminhar todos os eventos de log para um stream no Amazon Kinesis Data Streams. O padrão de filtro combina todos os eventos de log com o texto `Test` e os encaminha para o stream no Amazon Kinesis Data Streams.

Para criar uma política de assinatura em nível de conta para o Amazon Kinesis Data Streams

1. Crie um stream do de destino usando o seguinte comando:

```
$ C:\>  aws kinesis create-stream –stream-name "TestStream" –shard-count 1
```

2. Aguarde alguns minutos para que o fluxo fique ativo. Você pode verificar se o fluxo está ativo usando o comando describe-stream para verificar a propriedade. StreamDescription.StreamStatus

```
aws kinesis describe-stream --stream-name "TestStream"
```

   A seguir está um exemplo de saída:

```
{
    "StreamDescription": {
        "StreamStatus": "ACTIVE",
        "StreamName": "TestStream",
        "StreamARN": "arn:aws:kinesis:region:123456789012:stream/TestStream",
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "EndingHashKey": "EXAMPLE84634633746074317682114555",
                    "StartingHashKey": "0"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
                    "EXAMPLE688818456679503831981458784591352702181572610"
                }
            }
        ]
    }
}
```

3. Crie a função do IAM que concederá permissão ao CloudWatch Logs para colocar dados em seu stream. Primeiro, você precisará criar uma política de confiança em um arquivo (por exemplo, `~/TrustPolicyForCWL-Kinesis.json`). Use um editor de texto para criar esta política.

Esta política inclui uma chave de contexto de condição global `aws:SourceArn` para ajudar a evitar o problema de segurança 'confused deputy'. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "ArnLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Observe o valor Role.Arn retornado, pois ele também será necessário em uma etapa posterior:

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

A seguir, veja um exemplo de saída.

```
{
    "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"
    }
}
```

5. Crie uma política de permissões para definir quais ações CloudWatch os registros podem fazer em sua conta. Primeiro, você criará uma política de permissões em um arquivo (por exemplo, `~/PermissionsForCWL-Kinesis.json`). Use um editor de texto para criar esta política. Não use o console do IAM para criá-la.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/TestStream"
    }
  ]
}
```

6. Associe a política de permissões com a função usando o seguinte comando put-role-policy:

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. Depois que o stream estiver no estado Ativo e você tiver criado a função do IAM, você poderá criar a política de filtro de assinatura de CloudWatch registros. A política inicia imediatamente o fluxo de dados de log em tempo real para o seu fluxo. Neste exemplo, todos os eventos de log que contêm a string ERROR são transmitidos, exceto os eventos dos grupos de log denominados `LogGroupToExclude1` e `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. Depois de configurar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro recebidos que correspondem ao padrão do filtro e aos critérios de seleção para o seu stream.

   O campo `selection-criteria` é opcional, mas é importante para excluir de um filtro de assinatura os grupos de logs que possam causar uma repetição infinita de logs. Para obter mais informações sobre esse problema e determinar quais grupos de logs devem ser excluídos, consulte Prevenção de repetição de logs. Atualmente, NOT IN é o único operador compatível com `selection-criteria`.

   Você pode verificar o fluxo de eventos de log usando um iterador de fragmentos do Amazon Kinesis Data Streams e usando o `get-records` comando Amazon Kinesis Data Streams para buscar alguns registros do Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator --stream-name TestStream --shard-id
 shardId-000000000000 --shard-iterator-type TRIM_HORIZON
```

```
{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
}
```

```
aws kinesis get-records --limit 10 --shard-iterator "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL
+wev+e2P4djJg4L9wmXKvQYoE+rMUiFq
+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f+OIK8zM5My8ID
+g6rMo7UKWeI4+IWiK2OSh0uP"
```

   Talvez seja necessário usar esse comando algumas vezes antes que o Amazon Kinesis Data Streams comece a retornar dados.

   É necessário esperar para ver uma resposta com um conjunto de registros. O atributo Data em um registro do Amazon Kinesis Data Streams é codificado em base64 e compactado com o

formato gzip. Você pode examinar os dados brutos na linha de comando usando os seguintes comandos Unix:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "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"
}
```

Os principais elementos da estrutura de dados são os seguintes:

messageType

As mensagens de dados usarão o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros do Amazon Kinesis Data Streams com o tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade "id" é um identificador exclusivo de cada evento de log.

policyLevel

O nível em que a política foi aplicada. "ACCOUNT_LEVEL_POLICY" é a `policyLevel` para uma política de filtro de assinatura ao nível da conta.

Exemplo 2: filtros de assinatura com AWS Lambda

Neste exemplo, você criará uma política de filtro de assinatura no nível da conta do CloudWatch Logs que envia dados de registro para sua AWS Lambda função.

> **Warning**
>
> Antes de criar a função Lambda, calcule o volume de dados de log que será gerado. Lembre-se de criar uma função que suporte esse volume. Se a função não puder lidar com o volume, o fluxo de logs será submetido ao controle de utilização. Como os eventos de log de todos os grupos de logs ou de um subconjunto dos grupos de logs da conta são encaminhados para

o destino, existe o risco da aplicação de controle de utilização. Para obter mais informações sobre limites do Lambda, consulte Limites do AWS Lambda.

Para criar uma política de filtro de assinatura ao nível da conta para o Lambda

1. Crie a AWS Lambda função.

   Certifique-se de ter configurado a função de execução do Lambda. Para obter mais informações, consulte Etapa 2.2: Criar uma função do IAM (função de execução) no Guia do desenvolvedor do AWS Lambda .

2. Abra um editor de texto e crie um arquivo chamado `helloWorld.js` com o seguinte conteúdo:

```
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. Compacte o arquivo hello World.js e salve-o com o nome`helloWorld.zip`.

4. Use o comando a seguir, no qual a função é a função de execução do Lambda configurada na primeira etapa:

```
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. Conceda ao CloudWatch Logs a permissão para executar sua função. Use o comando a seguir, substituindo a conta do espaço reservado por sua própria conta.

```
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. Crie uma política de filtro de assinatura ao nível da conta usando o comando a seguir, substituindo a conta do espaço reservado por sua própria conta. Neste exemplo, todos os eventos de log que contêm a string ERROR são transmitidos, exceto os eventos dos grupos de log denominados `LogGroupToExclude1` e `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"
```

   Depois de configurar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro recebidos que correspondem ao padrão do filtro e aos critérios de seleção para o seu stream.

   O campo `selection-criteria` é opcional, mas é importante para excluir de um filtro de assinatura os grupos de logs que possam causar uma repetição infinita de logs. Para obter mais informações sobre esse problema e determinar quais grupos de logs devem ser excluídos, consulte Prevenção de repetição de logs. Atualmente, NOT IN é o único operador compatível com `selection-criteria`.

7. (Opcional) Teste usando um exemplo de evento de log. Em um prompt de comando, execute o seguinte comando, que colocará uma mensagem de log simples no stream assinado.

   Para ver a saída da sua função Lambda, navegue até a função Lambda, onde você verá a saída em/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\"}]"
```

É necessário esperar para ver uma resposta com uma matriz do Lambda. O atributo Data (Dados) em um registro do Lambda é codificado em base64 e compactado com o formato gzip. A carga útil real recebida pelo Lambda está no seguinte formato `{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }`. Você pode examinar os dados brutos na linha de comando usando os seguintes comandos Unix:

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "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**
>
> O filtro de assinatura ao nível da conta não será aplicado ao grupo de logs da função do Lambda de destino. Isso evita uma repetição infinita de logs que pode levar a um aumento nas faturas de ingestão. Para ter mais informações sobre esse problema, consulte Prevenção de repetição de logs.

Os principais elementos da estrutura de dados são os seguintes:

messageType

As mensagens de dados usarão o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros do Amazon Kinesis Data Streams com o tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade "id" é um identificador exclusivo de cada evento de log.

policyLevel

O nível em que a política foi aplicada. "ACCOUNT_LEVEL_POLICY" é a `policyLevel` para uma política de filtro de assinatura ao nível da conta.

Exemplo 3: filtros de assinatura com o Amazon Data Firehose

Neste exemplo, você criará uma política de filtro de assinatura no nível da conta do CloudWatch Logs que enviará eventos de log recebidos que correspondam aos filtros definidos para seu stream de entrega do Amazon Data Firehose. Os dados enviados do CloudWatch Logs para o Amazon Data Firehose já estão compactados com a compactação gzip de nível 6, portanto, você não precisa usar a compactação no seu stream de entrega do Firehose. Você pode usar o atributo de descompactação no Firehose para descompactar automaticamente os logs. Para obter mais informações, consulte Gravando no Kinesis Data Firehose usando CloudWatch registros.

> **Warning**
>
> Antes de criar o fluxo do Firehose, calcule o volume de dados de log que será gerado. Certifique-se de criar um fluxo do Firehose que possa lidar com esse volume. Se o fluxo não puder suportar o volume, o fluxo de logs será limitado. Para obter mais informações sobre os limites de volume de fluxo do Firehose, consulte Amazon Data Firehose Data Limits.

Para criar um filtro de assinatura para o Firehose

1. Crie um bucket do Amazon Simple Storage Service (Amazon S3). Recomendamos que você use um bucket criado especificamente para CloudWatch registros. No entanto, se você desejar usar um bucket existente, vá para a etapa 2.

   Execute o comando a seguir, substituindo o espaço reservado Region (Região) pela região que você deseja usar:

   ```
   aws s3api create-bucket --bucket amzn-s3-demo-bucket2 --create-bucket-configuration
    LocationConstraint=region
   ```

   A seguir está um exemplo de saída:

   ```
   {
   ```

```
    "Location": "/amzn-s3-demo-bucket2"
}
```

2. Crie o perfil do IAM que concede ao Amazon Data Firehose permissão para colocar dados no bucket do Amazon S3.

   Para obter mais informações, consulte Controlling Access with Amazon Data Firehose no Amazon Data Firehose Developer Guide.

   Primeiro, use um editor de texto para criar uma política de confiança em um arquivo ~/TrustPolicyForFirehose.json, da seguinte maneira:

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "firehose.amazonaws.com" },
    "Action": "sts:AssumeRole"
    }
}
```

3. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Anote o Role.Arn valor retornado, pois você precisará dele em uma etapa posterior:

```
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. Crie uma política de permissões para definir quais ações o Firehose pode fazer em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/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. Associe a política de permissões com a função usando o seguinte comando put-role-policy:

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-
Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. Crie um fluxo de entrega do Firehose de destino como se segue, substituindo os valores de espaço reservado para RoleARN e BucketARN pelos ARNs do perfil e do bucket que você criou:

```
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"}'
```

O NFirehose usa automaticamente um prefixo no formato de hora YYYY/MM/DD/HH UTC para objetos do Amazon S3 entregues. Você pode especificar um prefixo extra a ser incluído na

frente do prefixo de formato de tempo. Se o prefixo terminar com uma barra (/), ele aparecerá como uma pasta no bucket do Amazon S3.

7. Aguarde alguns minutos para o fluxo ficar ativo. Você pode usar o comando Firehose describe-delivery-stream para verificar a propriedade. DeliveryStreamDescription.DeliveryStreamStatus Além disso, observe o valor DeliveryStreamDescription.DeliveryStreamARN, pois ele será necessário em uma etapa posterior:

```
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. Crie a função do IAM que concede permissão ao CloudWatch Logs para colocar dados em seu stream de entrega do Firehose. Primeiro, use um editor de texto para criar uma política de confiança em um arquivo `~/TrustPolicyForCWL.json`:

Esta política inclui uma chave de contexto de condição global `aws:SourceArn` para ajudar a evitar o problema de segurança 'confused deputy'. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
         "ArnLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
     }
  }
}
```

9. Use o comando create-role para criar a função do IAM especificando o arquivo de política de confiança. Anote o Role.Arn valor retornado, pois você precisará dele em uma etapa posterior:

```
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": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
```

```
        "RoleName": "CWLtoKinesisFirehoseRole",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/CWLtoKinesisFirehoseRole"
    }
}
```

10. Crie uma política de permissões para definir quais ações CloudWatch os registros podem fazer em sua conta. Primeiro, use um editor de texto para criar um arquivo de política de permissões (por exemplo, ~/PermissionsForCWL.json):

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
      }
    ]
}
```

11. Associe a política de permissões com a função usando o comando put-role-policy:

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. Depois que o stream de entrega do Amazon Data Firehose estiver ativo e você tiver criado a função IAM, você poderá criar a política de filtro de assinatura no nível da conta do CloudWatch Logs. A política inicia imediatamente o fluxo de dados de log em tempo real do grupo de logs escolhido para o fluxo de entrega do Amazon Data Firehose:

```
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. Depois de configurar o filtro de assinatura, o CloudWatch Logs encaminha os eventos de log recebidos que correspondem ao padrão do filtro para seu stream de entrega do Amazon Data Firehose.

    O campo `selection-criteria` é opcional, mas é importante para excluir de um filtro de assinatura os grupos de logs que possam causar uma repetição infinita de logs. Para obter mais informações sobre esse problema e determinar quais grupos de logs devem ser excluídos, consulte Prevenção de repetição de logs. Atualmente, NOT IN é o único operador compatível com `selection-criteria`.

    Seus dados começarão a aparecer no Amazon S3 com base no intervalo de buffer de tempo definido no fluxo de entrega do Amazon Data Firehose. Quando tiver passado tempo suficiente, você poderá conferir seus dados verificando o bucket do Amazon S3.

```
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": {}
}
```

Os dados no objeto do Amazon S3 são compactados com o formato gzip. Você pode examinar os dados brutos na linha de comando usando o seguinte comando Unix:

```
zcat testfile.gz
```

Cross-account Assinaturas entre regiões

Você pode colaborar com o proprietário de uma AWS conta diferente e receber seus eventos de log em seus AWS recursos, como um stream do Amazon Kinesis ou do Amazon Data Firehose (isso é conhecido como compartilhamento de dados entre contas). Por exemplo, esses dados de eventos de log podem ser lidos de um stream centralizado do Amazon Kinesis Data Streams ou Firehose para realizar processamento e análise personalizados. O processamento personalizado é especialmente útil quando você colabora e analisa dados entre várias contas.

Por exemplo, um grupo de segurança de informações de uma empresa pode querer analisar dados de detecção de invasões em tempo real ou comportamentos anormais para que possa realizar uma auditoria de contas em todas as divisões da empresa, coletando seus logs de produção federados para processamento central. Um fluxo em tempo real de dados de eventos entre essas contas pode ser montado e entregue aos grupos de segurança da informação, que podem usar o Amazon Kinesis Data Streams para anexar os dados aos seus sistemas analíticos de segurança existentes.

> **Note**
>
> O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o recurso da AWS para o qual o destino aponta pode estar localizado em uma região diferente. Nos exemplos das seções a seguir, todos os Region-specific recursos são criados no Leste dos EUA (Norte da Virgínia).

Se você configurou AWS Organizations e está trabalhando com contas de membros, pode usar a centralização de registros para coletar dados de registro das contas de origem em uma conta de monitoramento central.

Ao trabalhar com grupos de logs centralizados, você pode usar essas dimensões de campos do sistema ao criar filtros de assinatura:

- `@aws.account`- Essa dimensão representa a ID da AWS conta da qual o evento de registro se originou.
- `@aws.region`- Essa dimensão representa a AWS região em que o evento de log foi gerado.
- `@source.log`- Essa dimensão representa o grupo de registros de origem do qual o evento de registro se originou.

Essas dimensões ajudam a identificar a origem dos dados de log, permitindo uma filtragem e análise mais granulares das métricas derivadas de logs centralizados.

Tópicos

- Cross-account Compartilhamento de dados de log entre regiões usando o Amazon Kinesis Data Streams
- Cross-account Compartilhamento de dados de registro entre regiões usando Firehose
- Cross-account Assinaturas em nível de conta entre regiões usando o Amazon Kinesis Data Streams
- Cross-account Assinaturas em nível de conta entre regiões usando Firehose

Cross-account Compartilhamento de dados de log entre regiões usando o Amazon Kinesis Data Streams

Ao criar uma assinatura entre contas, você pode especificar uma única conta ou organização para ser o remetente. Se você especificar uma organização, esse procedimento permitirá que todas as contas da organização enviem logs para a conta do receptor.

Para compartilhar dados de log entre contas, você precisa estabelecer um remetente e um destinatário dos dados de log:

- Remetente de dados de registro — obtém as informações de destino do destinatário e informa ao CloudWatch Logs que está pronto para enviar seus eventos de registro para o destino especificado. Nos procedimentos do restante desta seção, o remetente dos dados de registro é mostrado com um número de AWS conta fictício de 111111111111.

  Se você quiser que várias contas em uma organização enviem logs para uma conta de destinatário, você pode criar uma política que conceda a todas as contas da organização a permissão para enviar logs para a conta do destinatário. Você ainda precisa configurar filtros de assinatura diferentes para cada conta de remetente.
- Destinatário dos dados de log — configura um destino que encapsula um stream do Amazon Kinesis Data Streams e informa aos CloudWatch Logs que o destinatário deseja receber dados de log. O destinatário, então, compartilha as informações sobre esse destino com o remetente. Nos procedimentos do restante desta seção, o destinatário dos dados de log é mostrado com um número de AWS conta fictício de 999999999999.

Para começar a receber eventos de registro de usuários de várias contas, o destinatário dos dados de registro primeiro cria um destino de CloudWatch registros. Cada destino consiste nos seguintes elementos-chave:

Nome do destino

O nome do destino que você deseja criar.

ARN de destino

O nome de recurso da Amazon (ARN) do AWS recurso que você deseja usar como destino do feed de assinatura.

ARN de função

Uma função AWS Identity and Access Management (IAM) que concede aos CloudWatch Logs as permissões necessárias para colocar dados no stream escolhido.

Política de acesso

Documento de políticas do IAM (no formato JSON, gravado usando a gramática de políticas do IAM) que controla o conjunto de usuários que têm permissão para gravar em seu destino.

> **Note**
>
> O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o AWS recurso para o qual o destino aponta pode estar localizado em uma região diferente. Nos exemplos das seções a seguir, todos os Region-specific recursos são criados no Leste dos EUA (Norte da Virgínia).

Tópicos

- Como configurar uma nova assinatura entre contas
- Atualizar uma assinatura existente entre contas

Como configurar uma nova assinatura entre contas

Siga as etapas nestas seções para configurar uma nova assinatura de log entre contas.

Tópicos

- Etapa 1: criar um destino
- Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.
- Etapa 3: permissões Add/validate do IAM para o destino entre contas
- Etapa 4: criar um filtro de assinatura
- Validação do fluxo de eventos de logs
- Modificar a associação de destino no runtime

Etapa 1: criar um destino

> **Important**
>
> As etapas deste procedimento devem ser processadas na conta destinatária dos dados do log.

Neste exemplo, a conta do destinatário dos dados de registro tem um ID de conta de 999999999999, enquanto o ID da AWS conta do remetente dos dados de registro é 111111111111. AWS

Este exemplo cria um destino usando um stream do Amazon Kinesis Data Streams chamado RecipientStream e uma função que permite que CloudWatch os Logs gravem dados nele.

Quando o destino é criado, o CloudWatch Logs envia uma mensagem de teste para o destino em nome da conta do destinatário. Quando o filtro de assinatura estiver ativo posteriormente, o CloudWatch Logs envia eventos de registro para o destino em nome da conta de origem.

Para criar um destino

1. Na conta do destinatário, crie um stream de destino no Amazon Kinesis Data Streams. Em um prompt de comando, digite:

   ```
   aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
   ```

2. Aguarde até que o fluxo do fique ativo. Você pode usar o comando aws kinesis describe-stream para verificar a propriedade. StreamDescription.StreamStatus Além disso, anote o StreamDescription.StreamARN valor porque você o passará para o CloudWatch Logs posteriormente:

   ```
   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": "340282366920938463463374607431768EXAMPLE",
   ```

```
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
 "495511352186888184566795038319814587845913527 0218EXAMPLE"
        }
      }
    ]
  }
}
```

Pode levar um ou dois minutos para o seu stream aparecer no estado ativo.

3. Crie a função do IAM que concede aos CloudWatch Logs a permissão para colocar dados em seu stream. Primeiro, você precisará criar uma política de confiança em um arquivo ~/TrustPolicyForCWL.json. Use um editor de texto para criar esse arquivo de política, não use o console do IAM.

   Esta política inclui uma chave de contexto de condição global `aws:SourceArn` que especifica o `sourceAccountId` para evitar o problema de segurança confused deputy. Se você ainda não souber o ID da conta de origem na primeira chamada, recomendamos que você coloque o ARN de destino no campo ARN de origem. Nas chamadas subsequentes, você deve definir o ARN de origem como o ARN de origem real que você coletou da primeira chamada. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Condition": {
            "ArnLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
```

```
}
```

4. Use o comando aws iam create-role para criar a função do IAM especificando o arquivo de política de confiança. Anote o Role.Arn valor retornado, pois ele também será passado para o CloudWatch Logs posteriormente:

```
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": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. Crie uma política de permissões para definir quais ações CloudWatch os registros podem realizar em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/: PermissionsForCWL.json

```
{
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. Associe a política de permissões à função usando o comando 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. Depois que o stream estiver no estado ativo e você tiver criado a função do IAM, você poderá criar o destino de CloudWatch registros.

   a. Esta etapa não associará uma política de acesso ao seu destino e é apenas a primeira etapa de duas que concluirá uma criação de destino. Anote o DestinationArn que for retornado na carga útil:

```
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. Depois que a etapa 7a for concluída, na conta destinatária dos dados de log, associe uma política de acesso ao destino. Essa política deve especificar a PutSubscriptionFilter ação logs: e conceder permissão à conta do remetente para acessar o destino.

      A política concede permissão à AWS conta que envia registros. Você pode especificar apenas essa conta na política ou, se a conta de remetente for parte de uma organização, a política poderá especificar o ID da organização. Dessa forma, você pode criar apenas uma

política para permitir que várias contas em uma organização enviem logs para essa conta de destino.

Use um editor de texto para criar um arquivo chamado `~/AccessPolicy.json` com uma das seguintes declarações de política.

Este primeiro exemplo de política permite que todas as contas na organização que tenham um ID de `o-1234567890` enviem logs à conta do destinatário.

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"
                    ]
                }
            }
        }
    ]
}
```

Este próximo exemplo permite que apenas a conta remetente dos dados de log (111111111111) envie logs à conta destinatária dos dados de 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. Anexe a política criada na etapa anterior ao destino.

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

Essa política de acesso permite que os usuários na AWS Conta com ID 111111111111 liguem PutSubscriptionFilter contra o destino com o ARN arn:aws:logs ::999999999999:destination:testDestination. *region* Qualquer tentativa de outro usuário de ligar PutSubscriptionFilter contra esse destino será rejeitada.

Para validar os privilégios de um usuário com base em uma política de acesso, consulte Usar o validador de políticas no Manual do usuário do IAM.

Quando terminar, se estiver usando AWS Organizations suas permissões entre contas, siga as etapas emEtapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.. Se você estiver concedendo permissões diretamente para a outra conta em vez de usar Organizations, você pode pular essa etapa e prosseguir para Etapa 4: criar um filtro de assinatura.

Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.

Na seção anterior, se você criou o destino usando uma política de acesso que concede permissões à organização em que está a conta 111111111111, em vez de conceder permissões diretamente para a conta 111111111111, siga as etapas nesta seção. Caso contrário, pule para Etapa 4: criar um filtro de assinatura.

As etapas desta seção criam uma função do IAM, que CloudWatch pode assumir e validar se a conta do remetente tem permissão para criar um filtro de assinatura em relação ao destino do destinatário.

Siga as etapas nesta seção na conta do remetente. A função deve existir na conta do remetente e você especifica o ARN dessa função no filtro de assinatura. Neste exemplo, a conta de remetente é `111111111111`.

Para criar a função do IAM necessária para assinaturas de registros entre contas usando AWS Organizations

1. Crie a seguinte política de confiança em um arquivo `/TrustPolicyForCWLSubscriptionFilter.json`. Use um editor de texto para criar esse arquivo de política; não use o console do IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. Crie uma função do IAM que use essa política. Anote o valor `Arn` retornado pelo comando, pois você precisará dele posteriormente nesse procedimento. Neste exemplo, usamos `CWLtoSubscriptionFilterRole` como o nome da função que estamos criando.

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. Crie uma política de permissões para definir as ações que o CloudWatch Logs pode realizar em sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado `~/PermissionsForCWLSubscriptionFilter.json`.

```
{
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

b. Insira o seguinte comando para associar a política de permissões que você acabou de criar à função criada na etapa 2.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

Quando terminar, você pode prosseguir para Etapa 4: criar um filtro de assinatura.

Etapa 3: permissões Add/validate do IAM para o destino entre contas

De acordo com a lógica de avaliação de políticas AWS entre contas, para acessar qualquer recurso entre contas (como um stream do Kinesis ou Firehose usado como destino para um filtro de assinatura), você deve ter uma política baseada em identidade na conta de envio que forneça acesso explícito ao recurso de destino entre contas. Para obter mais informações sobre a lógica de avaliação de políticas, consulte lógica Cross-account de avaliação de políticas.

Você pode associar a política baseada em identidade ao perfil do IAM ou ao usuário do IAM que você está usando para criar o filtro de assinatura. Essa política deve estar presente na conta de envio. Se você estiver usando a função de administrador para criar o filtro de assinatura, poderá ignorar esta etapa e prosseguir para Etapa 4: criar um filtro de assinatura.

Para adicionar ou validar as permissões do IAM necessárias entre contas

1. Digite o comando a seguir para verificar qual função do IAM ou usuário do IAM está sendo usado para executar comandos de AWS registros.

   ```
   aws sts get-caller-identity
   ```

   Esse comando retorna uma saída semelhante à seguinte:

   ```
   {
   ```

```
"UserId": "User ID",
"Account": "sending account id",
"Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

Anote o valor representado por *RoleName* ou*UserName*.

2. Faça login Console de gerenciamento da AWS na conta de envio e pesquise as políticas anexadas com a função do IAM ou o usuário do IAM retornados na saída do comando que você inseriu na etapa 1.
3. Verifique se as políticas vinculadas a esse perfil ou usuário fornecem permissões explícitas para chamar `logs:PutSubscriptionFilter` no recurso de destino entre contas.

   A política a seguir fornece permissões para criar um filtro de assinatura em qualquer recurso de destino somente em uma única AWS conta, conta999999999999:

   JSON

   ```
   {
        "Version":"2012-10-17",
        "Statement": [
           {
               "Sid": "AllowSubscriptionFiltersOnAccountResources",
               "Effect": "Allow",
               "Action": "logs:PutSubscriptionFilter",
               "Resource": [
                   "arn:aws:logs:*:*:log-group:*",
                   "arn:aws:logs:*:123456789012:destination:*"
               ]
           }
       ]
   }
   ```

   A política a seguir fornece permissões para criar um filtro de assinatura somente em um recurso de destino específico chamado `sampleDestination` em uma única AWS conta, conta123456789012:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSubscriptionFiltersonAccountResource",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:sampleDestination"
            ]
        }
    ]
}
```

Etapa 4: criar um filtro de assinatura

Depois de criar um destino, a conta destinatária dos dados de log pode compartilhar o ARN do destino (arn:aws:logs:us-east-1:999999999999:destination:testDestination) com outras contas da AWS para que elas possam enviar seus eventos de log para o mesmo destino. Esses outros usuários de contas de envio criam um filtro de assinatura em seus respectivos grupos de log para esse destino. O filtro de assinatura filtrar inicia imediatamente o fluxo de dados de log em tempo real a partir do grupo de logs escolhido para o destino especificado.

> **Note**
>
> Se você estiver concedendo permissões para o filtro de assinatura a uma organização inteira, precisará usar o ARN do perfil do IAM que criou em Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM..

No exemplo a seguir, um filtro de assinatura é criado em uma conta de envio. O filtro é associado a um grupo de registros contendo AWS CloudTrail eventos para que cada atividade registrada feita pelas AWS credenciais “Raiz” seja entregue ao destino que você criou anteriormente. Esse destino encapsula um fluxo chamado ""RecipientStream.

O restante das etapas nas seções a seguir pressupõe que você tenha seguido as instruções em Enviando CloudTrail eventos para CloudWatch registros no Guia do AWS CloudTrail usuário e criado um grupo de registros que contém seus CloudTrail eventos. Essas etapas pressupõem que o nome desse grupo de logs é `CloudTrail/logs`.

Ao inserir o comando a seguir, certifique-se de estar conectado como usuário do IAM ou usando o perfil do IAM para o qual você adicionou a política, em Etapa 3: permissões Add/validate do IAM para o destino entre contas.

```
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"
```

O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o destino pode apontar para um AWS recurso, como um stream do Amazon Kinesis Data Streams, localizado em uma região diferente.

Validação do fluxo de eventos de logs

Depois de criar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro de entrada que correspondem ao padrão do filtro para o fluxo que está encapsulado no fluxo de destino chamado "". RecipientStream O proprietário do destino pode verificar se isso está acontecendo usando o comando aws kinesis get-shard-iterator para pegar um fragmento do Amazon Kinesis Data Streams e usando o comando aws kinesis get-records para buscar alguns registros do Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator \
      --stream-name RecipientStream \
      --shard-id shardId-000000000000 \
      --shard-iterator-type TRIM_HORIZON

{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
}
```

```
aws kinesis get-records \
      --limit 10 \
      --shard-iterator
      "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

> **Note**
>
> Talvez seja necessário executar novamente o comando get-records algumas vezes antes que o Amazon Kinesis Data Streams comece a retornar dados.

Você deve ver uma resposta com uma matriz de registros do Amazon Kinesis Data Streams. O atributo data no registro do Amazon Kinesis Data Streams é compactado no formato gzip e depois codificado em base64. Você pode examinar os dados brutos na linha de comando usando o seguinte comando Unix:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "owner": "111111111111",
    "logGroup": "CloudTrail/logs",
    "logStream": "111111111111_CloudTrail/logs_us-east-1",
    "subscriptionFilters": [
        "RecipientStream"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
```

```
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        }
    ]
}
```

Os principais elementos nesta estrutura de dados são os seguintes:

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

messageType

As mensagens de dados usam o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros do Amazon Kinesis Data Streams do tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade ID é um identificador exclusivo de cada evento de log.

Modificar a associação de destino no runtime

Pode haver situações em que você precise adicionar ou remover a associação de alguns usuários de um destino que você possua. Você pode usar o comando `put-destination-policy` em seu

destino com uma nova política de acesso. No exemplo a seguir, uma conta 111111111111 recém-adicionada é impedida de enviar mais dados de log e a conta 222222222222 é ativada.

1. Busque a política atualmente associada ao destino TestDestination e anote: 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. Atualize a política para refletir que a conta 111111111111 foi interrompida e a conta 222222222222 está habilitada. Coloque essa política no NewAccessPolicy.json arquivo ~/:

   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. Chame PutDestinationPolicy para associar a política definida no arquivo NewAccessPolicy.json com o destino:

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

Por fim, isso desativará os eventos de log do ID da conta 111111111111. Os eventos de log da ID da conta 222222222222 começarão a fluir para o destino assim que o proprietário da conta 222222222222 criar um filtro de assinatura.

Atualizar uma assinatura existente entre contas

Se você atualmente tiver uma assinatura de logs entre contas em que a conta de destino concede permissões apenas para contas de remetente específicas e quiser atualizar essa assinatura para que a conta de destino conceda acesso a todas as contas em uma organização, siga as etapas desta seção.

Tópicos

- Etapa 1: atualizar os filtros de assinatura
- Etapa 2: atualizar a política de acesso ao destino existente

Etapa 1: atualizar os filtros de assinatura

> **Note**
>
> Essa etapa é necessária apenas para assinaturas entre contas para logs criados pelos serviços listados em Ativar o registro de AWS serviços. Se você não estiver trabalhando com logs criados por um desses grupos de log, pule para Etapa 2: atualizar a política de acesso ao destino existente.

Em determinados casos, você deve atualizar os filtros de assinatura em todas as contas de remetente que estão enviando logs para a conta de destino. A atualização adiciona uma função do

IAM, que CloudWatch pode presumir e validar que a conta do remetente tem permissão para enviar registros para a conta do destinatário.

Siga as etapas desta seção para cada conta de remetente que você deseja atualizar para usar o ID da organização para as permissões de assinatura entre contas.

Nos exemplos desta seção, duas contas, 111111111111 e 222222222222 já têm filtros de assinatura criados para enviar logs para a conta 999999999999. Os valores de filtro de assinatura existentes são os seguintes:

```
## 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"
```

Se você precisar encontrar os valores do parâmetro do filtro de assinatura atual, insira o seguinte comando.

```
aws logs describe-subscription-filters
    \ --log-group-name "my-log-group-name"
```

Atualizar um filtro de assinatura para começar a usar IDs da organização para permissões de log entre contas

1. Crie a seguinte política de confiança em um arquivo ~/TrustPolicyForCWL.json. Use um editor de texto para criar esse arquivo de política; não use o console do IAM.

   ```
   {
     "Statement": {
       "Effect": "Allow",
       "Principal": { "Service": "logs.amazonaws.com" },
       "Action": "sts:AssumeRole"
     }
   }
   ```

2. Crie uma função do IAM que use essa política. Observe o valor Arn do valor Arn que for retornado pelo comando, pois você precisará dele posteriormente nesse procedimento. Neste exemplo, usamos CWLtoSubscriptionFilterRole como o nome da função que estamos criando.

```
aws iam create-role
    \ --role-name CWLtoSubscriptionFilterRole
    \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. Crie uma política de permissões para definir as ações que o CloudWatch Logs pode realizar em sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado `/PermissionsForCWLSubscriptionFilter.json`.

   ```
   {
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "logs:PutLogEvents",
               "Resource": "arn:aws:logs:region:111111111111:log-
   group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
           }
       ]
   }
   ```

   b. Insira o seguinte comando para associar a política de permissões que você acabou de criar à função criada na etapa 2.

   ```
   aws iam put-role-policy
       --role-name CWLtoSubscriptionFilterRole
       --policy-name Permissions-Policy-For-CWL-Subscription-filter
       --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
   ```

4. Insira o comando a seguir para atualizar o filtro de assinatura.

```
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"
```

Etapa 2: atualizar a política de acesso ao destino existente

Depois de atualizar os filtros de assinatura em todas as contas de remetente, você poderá atualizar a política de acesso de destino na conta do destinatário.

Nos exemplos a seguir, a conta do destinatário é `999999999999` e o destino é chamado de `testDestination`.

A atualização habilita todas as contas que fazem parte da organização com ID de `o-1234567890` a enviar logs à conta destinatária. Somente as contas que tiverem filtros de assinatura criados enviarão logs para a conta do destinatário.

Atualizar a política de acesso de destino na conta do destinatário para começar a usar um ID de organização para permissões

1. Na conta destinatária, use um editor de texto para criar um arquivo `~/AccessPolicy.json` com o seguinte conteúdo.

   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. Insira o seguinte comando para anexar a política que você acabou de criar ao destino existente. Para atualizar um destino para usar uma política de acesso com uma ID da organização em vez de uma política de acesso que lista IDs de AWS contas específicas, inclua o `force` parâmetro.

> **Warning**
>
> Se você estiver trabalhando com registros enviados por um AWS serviço listado emAtivar o registro de AWS serviços, antes de realizar essa etapa, você deve primeiro atualizar os filtros de assinatura em todas as contas do remetente, conforme explicado emEtapa 1: atualizar os filtros de assinatura.

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
    \ --force
```

Cross-account Compartilhamento de dados de registro entre regiões usando Firehose

Para compartilhar dados de log entre contas, você precisa estabelecer um remetente e um destinatário dos dados de log:

- Remetente de dados de registro — obtém as informações de destino do destinatário e informa ao CloudWatch Logs que ele está pronto para enviar seus eventos de registro para o destino especificado. Nos procedimentos do restante desta seção, o remetente dos dados de registro é mostrado com um número de AWS conta fictício de 111111111111.
- Destinatário dos dados de log — configura um destino que encapsula um stream do Amazon Kinesis Data Streams e informa aos CloudWatch Logs que o destinatário deseja receber dados de log. O destinatário, então, compartilha as informações sobre esse destino com o remetente. Nos procedimentos do restante desta seção, o destinatário dos dados de registro é mostrado com um número de AWS conta fictício de 222222222222.

O exemplo desta seção usa um fluxo de entrega do Firehose com armazenamento do Amazon S3. Também é possível configurar fluxos de entrega do Firehose com configurações diferentes. Para obter mais informações, consulte Criar um fluxo de entrega do Firehose.

> **Note**
>
> O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o AWS recurso para o qual o destino aponta pode estar localizado em uma região diferente.

> **Note**
>
> O filtro de assinatura do Firehose para um fluxo de entrega na mesma conta e entre regiões é compatível.

Tópicos

- Etapa 1: criar um fluxo de entrega do Firehose
- Etapa 2: Criar um destino
- Etapa 3: permissões Add/validate do IAM para o destino entre contas
- Etapa 4: criar um filtro de assinatura
- Validação do fluxo de eventos de log
- Modificação da associação de destino no runtime

Etapa 1: criar um fluxo de entrega do Firehose

> **Important**
>
> Antes de concluir as etapas a seguir, você deve usar uma política de acesso para que o Firehose possa acessar o bucket do Amazon S3. Para obter mais informações, consulte Controlling Access no Amazon Data Firehose Developer Guide.
> Todas as etapas desta seção (Etapa 1) devem ser realizadas na conta destinatária dos dados do log.
> Leste dos EUA (N. da Virgínia) é a região usada nos exemplos de comando a seguir. Substitua essa região pela região correta da implantação.

Para criar um fluxo de entrega do Firehose para ser usado como destino

1. Criar um bucket do Amazon S3:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-1
```

2. Crie o perfil do IAM que concede ao Firehose permissão para colocar dados no bucket.

   a. Primeiro, use um editor de texto para criar uma política de confiança em um arquivo `~/TrustPolicyForFirehose.json`.

   ```
   { "Statement": { "Effect": "Allow", "Principal": { "Service":
    "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
    { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
   ```

   b. Crie a função do IAM, especificando o arquivo de política de confiança que você acabou de criar.

   ```
   aws iam create-role \
       --role-name FirehosetoS3Role \
       --assume-role-policy-document file://~/TrustPolicyForFirehose.json
   ```

   c. A saída deste comando será parecida com o exemplo a seguir. Anote os valores do nome da função e do ARN da função.

   ```
   {
       "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. Crie uma política de permissões para definir as ações que o Firehose pode realizar na sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado `~/PermissionsForFirehose.json`. Dependendo do caso de uso, pode ser necessário adicionar mais permissões a esse arquivo.

```
{
    "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. Insira o seguinte comando para associar a política de permissões que você acabou de criar ao perfil do IAM.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. Insira o comando a seguir para criar o fluxo de entrega do Firehose. Substitua *my-role-arn* e *amzn-s3-demo-bucket2-arn* com os valores corretos para sua implantação.

```
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"}'
```

A saída deve ser semelhante à seguinte:

```
{
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

Etapa 2: Criar um destino

> **Important**
>
> As etapas deste procedimento devem ser processadas na conta destinatária dos dados do log.

Quando o destino é criado, o CloudWatch Logs envia uma mensagem de teste para o destino em nome da conta do destinatário. Quando o filtro de assinatura estiver ativo posteriormente, o CloudWatch Logs envia eventos de registro para o destino em nome da conta de origem.

Para criar um destino

1. Aguarde até o fluxo do Firehose que você criou em Etapa 1: criar um fluxo de entrega do Firehose ficar ativo. Você pode usar o comando a seguir para verificar a StreamDescription.StreamStatus propriedade.

   ```
   aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
   ```

   Além disso, anote o DeliveryStreamDescription.DeliveryStreamARN valor, pois você precisará usá-lo em uma etapa posterior. Exemplo de saída desse comando:

   ```
   {
       "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
    }
}
```

Pode levar um ou dois minutos para seu fluxo de entrega ser exibido com estado ativo.

2. Quando o stream de entrega estiver ativo, crie a função IAM que concederá aos CloudWatch Logs a permissão para colocar dados em seu stream do Firehose. Primeiro, você precisará criar uma política de confiança em um arquivo ~/ TrustPolicyForCWL.json. Use um editor de texto para criar esta política. Para obter mais informações sobre endpoints do CloudWatch Logs, consulte Endpoints e cotas do Amazon CloudWatch Logs.

   Esta política inclui uma chave de contexto de condição global `aws:SourceArn` que especifica o `sourceAccountId` para evitar o problema de segurança confused deputy. Se você ainda não souber o ID da conta de origem na primeira chamada, recomendamos que você coloque o ARN de destino no campo ARN de origem. Nas chamadas subsequentes, você deve definir o ARN de origem como o ARN de origem real que você coletou da primeira chamada. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.region.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "ArnLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
     }
}
```

3. Use o comando aws iam create-role para criar a função do IAM, especificando o arquivo de política de confiança que você acabou de criar.

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

Este é um exemplo de saída. Anote o valor `Role.Arn` retornado, pois será necessário utilizá-lo em uma etapa posterior.

```
{
    "Role": {
        "Path": "/",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "RoleId": "AROAR3BXASEKYJYWF243H",
        "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": {
                    "ArnLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. Crie uma política de permissões para definir quais ações CloudWatch os registros podem realizar em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/: PermissionsForCWL.json

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
```

```
}
```

5. Associe a política de permissões com a função inserindo o seguinte comando:

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Depois que o stream de entrega do Firehose estiver no estado ativo e você tiver criado a função IAM, você poderá criar o destino do CloudWatch Logs.

   a. Esta etapa não associará uma política de acesso ao seu destino e só é a primeira etapa das duas concluirá uma criação de destino. Anote o ARN do novo destino que for retornado na carga útil, pois você usará isso como `destination.arn` em uma etapa posterior.

```
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. Depois que a etapa anterior for concluída, na conta destinatária dos dados de log (222222222222), associe uma política de acesso ao destino.

      Essa política permite que a conta remetente dos dados de log (111111111111 conta) acesse o destino apenas na conta destinatária dos dados de log (222222222222). Você pode usar um editor de texto para colocar essa política no AccessPolicy.json arquivo ~/:

      JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:us-
east-1:222222222222:destination:testFirehoseDestination"
    }
  ]
}
```

c. Isso cria uma política que define quem tem acesso de gravação ao destino. Essa política deve especificar a PutSubscriptionFilter ação logs: para acessar o destino. Cross-account os usuários usarão a PutSubscriptionFilter ação para enviar eventos de log para o destino:

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
    --access-policy file://~/AccessPolicy.json
```

Etapa 3: permissões Add/validate do IAM para o destino entre contas

De acordo com a lógica de avaliação de políticas AWS entre contas, para acessar qualquer recurso entre contas (como um stream do Kinesis ou Firehose usado como destino para um filtro de assinatura), você deve ter uma política baseada em identidade na conta de envio que forneça acesso explícito ao recurso de destino entre contas. Para obter mais informações sobre a lógica de avaliação de políticas, consulte lógica Cross-account de avaliação de políticas.

Você pode associar a política baseada em identidade ao perfil do IAM ou ao usuário do IAM que você está usando para criar o filtro de assinatura. Essa política deve estar presente na conta de envio. Se você estiver usando a função de administrador para criar o filtro de assinatura, poderá ignorar esta etapa e prosseguir para Etapa 4: criar um filtro de assinatura.

Para adicionar ou validar as permissões do IAM necessárias entre contas

1. Digite o comando a seguir para verificar qual função do IAM ou usuário do IAM está sendo usado para executar comandos de AWS registros.

```
aws sts get-caller-identity
```

Esse comando retorna uma saída semelhante à seguinte:

```
{
"UserId": "User ID",
"Account": "sending account id",
"Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

Anote o valor representado por *RoleName* ou*UserName*.

2. Faça login Console de gerenciamento da AWS na conta de envio e pesquise as políticas anexadas com a função do IAM ou o usuário do IAM retornados na saída do comando que você inseriu na etapa 1.
3. Verifique se as políticas vinculadas a esse perfil ou usuário fornecem permissões explícitas para chamar `logs:PutSubscriptionFilter` no recurso de destino entre contas.

   A política a seguir fornece permissões para criar um filtro de assinatura em qualquer recurso de destino somente em uma única AWS conta, conta999999999999:

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid":
 "AllowSubscriptionFiltersOnAnyResourceInOneSpecificAccount",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:*"
            ]
        }
    ]
}
```

A política a seguir fornece permissões para criar um filtro de assinatura somente em um recurso de destino específico chamado `sampleDestination` em uma única AWS conta, conta`123456789012`:

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"
            ]
        }
    ]
}
```

Etapa 4: criar um filtro de assinatura

Alterne para a conta de envio, que é 111111111111 neste exemplo. Agora, você criará o filtro de assinatura na conta de envio. Neste exemplo, o filtro é associado a um grupo de registros contendo AWS CloudTrail eventos para que cada atividade registrada feita pelas AWS credenciais “Raiz” seja entregue ao destino que você criou anteriormente. Para obter mais informações sobre como enviar AWS CloudTrail eventos para CloudWatch registros, consulte Enviando CloudTrail eventos para CloudWatch registros no Guia do AWS CloudTrail usuário.

Ao inserir o comando a seguir, certifique-se de estar conectado como usuário do IAM ou usando o perfil do IAM para o qual você adicionou a política, em Etapa 3: permissões Add/validate do IAM para o destino entre contas.

```
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"
```

O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o destino pode apontar para um AWS recurso, como um stream Firehose, localizado em uma região diferente.

Validação do fluxo de eventos de log

Depois de criar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro recebidos que correspondem ao padrão do filtro para o stream de entrega do Firehose. Os dados começam a aparecer no bucket do Amazon S3 com base no intervalo de buffer de tempo definido no fluxo de entrega do Firehose. Quando tiver passado tempo suficiente, você poderá conferir seus dados verificando o bucket do Amazon S3. Para verificar o bucket, insira este comando:

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

A saída desse comando será semelhante a:

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

Em seguida, você pode recuperar um objeto específico do bucket digitando o comando a seguir. Substitua o valor de `key` pelo valor encontrado no comando anterior.

```
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
```

Os dados no objeto do Amazon S3 são compactados com o formato gzip. É possível examinar os dados brutos na linha de comando usando os seguintes comandos:

Linux

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

Modificação da associação de destino no runtime

Pode haver situações em que você precise adicionar ou remover remetentes de log de um destino que você possua. Você pode usar a PutDestinationPolicy ação em seu destino com a nova política de acesso. No exemplo a seguir, uma conta 111111111111 recém-adicionada é impedida de enviar mais dados de log e a conta 333333333333 é habilitada.

1. Busque a política atualmente associada ao destino TestDestination e anote: 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. Atualize a política para refletir que a conta 111111111111 foi interrompida e a conta 333333333333 está habilitada. Coloque essa política no NewAccessPolicy.json arquivo ~/:

   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. Use o comando a seguir para associar a política definida no NewAccessPolicy.json arquivo ao destino:

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \

    --access-policy file://~/NewAccessPolicy.json
```

   Por fim, isso desativa os eventos de log do ID da conta 111111111111. Os eventos de log da ID da conta 333333333333 começarão a fluir para o destino assim que o proprietário da conta 333333333333 criar um filtro de assinatura.

Cross-account Assinaturas em nível de conta entre regiões usando o Amazon Kinesis Data Streams

Ao criar uma assinatura entre contas, você pode especificar uma única conta ou organização para ser o remetente. Se você especificar uma organização, esse procedimento permitirá que todas as contas da organização enviem logs para a conta do receptor.

Para compartilhar dados de log entre contas, você precisa estabelecer um remetente e um destinatário dos dados de log:

- Remetente de dados de registro — obtém as informações de destino do destinatário e informa ao CloudWatch Logs que está pronto para enviar seus eventos de registro para o destino especificado. Nos procedimentos do restante desta seção, o remetente dos dados de registro é mostrado com um número de AWS conta fictício de 111111111111.

  Se você quiser que várias contas em uma organização enviem logs para uma conta de destinatário, você pode criar uma política que conceda a todas as contas da organização a permissão para enviar logs para a conta do destinatário. Você ainda precisa configurar filtros de assinatura diferentes para cada conta de remetente.
- Destinatário dos dados de log — configura um destino que encapsula um stream do Amazon Kinesis Data Streams e informa aos CloudWatch Logs que o destinatário deseja receber dados de log. O destinatário, então, compartilha as informações sobre esse destino com o remetente. Nos procedimentos do restante desta seção, o destinatário dos dados de log é mostrado com um número de AWS conta fictício de 999999999999.

Para começar a receber eventos de registro de usuários de várias contas, o destinatário dos dados de registro primeiro cria um destino de CloudWatch registros. Cada destino consiste nos seguintes elementos-chave:

Nome do destino

O nome do destino que você deseja criar.

ARN de destino

O nome de recurso da Amazon (ARN) do AWS recurso que você deseja usar como destino do feed de assinatura.

ARN de função

Uma função AWS Identity and Access Management (IAM) que concede aos CloudWatch Logs as permissões necessárias para colocar dados no stream escolhido.

Política de acesso

Documento de políticas do IAM (no formato JSON, gravado usando a gramática de políticas do IAM) que controla o conjunto de usuários que têm permissão para gravar em seu destino.

> **Note**
>
> O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o AWS recurso para o qual o destino aponta pode estar localizado em uma região diferente. Nos exemplos das seções a seguir, todos os Region-specific recursos são criados no Leste dos EUA (Norte da Virgínia).

Tópicos

- Como configurar uma nova assinatura entre contas
- Atualizar uma assinatura existente entre contas

Como configurar uma nova assinatura entre contas

Siga as etapas nestas seções para configurar uma nova assinatura de log entre contas.

Tópicos

- Etapa 1: criar um destino
- Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.
- Etapa 3: criar uma política de filtro de assinatura ao nível da conta
- Validação do fluxo de eventos de logs
- Modificar a associação de destino no runtime

Etapa 1: criar um destino

> **Important**
>
> As etapas deste procedimento devem ser processadas na conta destinatária dos dados do log.

Neste exemplo, a conta do destinatário dos dados de registro tem um ID de conta de 999999999999, enquanto o ID da AWS conta do remetente dos dados de registro é 111111111111. AWS

Este exemplo cria um destino usando um stream do Amazon Kinesis Data Streams chamado RecipientStream e uma função que permite que CloudWatch os Logs gravem dados nele.

Quando o destino é criado, o CloudWatch Logs envia uma mensagem de teste para o destino em nome da conta do destinatário. Quando o filtro de assinatura estiver ativo posteriormente, o CloudWatch Logs envia eventos de registro para o destino em nome da conta de origem.

Para criar um destino

1. Na conta do destinatário, crie um stream de destino no Amazon Kinesis Data Streams. Em um prompt de comando, digite:

```
aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
```

2. Aguarde até que o fluxo do fique ativo. Você pode usar o comando aws kinesis describe-stream para verificar a propriedade. StreamDescription.StreamStatus Além disso, anote o StreamDescription.StreamARN valor porque você o passará para o CloudWatch Logs posteriormente:

```
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": "340282366920938463463374607431768EXAMPLE",
          "StartingHashKey": "0"
        },
        "SequenceNumberRange": {
          "StartingSequenceNumber":
 "49551135218688818456679503831981458784591352702 18EXAMPLE"
        }
      }
    ]
  }
}
```

Pode levar um ou dois minutos para o seu stream aparecer no estado ativo.

3. Crie a função do IAM que concede aos CloudWatch Logs a permissão para colocar dados em seu stream. Primeiro, você precisará criar uma política de confiança em um arquivo ~/

TrustPolicyForCWL.json. Use um editor de texto para criar esse arquivo de política, não use o console do IAM.

Esta política inclui uma chave de contexto de condição global `aws:SourceArn` que especifica o `sourceAccountId` para evitar o problema de segurança confused deputy. Se você ainda não souber o ID da conta de origem na primeira chamada, recomendamos que você coloque o ARN de destino no campo ARN de origem. Nas chamadas subsequentes, você deve definir o ARN de origem como o ARN de origem real que você coletou da primeira chamada. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Condition": {
            "ArnLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
}
```

4. Use o comando aws iam create-role para criar a função do IAM especificando o arquivo de política de confiança. Anote o Role.Arn valor retornado, pois ele também será passado para o CloudWatch Logs posteriormente:

```
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": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2023-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. Crie uma política de permissões para definir quais ações CloudWatch os registros podem realizar em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/: PermissionsForCWL.json

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. Associe a política de permissões à função usando o comando 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. Depois que o stream estiver no estado ativo e você tiver criado a função do IAM, você poderá criar o destino de CloudWatch registros.

   a. Esta etapa não associará uma política de acesso ao seu destino e é apenas a primeira etapa de duas que concluirá uma criação de destino. Anote o DestinationArn que for retornado na carga útil:

```
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. Depois que a etapa 7a for concluída, na conta destinatária dos dados de log, associe uma política de acesso ao destino. Essa política deve especificar a PutSubscriptionFilter ação logs: e conceder permissão à conta do remetente para acessar o destino.

      A política concede permissão à AWS conta que envia registros. Você pode especificar apenas essa conta na política ou, se a conta de remetente for parte de uma organização, a política poderá especificar o ID da organização. Dessa forma, você pode criar apenas uma política para permitir que várias contas em uma organização enviem logs para essa conta de destino.

      Use um editor de texto para criar um arquivo chamado `~/AccessPolicy.json` com uma das seguintes declarações de política.

      Este primeiro exemplo de política permite que todas as contas na organização que tenham um ID de `o-1234567890` enviem logs à conta do destinatário.

      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"
                    ]
                }
            }
        }
    ]
}
```

Este próximo exemplo permite que apenas a conta remetente dos dados de log (111111111111) envie logs à conta destinatária dos dados de 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. Anexe a política criada na etapa anterior ao destino.

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

Essa política de acesso permite que os usuários na AWS Conta com ID 111111111111 liguem PutSubscriptionFilter contra o destino com o ARN arn:aws:logs ::999999999999:destination:testDestination. *region* Qualquer tentativa de outro usuário de ligar PutSubscriptionFilter contra esse destino será rejeitada.

Para validar os privilégios de um usuário com base em uma política de acesso, consulte Usar o validador de políticas no Manual do usuário do IAM.

Quando terminar, se estiver usando AWS Organizations suas permissões entre contas, siga as etapas emEtapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.. Se você estiver concedendo permissões diretamente para a outra conta em vez de usar Organizations, você pode pular essa etapa e prosseguir para Etapa 3: criar uma política de filtro de assinatura ao nível da conta.

Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM.

Na seção anterior, se você criou o destino usando uma política de acesso que concede permissões à organização em que está a conta 111111111111, em vez de conceder permissões diretamente para a conta 111111111111, siga as etapas nesta seção. Caso contrário, pule para Etapa 3: criar uma política de filtro de assinatura ao nível da conta.

As etapas desta seção criam uma função do IAM, que CloudWatch pode assumir e validar se a conta do remetente tem permissão para criar um filtro de assinatura em relação ao destino do destinatário.

Siga as etapas nesta seção na conta do remetente. A função deve existir na conta do remetente e você especifica o ARN dessa função no filtro de assinatura. Neste exemplo, a conta de remetente é 111111111111.

Para criar a função do IAM necessária para assinaturas de registros entre contas usando AWS Organizations

1. Crie a seguinte política de confiança em um arquivo `/TrustPolicyForCWLSubscriptionFilter.json`. Use um editor de texto para criar esse arquivo de política; não use o console do IAM.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. Crie uma função do IAM que use essa política. Anote o valor `Arn` retornado pelo comando, pois você precisará dele posteriormente nesse procedimento. Neste exemplo, usamos `CWLtoSubscriptionFilterRole` como o nome da função que estamos criando.

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. Crie uma política de permissões para definir as ações que o CloudWatch Logs pode realizar em sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado `~/PermissionsForCWLSubscriptionFilter.json`.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

b. Insira o seguinte comando para associar a política de permissões que você acabou de criar à função criada na etapa 2.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

Quando terminar, você pode prosseguir para Etapa 3: criar uma política de filtro de assinatura ao nível da conta.

Etapa 3: criar uma política de filtro de assinatura ao nível da conta

Depois de criar um destino, a conta destinatária dos dados de log pode compartilhar o ARN do destino (arn:aws:logs:us-east-1:999999999999:destination:testDestination) com outras contas da AWS para que elas possam enviar seus eventos de log para o mesmo destino. Esses outros usuários de contas de envio criam um filtro de assinatura em seus respectivos grupos de log para esse destino. O filtro de assinatura filtrar inicia imediatamente o fluxo de dados de log em tempo real a partir do grupo de logs escolhido para o destino especificado.

> **Note**
>
> Se você estiver concedendo permissões para o filtro de assinatura a uma organização inteira, precisará usar o ARN do perfil do IAM que criou em Etapa 2: (somente se estiver usando uma organização) Crie uma função do IAM..

No exemplo a seguir, uma política de filtro de assinatura ao nível da conta é criada em uma conta de envio. O filtro é associado à conta de envio 111111111111 de modo que todos os eventos de log que corresponderem ao filtro e aos critérios de seleção sejam entregues ao destino que você criou anteriormente. Esse destino encapsula um fluxo chamado ""RecipientStream.

O campo `selection-criteria` é opcional, mas é importante para excluir de um filtro de assinatura os grupos de logs que possam causar uma repetição infinita de logs. Para obter mais informações sobre esse problema e determinar quais grupos de logs devem ser excluídos, consulte Prevenção de repetição de logs. Atualmente, NOT IN é o único operador compatível com `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"
```

Os grupos de logs da conta de envio e o destino devem estar na mesma região da AWS . No entanto, o destino pode apontar para um AWS recurso, como um stream do Amazon Kinesis Data Streams, localizado em uma região diferente.

Validação do fluxo de eventos de logs

Depois de criar a política de filtro de assinatura no nível da conta, o CloudWatch Logs encaminha todos os eventos de registro de entrada que correspondam ao padrão de filtro e aos critérios de seleção para o fluxo que está encapsulado no fluxo de destino chamado "". RecipientStream O proprietário do destino pode verificar se isso está acontecendo usando o comando aws kinesis get-shard-iterator para pegar um fragmento do Amazon Kinesis Data Streams e usando o comando aws kinesis get-records para buscar alguns registros do Amazon Kinesis Data Streams:

```
aws kinesis get-shard-iterator \
      --stream-name RecipientStream \
      --shard-id shardId-000000000000 \
      --shard-iterator-type TRIM_HORIZON

{
    "ShardIterator":
    "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
}

aws kinesis get-records \
      --limit 10 \
      --shard-iterator
      "AAAAAAAAAAFGU/
kLvNggvndHq2UIFOw5PZc6F01s3e3afsSscRM70JSbjIefg2ub07nk1y6CDxYR1UoGHJNP4m4NFUetzfL+wev
```

```
+e2P4djJg4L9wmXKvQYoE+rMUiFq+p4Cn3IgvqOb5dRA0yybNdRcdzvnC35KQANoHzzahKdRGb9v4scv+3vaq+f
+OIK8zM5My8ID+g6rMo7UKWeI4+IWiKEXAMPLE"
```

> **Note**
>
> Talvez seja necessário executar o `get-records` comando novamente algumas vezes antes que o Amazon Kinesis Data Streams comece a retornar dados.

Você deve ver uma resposta com uma matriz de registros do Amazon Kinesis Data Streams. O atributo data no registro do Amazon Kinesis Data Streams é compactado no formato gzip e depois codificado em base64. Você pode examinar os dados brutos na linha de comando usando o seguinte comando Unix:

```
echo -n "<Content of Data>" | base64 -d | zcat
```

Os dados base64 decodificados e descompactados têm o formato JSON com a seguinte estrutura:

```
{
    "owner": "111111111111",
    "logGroup": "CloudTrail/logs",
    "logStream": "111111111111_CloudTrail/logs_us-east-1",
    "subscriptionFilters": [
        "RecipientStream"
    ],
    "messageType": "DATA_MESSAGE",
    "logEvents": [
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        },
        {
            "id": "31953106606966983378809025079804211143289615424 29EXAMPLE",
```

```
            "timestamp": 1432826855000,
            "message": "{\"eventVersion\":\"1.03\",\"userIdentity\":{\"type\":\"Root
\"}"
        }
    ]
}
```

Os principais elementos da estrutura de dados são os seguintes:

messageType

As mensagens de dados usarão o tipo "DATA_MESSAGE". Às vezes, CloudWatch os registros podem emitir registros do Amazon Kinesis Data Streams com o tipo “CONTROL_MESSAGE”, principalmente para verificar se o destino está acessível.

proprietário

O ID da AWS conta dos dados de registro de origem.

logGroup

O nome do grupo de logs dos dados de log de origem.

logStream

O nome do stream de log dos dados de log de origem.

subscriptionFilters

A lista de nomes de filtro de assinatura que corresponderam aos dados de log de origem.

logEvents

Os dados de log reais, representados como um conjunto de registros de eventos de log. A propriedade "id" é um identificador exclusivo de cada evento de log.

policyLevel

O nível em que a política foi aplicada. "ACCOUNT_LEVEL_POLICY" é a `policyLevel` para uma política de filtro de assinatura ao nível da conta.

Modificar a associação de destino no runtime

Pode haver situações em que você precise adicionar ou remover a associação de alguns usuários de um destino que você possua. Você pode usar o comando `put-destination-policy` em seu

destino com uma nova política de acesso. No exemplo a seguir, uma conta 111111111111 recém-adicionada é impedida de enviar mais dados de log e a conta 222222222222 é ativada.

1. Busque a política atualmente associada ao destino TestDestination e anote: 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. Atualize a política para refletir que a conta 111111111111 foi interrompida e a conta 222222222222 está habilitada. Coloque essa política no NewAccessPolicy.json arquivo ~/:

   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. Chame PutDestinationPolicy para associar a política definida no arquivo NewAccessPolicy.json com o destino:

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

Por fim, isso desativará os eventos de log do ID da conta 111111111111. Os eventos de log da ID da conta 222222222222 começarão a fluir para o destino assim que o proprietário da conta 222222222222 criar um filtro de assinatura.

Atualizar uma assinatura existente entre contas

Se você atualmente tiver uma assinatura de logs entre contas em que a conta de destino concede permissões apenas para contas de remetente específicas e quiser atualizar essa assinatura para que a conta de destino conceda acesso a todas as contas em uma organização, siga as etapas desta seção.

Tópicos

- Etapa 1: atualizar os filtros de assinatura
- Etapa 2: atualizar a política de acesso ao destino existente

Etapa 1: atualizar os filtros de assinatura

> **Note**
>
> Essa etapa é necessária apenas para assinaturas entre contas para logs criados pelos serviços listados em Ativar o registro de AWS serviços. Se você não estiver trabalhando com logs criados por um desses grupos de log, pule para Etapa 2: atualizar a política de acesso ao destino existente.

Em determinados casos, você deve atualizar os filtros de assinatura em todas as contas de remetente que estão enviando logs para a conta de destino. A atualização adiciona uma função do IAM, que CloudWatch pode presumir e validar que a conta do remetente tem permissão para enviar registros para a conta do destinatário.

Siga as etapas desta seção para cada conta de remetente que você deseja atualizar para usar o ID da organização para as permissões de assinatura entre contas.

Nos exemplos desta seção, duas contas, 111111111111 e 222222222222 já têm filtros de assinatura criados para enviar logs para a conta 999999999999. Os valores de filtro de assinatura existentes são os seguintes:

```
## Existing Subscription Filter parameter values
{
    "DestinationArn": "arn:aws:logs:region:999999999999:destination:testDestination",
    "FilterPattern": "{$.userIdentity.type = Root}",
    "Distribution": "Random"
}
```

Se você precisar encontrar os valores do parâmetro do filtro de assinatura atual, insira o seguinte comando.

```
aws logs describe-account-policies \
--policy-type "SUBSCRIPTION_FILTER_POLICY" \
--policy-name "CrossAccountStreamsExamplePolicy"
```

Atualizar um filtro de assinatura para começar a usar IDs da organização para permissões de log entre contas

1. Crie a seguinte política de confiança em um arquivo ~/TrustPolicyForCWL.json. Use um editor de texto para criar esse arquivo de política; não use o console do IAM.

   ```
   {
     "Statement": {
       "Effect": "Allow",
       "Principal": { "Service": "logs.amazonaws.com" },
       "Action": "sts:AssumeRole"
     }
   }
   ```

2. Crie uma função do IAM que use essa política. Observe o valor `Arn` do valor `Arn` que for retornado pelo comando, pois você precisará dele posteriormente nesse procedimento. Neste exemplo, usamos `CWLtoSubscriptionFilterRole` como o nome da função que estamos criando.

```
aws iam create-role
    \ --role-name CWLtoSubscriptionFilterRole
    \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. Crie uma política de permissões para definir as ações que o CloudWatch Logs pode realizar em sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado `/PermissionsForCWLSubscriptionFilter.json`.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. Insira o seguinte comando para associar a política de permissões que você acabou de criar à função criada na etapa 2.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. Insira o comando a seguir para atualizar a política de filtro de assinatura.

```
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"
```

Etapa 2: atualizar a política de acesso ao destino existente

Depois de atualizar os filtros de assinatura em todas as contas de remetente, você poderá atualizar a política de acesso de destino na conta do destinatário.

Nos exemplos a seguir, a conta do destinatário é `999999999999` e o destino é chamado de `testDestination`.

A atualização habilita todas as contas que fazem parte da organização com ID de `o-1234567890` a enviar logs à conta destinatária. Somente as contas que tiverem filtros de assinatura criados enviarão logs para a conta do destinatário.

Atualizar a política de acesso de destino na conta do destinatário para começar a usar um ID de organização para permissões

1. Na conta destinatária, use um editor de texto para criar um arquivo `~/AccessPolicy.json` com o seguinte conteúdo.

   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. Insira o seguinte comando para anexar a política que você acabou de criar ao destino existente. Para atualizar um destino para usar uma política de acesso com uma ID da organização em vez de uma política de acesso que lista IDs de AWS contas específicas, inclua o `force` parâmetro.

> **Warning**
>
> Se você estiver trabalhando com registros enviados por um AWS serviço listado emAtivar o registro de AWS serviços, antes de realizar essa etapa, você deve primeiro atualizar os filtros de assinatura em todas as contas do remetente, conforme explicado emEtapa 1: atualizar os filtros de assinatura.

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
    \ --force
```

Cross-account Assinaturas em nível de conta entre regiões usando Firehose

Para compartilhar dados de log entre contas, você precisa estabelecer um remetente e um destinatário dos dados de log:

- Remetente de dados de registro — obtém as informações de destino do destinatário e informa ao CloudWatch Logs que ele está pronto para enviar seus eventos de registro para o destino especificado. Nos procedimentos do restante desta seção, o remetente dos dados de registro é mostrado com um número de AWS conta fictício de 111111111111.
- Destinatário dos dados de log — configura um destino que encapsula um stream do Amazon Kinesis Data Streams e informa aos CloudWatch Logs que o destinatário deseja receber dados de log. O destinatário, então, compartilha as informações sobre esse destino com o remetente. Nos

procedimentos do restante desta seção, o destinatário dos dados de registro é mostrado com um número de AWS conta fictício de 222222222222.

O exemplo desta seção usa um fluxo de entrega do Firehose com armazenamento do Amazon S3. Também é possível configurar fluxos de entrega do Firehose com configurações diferentes. Para obter mais informações, consulte Criar um fluxo de entrega do Firehose.

> **Note**
>
> O grupo de registros e o destino devem estar na mesma AWS região. No entanto, o AWS recurso para o qual o destino aponta pode estar localizado em uma região diferente.

> **Note**
>
> O filtro de assinatura do Firehose para um fluxo de entrega na mesma conta e entre regiões é compatível.

Tópicos

- Etapa 1: criar um fluxo de entrega do Firehose
- Etapa 2: Criar um destino
- Etapa 3: criar uma política de filtro de assinatura ao nível da conta
- Validação do fluxo de eventos de log
- Modificação da associação de destino no runtime

Etapa 1: criar um fluxo de entrega do Firehose

> **Important**
>
> Antes de concluir as etapas a seguir, você deve usar uma política de acesso para que o Firehose possa acessar o bucket do Amazon S3. Para obter mais informações, consulte Controlling Access no Amazon Data Firehose Developer Guide.
> Todas as etapas desta seção (Etapa 1) devem ser realizadas na conta destinatária dos dados do log.

Leste dos EUA (N. da Virgínia) é a região usada nos exemplos de comando a seguir. Substitua essa região pela região correta da implantação.

Para criar um fluxo de entrega do Firehose para ser usado como destino

1. Criar um bucket do Amazon S3:

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-1
```

2. Crie o perfil do IAM que concede ao Firehose permissão para colocar dados no bucket.

   a. Primeiro, use um editor de texto para criar uma política de confiança em um arquivo ~/TrustPolicyForFirehose.json.

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
 "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
 { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
```

   b. Crie a função do IAM, especificando o arquivo de política de confiança que você acabou de criar.

```
aws iam create-role \
    --role-name FirehosetoS3Role \
    --assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

   c. A saída deste comando será parecida com o exemplo a seguir. Anote os valores do nome da função e do ARN da função.

```
{
    "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. Crie uma política de permissões para definir as ações que o Firehose pode realizar na sua conta.

   a. Primeiro, use um editor de texto para criar a seguinte política de permissões em um arquivo chamado ~/PermissionsForFirehose.json. Dependendo do caso de uso, pode ser necessário adicionar mais permissões a esse arquivo.

```
{
    "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. Insira o seguinte comando para associar a política de permissões que você acabou de criar ao perfil do IAM.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. Insira o comando a seguir para criar o fluxo de entrega do Firehose. Substitua *my-role-arn* e *amzn-s3-demo-bucket2-arn* com os valores corretos para sua implantação.

```
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"}'
```

   A saída deve ser semelhante à seguinte:

```
{
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

Etapa 2: Criar um destino

> **Important**
>
> As etapas deste procedimento devem ser processadas na conta destinatária dos dados do log.

Quando o destino é criado, o CloudWatch Logs envia uma mensagem de teste para o destino em nome da conta do destinatário. Quando o filtro de assinatura estiver ativo posteriormente, o CloudWatch Logs envia eventos de registro para o destino em nome da conta de origem.

Para criar um destino

1. Aguarde até o fluxo do Firehose que você criou em Etapa 1: criar um fluxo de entrega do Firehose ficar ativo. Você pode usar o comando a seguir para verificar a StreamDescription.StreamStatus propriedade.

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
```

   Além disso, anote o DeliveryStreamDescription.DeliveryStreamARN valor, pois você precisará usá-lo em uma etapa posterior. Exemplo de saída desse comando:

```
{
    "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
    }
}
```

Pode levar um ou dois minutos para seu fluxo de entrega ser exibido com estado ativo.

2. Quando o stream de entrega estiver ativo, crie a função IAM que concederá aos CloudWatch Logs a permissão para colocar dados em seu stream do Firehose. Primeiro, você precisará criar uma política de confiança em um arquivo ~/ TrustPolicyForCWL.json. Use um editor de texto para criar esta política. Para obter mais informações sobre endpoints do CloudWatch Logs, consulte Endpoints e cotas do Amazon CloudWatch Logs.

   Esta política inclui uma chave de contexto de condição global `aws:SourceArn` que especifica o `sourceAccountId` para evitar o problema de segurança confused deputy. Se você ainda não souber o ID da conta de origem na primeira chamada, recomendamos que você coloque o ARN de destino no campo ARN de origem. Nas chamadas subsequentes, você deve definir o ARN de origem como o ARN de origem real que você coletou da primeira chamada. Para obter mais informações, consulte Prevenção de 'confused deputy'.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "ArnLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
     }
}
```

3. Use o comando aws iam create-role para criar a função do IAM, especificando o arquivo de política de confiança que você acabou de criar.

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

Este é um exemplo de saída. Anote o valor `Role.Arn` retornado, pois será necessário utilizá-lo em uma etapa posterior.

```
{
    "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": {
                    "ArnLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. Crie uma política de permissões para definir quais ações CloudWatch os registros podem realizar em sua conta. Primeiro, use um editor de texto para criar uma política de permissões em um arquivo ~/: PermissionsForCWL.json

```
{
```

```
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
}
```

5. Associe a política de permissões com a função inserindo o seguinte comando:

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Depois que o stream de entrega do Firehose estiver no estado ativo e você tiver criado a função IAM, você poderá criar o destino do CloudWatch Logs.

   a. Esta etapa não associará uma política de acesso ao seu destino e só é a primeira etapa das duas concluirá uma criação de destino. Anote o ARN do novo destino que for retornado na carga útil, pois você usará isso como `destination.arn` em uma etapa posterior.

```
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. Depois que a etapa anterior for concluída, na conta destinatária dos dados de log (222222222222), associe uma política de acesso ao destino. Essa política permite que a conta remetente dos dados de log (111111111111 conta) acesse o destino apenas na conta

destinatária dos dados de log (222222222222). Você pode usar um editor de texto para colocar essa política no arquivo ~/AccessPolicy.json:

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. Isso cria uma política que define quem tem acesso de gravação ao destino. Essa política deve especificar as logs:PutAccountPolicy ações logs:PutSubscriptionFilter e para acessar o destino. Cross-account os usuários usarão PutAccountPolicy as ações PutSubscriptionFilter e para enviar eventos de log ao destino.

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
    --access-policy file://~/AccessPolicy.json
```

Etapa 3: criar uma política de filtro de assinatura ao nível da conta

Alterne para a conta de envio, que é 111111111111 neste exemplo. Agora, você criará a política de filtro de assinatura ao nível da conta na conta de envio. Nesse exemplo, o filtro faz com que todos os eventos de log que contêm a string ERR0R em todos os grupos de logs, exceto dois, sejam entregues ao destino que você criou anteriormente.

```
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"
```

Os grupos de logs da conta de envio e o destino devem estar na mesma região da AWS . No entanto, o destino pode apontar para um AWS recurso, como um stream Firehose, localizado em uma região diferente.

Validação do fluxo de eventos de log

Depois de criar o filtro de assinatura, o CloudWatch Logs encaminha todos os eventos de registro recebidos que correspondem ao padrão do filtro e aos critérios de seleção para o stream de entrega do Firehose. Os dados começam a aparecer no bucket do Amazon S3 com base no intervalo de buffer de tempo definido no fluxo de entrega do Firehose. Quando tiver passado tempo suficiente, você poderá conferir seus dados verificando o bucket do Amazon S3. Para verificar o bucket, insira este comando:

```
aws s3api list-objects --bucket 'amzn-s3-demo-bucket'
```

A saída desse comando será semelhante a:

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

Em seguida, você pode recuperar um objeto específico do bucket digitando o comando a seguir. Substitua o valor de `key` pelo valor encontrado no comando anterior.

```
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
```

Os dados no objeto do Amazon S3 são compactados com o formato gzip. É possível examinar os dados brutos na linha de comando usando os seguintes comandos:

Linux

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

Modificação da associação de destino no runtime

Pode haver situações em que você precise adicionar ou remover remetentes de log de um destino que você possua. Você pode usar as `PutAccountPolicy` ações PutDestinationPolicy e em seu destino com a nova política de acesso. No exemplo a seguir, uma conta 111111111111 recém-adicionada é impedida de enviar mais dados de log e a conta 333333333333 é habilitada.

1. Busque a política atualmente associada ao destino TestDestination e anote: AccessPolicy

   ```
   aws logs describe-destinations \
       --destination-name-prefix "testFirehoseDestination"
   ```

   Os dados retornados podem ser assim.

   ```
   {
       "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. Atualize a política para refletir que a conta 111111111111 foi interrompida e a conta 333333333333 está habilitada. Coloque essa política no NewAccessPolicy.json arquivo ~/:

   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"
       }
     ]
   }
   ```

3. Use o comando a seguir para associar a política definida no NewAccessPolicy.json arquivo ao destino:

   ```
   aws logs put-destination-policy \
       --destination-name "testFirehoseDestination" \

       --access-policy file://~/NewAccessPolicy.json
   ```

Por fim, isso desativa os eventos de log do ID da conta 111111111111. Os eventos de log da ID da conta 333333333333 começarão a fluir para o destino assim que o proprietário da conta 333333333333 criar um filtro de assinatura.

Prevenção de ‘confused deputy’

O problema "confused deputy" é um problema de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executar a ação. Em AWS, a falsificação de identidade entre serviços pode resultar no confuso problema do delegado. Cross-service a representação pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado de modo a usar suas permissões para atuar nos recursos de outro cliente de uma forma na qual ele não deveria ter permissão para acessar. Para evitar isso, AWS fornece ferramentas que ajudam você a proteger seus dados para todos os serviços com diretores de serviços que receberam acesso aos recursos da sua conta.

Recomendamos o uso das chaves de contexto de condição global `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` e `aws:SourceOrgPaths` nas políticas de recursos para limitar as permissões ao recurso dadas a outro serviço. Use `aws:SourceArn` se quiser associar apenas um recurso ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços. Use `aws:SourceOrgID` se quiser permitir que qualquer recurso de qualquer conta de uma organização seja associado ao uso entre serviços. Use `aws:SourceOrgPaths` se quiser associar qualquer recurso das contas em um caminho do AWS Organizations seja associado ao uso entre serviços. Para obter mais informações sobre como usar e entender os caminhos, consulte Compreender o caminho da AWS Organizations entidade.

A maneira mais eficaz de se proteger contra o problema do substituto confuso é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso. Se você não souber o ARN completo do recurso ou especificar vários recursos, use a chave de condição de contexto global `aws:SourceArn` com caracteres curinga (*) para as partes desconhecidas do ARN. Por exemplo, `.arn:aws:servicename:*:123456789012:*`

Se o valor do `aws:SourceArn` não contiver o ID da conta, como um ARN de bucket do Amazon S3, você deverá usar ambos, a `aws:SourceAccount` e o `aws:SourceArn` para limitar as permissões.

Para se proteger do problema de "confused deputy" em grande escala, use a chave de contexto de condição global `aws:SourceOrgID` ou `aws:SourceOrgPaths` com o ID ou o caminho da organização do recurso nas políticas baseadas em recursos. As políticas que incluem a chave `aws:SourceOrgID` ou `aws:SourceOrgPaths` incluem automaticamente as contas corretas e você não tem que atualizar manualmente as políticas quando adiciona, remove ou move contas na organização.

As políticas documentadas para conceder acesso aos CloudWatch Logs para gravar dados no Amazon Kinesis Data Streams e no Firehose Etapa 1: criar um destino e Etapa 2: Criar um destino mostram como você pode usar a chave de contexto de condição `aws:SourceArn` global para ajudar a evitar o confuso problema do deputado.

Prevenção de repetição de logs

Existe o risco de causar uma recursão infinita de registros com filtros de assinatura, o que pode levar a um grande aumento na cobrança por ingestão nos CloudWatch registros e em seu destino, se não for evitado. Isso pode ocorrer quando um filtro de assinatura é associado a um grupo de logs que recebe eventos de log como resultado do fluxo de trabalho de entrega da assinatura. Os logs ingeridos no grupo de logs serão entregues ao destino, fazendo com que o grupo de logs ingira mais logs, que serão então encaminhados novamente para o destino, criando um loop recursivo.

Por exemplo, considere um filtro de assinatura com o destino Firehose, que entrega eventos de log para o Amazon S3. Além disso, existe também uma função do Lambda que processa os novos eventos entregues ao Amazon S3 e que também produz alguns logs. Se o filtro de assinatura for aplicado ao grupo de logs da função do Lambda, os eventos de log produzidos pela função serão encaminhados para o Firehose e o Amazon S3 no destino, que então invocarão a função novamente, fazendo com que mais logs sejam produzidos e encaminhados para o Firehose e o Amazon S3, causando outra invocação da função e assim por diante. Isso ocorrerá em um loop infinito, levando a um aumento inesperado nas faturas de ingestão de logs, do Firehose e do Amazon S3.

Se a função Lambda estiver vinculada a uma VPC com registros de fluxo habilitados para CloudWatch registros, o grupo de registros da VPC também poderá causar uma recursão de registros.

Recomendamos que você não aplique filtros de assinatura a grupos de logs que fizerem parte do seu fluxo de trabalho de entrega de assinatura. Para filtros de assinatura ao nível da conta, use o

parâmetro `selectionCriteria` na API `PutAccountPolicy` para excluir esses grupos de logs da política.

Ao excluir grupos de registros, considere os seguintes AWS serviços que produzem registros e podem fazer parte de seus fluxos de trabalho de entrega de assinaturas:

- Amazon EC2 com Fargate
- Lambda
- AWS Funções de etapa
- Registros de fluxo do Amazon VPC que estão habilitados para CloudWatch registros

> **Note**
>
> Eventos de log produzidos por um grupo de logs de um destino do Lambda não serão encaminhados de volta para a função do Lambda para uma política de filtro de assinatura ao nível da conta. Nesse caso, não é necessário excluir o grupo de logs da função do Lambda usando o `selectionCriteria` para políticas de assinaturas de contas.

Sintaxe de padrões de filtros para filtros de métricas, filtros de assinatura, filtros de eventos de log e Live Tail.

> **Note**
>
> Para obter informações sobre como consultar seus grupos de CloudWatch logs com a linguagem de consulta Amazon Logs Insights, consulteCloudWatch Sintaxe de consulta de linguagem do Logs Insights.

Com o CloudWatch Logs, você pode usar filtros métricos para transformar dados de log em métricas acionáveis, filtros de assinatura para encaminhar eventos de log para outros AWS serviços, filtrar eventos de log para pesquisar eventos de log e Live Tail para visualizar interativamente seus registros em tempo real à medida que são ingeridos.

Os padrões de filtro formam a sintaxe que os filtros de métrica, os filtros de assinatura, eventos de log e o Live Tail usam para fazer a correspondência de termos em eventos de log. Os termos podem ser palavras, frases exatas ou valores numéricos. Expressões regulares (regex) podem ser usadas para criar padrões de filtros autônomos ou podem ser incorporadas a padrões de filtro JSON e delimitados por espaços.

Crie padrões de filtro com os termos que você deseja corresponder. Os padrões de filtro retornam apenas os eventos de log que contêm os termos que você definiu. Você pode testar padrões de filtro no CloudWatch console.

Tópicos

- Sintaxe de expressões regulares (regex) compatíveis
- Como usar padrões de filtros para fazer a correspondência de termos usando uma expressão regular (regex)
- Como usar padrões de filtros para fazer a correspondência de termos em eventos de log não estruturados
- Como usar padrões de filtros para fazer a correspondência de termos em eventos de log JSON
- Como usar padrões de filtros para fazer correspondência de termos em eventos de log delimitados por espaços

Sintaxe de expressões regulares (regex) compatíveis

Sintaxe de regex compatível

Ao usar regex para pesquisar e filtrar dados de log, é necessário envolver suas expressões com `%`.

Os padrões de filtros com regex só podem incluir o seguinte:

- Caracteres alfanuméricos: um caractere alfanumérico é um caractere que é uma letra (A a Z ou a a z) ou um dígito (0 a 9).
- Caracteres de símbolo compatíveis: “`:`”, “`_`”, “`#`”, “`=`”, “`@`”, “`/`”, “`;`”, “`,`” e “`-`”. Por exemplo, `%something!%` seria rejeitado porque "`!`" não é aceito.
- Operadores compatíveis: "`^`", "`$`", "`?`", "`[`", "`]`", "`{`", "`}`", "`|`", "`\`", "`*`", "`+`", e "`.`", entre outros.

Os operadores `(` e `)` e não são aceitos. Não é possível usar parênteses para definir um subpadrão.

Multi-byte caracteres não são suportados.

> **Note**
>
> Cotas
> Há um máximo de 5 padrões de filtros contendo regex para cada grupo de logs na criação de filtros de métricas ou filtros de assinatura.
> Há um limite de dois regex para cada padrão de filtro na criação de um padrão de filtro delimitado ou JSON para filtros de métricas e filtros de assinatura, ou ao filtrar eventos de log ou Live Tail.

Uso de operadores compatíveis

- `^`: fixa a correspondência no início de uma string. Por exemplo, `%^[hc]at%` corresponde a "hat" e "cat", mas somente no início de uma string.
- `$`: fixa a correspondência no final de uma string. Por exemplo, `%[hc]at$%` corresponde a "hat" e "cat", mas somente no final de uma string.
- `?`: corresponde a zero ou a uma ocorrência do termo precedente. Por exemplo, `%colou?r%` pode corresponder a "color" e "colour".
- `[]`: define uma classe de caracteres. Corresponde à lista de caracteres ou ao intervalo de caracteres contido nos colchetes. Por exemplo, `%[abc]%` corresponde a "a", "b", ou "c"; `%[a-z]%`

corresponde a qualquer letra minúscula de "a" a "z"; e `%[abcx-z]%` corresponde a "a", "b", "c", "x", "y", ou "z".

- `{m, n}`: corresponde ao termo anterior no mínimo m e não mais que n vezes. Por exemplo, `%a{3,5}%` corresponde somente a "aaa", "aaaa", e "aaaaa".

> **Note**
>
> Tanto m quanto n podem ser omitidos se você optar por não definir um mínimo ou máximo.

- `|`: booleano "Or", que corresponde ao termo em ambos os lados da barra vertical. Por exemplo:
  - `%gray|grey%` pode corresponder a “gray” ou “grey”
  - `%^starting|^initializing|^shutting down%` pode corresponder a “começando... “, ou “inicializando... “, ou “desligando”, mas não corresponderá a “pular a inicialização...”
  - `%abcc|ab[^c]$%`pode corresponder a “abcc...” e “aba”, mas não corresponderá a “aac...”
- `\`: caractere de escape que permite usar o significado literal de um operador em vez de seu significado especial. Por exemplo, `%\[.\]%` corresponde a qualquer caractere único cercado por "[" e "]", pois escape antecede os colchetes, como "[a]", "[b]", "[7]", "[@]", "[]]", e "[ ]".

> **Note**
>
> `%10\.10\.0\.1%` é a maneira correta de criar uma regex que corresponda ao endereço IP 10.10.0.1.

- `*`: corresponde a zero ou mais instâncias do termo precedente. Por exemplo, `%ab*c%` pode corresponder a "ac", "abc" e "abbbc"; `%ab[0-9]*%` pode corresponder a "ab", "ab0" e "ab129".
- `+`: corresponde a uma ou mais instâncias do termo precedente. Por exemplo, `%ab+c%` pode corresponder a "abc", "abbc" e "abbbc", mas não a "ac".
- `.`: corresponde a qualquer caractere sozinho. Por exemplo, `%.at%` corresponde a qualquer string de três caracteres que termina com "at", incluindo "hat", "cat", "bat", "4at", "#at" e " at" (começando com um espaço).

> **Note**
>
> Ao criar uma regex para fazer a correspondência de endereços IP, é importante colocar escape antes do operador `.`. Por exemplo, `%10.10.0.1%` pode corresponder a "10010,051", o que pode não ser o verdadeiro objetivo da expressão.

- `\d,\D`: corresponde a um caractere digit/non de -dígito. Por exemplo, `%\d%` é equivalente a `%[0-9]%` e `%\D%` é equivalente a `%[^0-9]%`.

  > **Note**
  >
  > O operador maiúsculo indica o inverso de sua contraparte minúscula.

- `\s,\S`: corresponde a um caractere de espaço em branco character/non -espaço em branco.

  > **Note**
  >
  > O operador maiúsculo indica o inverso de sua contraparte minúscula. Os caracteres de espaço em branco incluem os caracteres tab (`\t`), espaço ( ) e nova linha (`\n`).

- `\w,\W`: corresponde a um caractere alfanumérico character/non -alfanumérico. Por exemplo, `%\w%` é equivalente a `%[a-zA-Z_0-9]%` e `%\W%` é equivalente a `%[^a-zA-Z_0-9]%`.

  > **Note**
  >
  > O operador maiúsculo indica o inverso de sua contraparte minúscula.

- `\xhh`: corresponde ao mapeamento ASCII para um caractere hexadecimal de dois dígitos. `\x`é a sequência de escape que indica que os caracteres a seguir representam o valor hexadecimal para ASCII. hhespecifica os dois dígitos hexadecimais (0-9 e A-F) que apontam para um caractere na tabela ASCII.

  > **Note**
  >
  > É possível usar `\xhh` para combinar caracteres de símbolo que não são compatíveis com o padrão de filtros. Por exemplo, `%\x3A%` corresponde a `:`; e `%\x28%` corresponde a `(`.

Como usar padrões de filtros para fazer a correspondência de termos usando uma expressão regular (regex)

Fazer a correspondência de termos usando regex

É possível fazer a correspondência de termos em seus eventos de log usando um padrão regex cercado por % (sinais de percentual antes e depois do padrão regex). O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna todos os eventos de log que consistem na palavra-chave AUTORIZADO.

Para obter uma lista de expressões regulares compatíveis, consulte Expressões regulares aceitas.

```
%AUTHORIZED%
```

O padrão de filtro retorna mensagens de eventos de log da seguinte forma:

- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[SUCCESS 200] AUTHORIZED REQUEST`

Como usar padrões de filtros para fazer a correspondência de termos em eventos de log não estruturados

Fazer a correspondência de termos em eventos de log não estruturados

Os exemplos a seguir contêm trechos de código que mostram como é possível usar padrões de filtros para fazer a correspondência de termos em seus eventos de log não estruturados.

> **Note**
>
> Os padrões de filtro diferenciam letras maiúsculas de minúsculas. Coloque frases exatas e termos que incluam caracteres não alfanuméricos entre aspas duplas ("").

Example: Match a single term

O trecho de código a seguir mostra um exemplo de um padrão de filtro de termo único que retorna todos os eventos de log em que as mensagens contêm a palavra ERROR (ERRO).

```
ERROR
```

Este padrão de filtro procura mensagens de eventos de log da seguinte forma:

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

Example: Match multiple terms

O trecho de código a seguir mostra um exemplo de um padrão de filtro de termos múltiplos que retorna todos os eventos de log em que as mensagens contêm as palavras ERRO e ARGUMENTOS.

```
ERROR ARGUMENTS
```

O filtro retorna mensagens de eventos de log da seguinte forma:

- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

O padrão de filtro não retorna as seguintes mensagens de eventos de log porque elas não contêm os dois termos especificados no padrão do filtro.

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`

Example: Match optional terms

É possível usar a correspondência de padrões para criar padrões de filtros que retornem eventos de log contendo termos opcionais. Coloque um ponto de interrogação ("?") antes dos termos que você deseja corresponder. O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna todos os eventos de log em que as mensagens contêm a palavra ERRO ou ARGUMENTOS.

```
?ERROR ?ARGUMENTS
```

Este padrão de filtro procura mensagens de eventos de log da seguinte forma:

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

> **Note**
>
> Não é possível combinar o ponto de interrogação (“?”) com outros padrões de filtro, como termos de inclusão e exclusão. Se você combinar “?” com outros padrões de filtro, todos os termos com pontos de interrogação serão ignorados.
> Por exemplo, o padrão de filtro a seguir corresponde a todos os eventos que contêm a palavra `REQUEST`, mas os termos do filtro de ponto de interrogação (“?”) são ignorados e não têm efeito.
>
> ```
> ?ERROR ?ARGUMENTS REQUEST
> ```
>
> Correspondências de eventos de log
>
> - `[INFO] REQUEST FAILED`
> - `[WARN] UNAUTHORIZED REQUEST`
> - `[ERROR] 400 BAD REQUEST`

Example: Match exact phrases

O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna os eventos de log nos quais as mensagens contêm a frase ERRO INTERNO DO SERVIDOR.

```
"INTERNAL SERVER ERROR"
```

O padrão de filtro retorna a seguinte mensagem de eventos de log:

- `[ERROR 500] INTERNAL SERVER ERROR`

Example: Include and exclude terms

Você pode criar padrões de filtro que retornem eventos de log nos quais as mensagens incluam alguns termos e excluam outros. Coloque um símbolo de menos ("-") antes dos termos que você deseja excluir da busca. O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna eventos de log onde as mensagens incluem o termo ERROR e não incluem o termo ARGUMENTS.

```
ERROR -ARGUMENTS
```

O padrão de filtro retorna mensagens de eventos de log da seguinte forma:

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`

O padrão de filtro não retorna as mensagens de eventos de log a seguir porque elas contêm a palavra ARGUMENTS.

- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

Example: Match everything

É possível corresponder a tudo nos eventos de log usando aspas duplas. O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna todos os eventos de log.

```
" "
```

Como usar padrões de filtros para fazer a correspondência de termos em eventos de log JSON

Escrever padrões de filtros para eventos de log JSON

Os exemplos a seguir descrevem a sintaxe para padrões de filtros que correspondem a termos JSON contendo strings e valores numéricos.

Writing filter patterns that match strings

É possível criar padrões de filtros para corresponder a strings em eventos de log JSON. O trecho de código a seguir mostra um exemplo de sintaxe de padrão de filtro baseado em string.

```
{ PropertySelector EqualityOperator String }
```

Coloque os padrões de filtro entre colchetes (“{}”). String-based os padrões de filtro devem conter as seguintes partes:

- Seletor de propriedades

  Defina seletores de propriedades com um cifrão seguido de um ponto ("$."). Os seletores de propriedade são strings alfanuméricas compatíveis com os caracteres de hífen ("-") e underline ("_"). Strings não são compatíveis com notação científica. Seletores de propriedades apontam para nós de valor em eventos de log JSON. Os nós de valor podem ser strings ou números. Coloque matrizes após seletores de propriedades. Os elementos nas matrizes seguem um sistema de numeração baseado em zero, o que significa que o primeiro elemento na matriz é o elemento 0, o segundo elemento é o elemento 1 e assim por diante. Coloque elementos entre

colchetes ("[]"). Se um seletor de propriedades apontar para uma matriz ou um objeto, o padrão de filtro não corresponderá ao formato do log. Se a propriedade JSON contiver um ponto (" . "), a notação de colchetes poderá ser usada para selecionar essa propriedade.

> **Note**
>
> Seletor de curingas
> É possível usar o curinga JSON para selecionar qualquer elemento da matriz ou qualquer campo de objeto JSON.
> Cotas
> É possível usar no máximo um seletor de curinga em um seletor de propriedades.

- Operador de igualdade

  Defina operadores de igualdade com um dos seguintes símbolos: igual ("=") ou diferente ("!="). Os operadores de igualdade retornam um valor booliano (true ou false).

- String

  Você pode colocar strings entre aspas duplas (""). Strings que contenham tipos que não sejam caracteres alfanuméricos ou underline devem ser colocadas entre aspas duplas. Use o asterisco ("*") como curinga para corresponder ao texto.

> **Note**
>
> É possível usar qualquer expressão regular condicional ao criar padrões de filtros para fazer a correspondência com termos em eventos de log JSON. Para obter uma lista de expressões regulares compatíveis, consulte Expressões regulares aceitas.

O trecho de código a seguir contém um exemplo de um padrão de filtro que mostra como formatar um padrão de filtro para corresponder um termo JSON a uma string.

```
{ $.eventType = "UpdateTrail" }
```

Writing filter patterns that match numeric values

É possível criar padrões de filtros para fazer a correspondência de valores numéricos em eventos de log JSON. O seguinte trecho de código mostra um exemplo de sintaxe para padrões de filtros que correspondem a valores numéricos.

```
{ PropertySelector NumericOperator Number }
```

Coloque padrões de filtros entre chaves ("{}"). Os padrões de filtros que correspondem a valores numéricos devem ter as seguintes partes:

- Seletor de propriedades

  Defina seletores de propriedades com um cifrão seguido de um ponto ("$."). Os seletores de propriedade são strings alfanuméricas compatíveis com os caracteres de hífen ("-") e underline ("_"). Strings não são compatíveis com notação científica. Seletores de propriedades apontam para nós de valor em eventos de log JSON. Os nós de valor podem ser strings ou números. Coloque matrizes após seletores de propriedades. Os elementos nas matrizes seguem um sistema de numeração baseado em zero, o que significa que o primeiro elemento na matriz é o elemento 0, o segundo elemento é o elemento 1 e assim por diante. Coloque elementos entre colchetes ("[]"). Se um seletor de propriedades apontar para uma matriz ou um objeto, o padrão de filtro não corresponderá ao formato do log. Se a propriedade JSON contiver um ponto ("."), a notação de colchetes poderá ser usada para selecionar essa propriedade.

  > **Note**
  >
  > Seletor de curingas
  > É possível usar o curinga JSON para selecionar qualquer elemento da matriz ou qualquer campo de objeto JSON.
  > Cotas
  > É possível usar no máximo um seletor de curinga em um seletor de propriedades.

- Operador numérico

  Defina operadores numéricos com um dos seguintes símbolos: maior que (">"), menor que ("<"), igual a ("="), diferente de ("! ="), maior ou igual a (">="), ou menor ou igual a ("<=").

- Número

Você pode usar números inteiros que contenham os símbolos de mais ("+") ou menos ("-") e seguir a notação científica. Use o asterisco ("*") como curinga para corresponder números.

O trecho de código a seguir contém exemplos que mostram como formatar padrões de filtros para fazer a correspondência de termos JSON com valores numéricos.

```
// 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 }
```

Fazer a correspondência de termos em eventos de log JSON usando expressões simples

Os exemplos a seguir contêm trechos de código que mostram como padrões de filtros podem fazer a correspondência de termos em um evento de log JSON.

> **Note**
>
> Se você testar um padrão de filtro de exemplo com o evento de log JSON de exemplo, deverá inserir o log JSON de exemplo em uma única linha.

Evento de 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

Esse padrão de filtro corresponde à string `"UpdateTrail"` na propriedade `"eventType"`.

```
{ $.eventType = "UpdateTrail" }
```

Example: Filter pattern that matches string values (IP address)

O padrão de filtro contém um curinga e corresponde à propriedade `"sourceIPAddress"` porque ele não contém um número com o prefixo `"123.123."`.

```
{ $.sourceIPAddress != 123.123.* }
```

Example: Filter pattern that matches a specific array element with a string value

Esse padrão de filtro corresponde ao elemento `"value"` na matriz `"arrayKey"`.

```
{ $.arrayKey[0] = "value" }
```

Example: Filter pattern that matches a string using regex

Esse padrão de filtro corresponde à string `"Trail"` na propriedade `"eventType"`.

```
{ $.eventType = %Trail% }
```

Example: Filter pattern that uses a wildcard to match values of any element in the array using regex

O padrão de filtro contém uma regex que corresponde ao elemento `"value"` na matriz `"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)

O padrão de filtro contém uma regex que corresponde ao elemento `"111.111.111.111"` na propriedade `"sourceIPAddress"`.

```
{ $.* = %111\.111\.111\.1[0-9]{1,2}% }
```

> **Note**
>
> Cotas
> É possível usar no máximo um seletor de curinga em um seletor de propriedades.

Example: Filter pattern that matches a JSON property with a period (.) in the key

Você pode criar padrões de filtro para corresponder aos nomes de propriedades JSON que contêm um ponto (`.`) usando a notação de colchetes. Coloque o nome da propriedade entre

aspas simples ()'. As aspas duplas ('') não são suportadas dentro da notação de colchetes e fazem com que o padrão do filtro falhe com um erro de sintaxe. O padrão de filtro a seguir corresponde à propriedade`cluster.name`.

```
{ $.['cluster.name'] = "c" }
```

Example: Filter pattern that matches JSON logs using IS

É possível criar padrões de filtros que correspondam a campos em logs JSON com a variável `IS`. A variável `IS` pode corresponder campos que contenham os valores `NULL`, `TRUE` ou `FALSE`. O padrão de filtro a seguir retorna logs JSON para os quais o valor de `SomeObject` é `NULL`.

```
{ $.SomeObject IS NULL }
```

Example: Filter pattern that matches JSON logs using NOT EXISTS

Você pode criar padrões de filtros com a variável `NOT EXISTS` para encontrar logs JSON que não contenham determinados campos nos dados de log. O padrão de filtro a seguir usa `NOT EXISTS` para retornar logs JSON que não contenham o campo `SomeOtherObject`.

```
{ $.SomeOtherObject NOT EXISTS }
```

> **Note**
>
> No momento, não há compatibilidade com as variáveis `IS NOT` e `EXISTS`.

Fazer a correspondência de termos em objetos JSON usando expressões compostas

É possível usar os operadores lógicos AND ("&&") e OR ("||") em padrões de filtros para criar expressões compostas que correspondam a eventos de log nos quais duas ou mais condições são verdadeiras. Expressões compostas são compatíveis com o uso de parênteses ("()") e com a seguinte ordem padrão de operações: () > && > ||. Os exemplos a seguir contêm trechos de código

que mostram como usar padrões de filtros com expressões compostas para corresponder a termos em um objeto JSON.

Objeto 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 (&&)

Este padrão de filtro contém uma expressão composta que faz a correspondência de `"id"` em `"user"` com um valor numérico de 1 e de `"email"` no primeiro elemento da matriz `"users"` com a string `"John.Doe@example.com"`.

```
{ ($.user.id = 1) && ($.users[0].email = "John.Doe@example.com") }
```

Example: Expression that matches using OR (||)

Esse padrão de filtro contém uma expressão composta que faz a correspondência de `"email"` em `"user"` com a 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 (&&)

Esse padrão de filtro contém uma expressão composta que não encontra uma correspondência porque a expressão não corresponde à terceira ação em `"actions"`.

```
{ ($.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch") &&
 $.actions[2] = "nonmatch" }
```

> **Note**
>
> Cotas
> É possível usar no máximo um seletor de curinga em um seletor de propriedades e até três seletores de curinga em um padrão de filtro com expressões compostas.

Example: Expression that doesn't match using OR (||)

O padrão de filtro contém uma expressão composta que não encontra uma correspondência porque a expressão não corresponde à primeira propriedade em `"users"` ou à terceira ação em `"actions"`.

```
{ ($.user.id = 2 && $.users[0].email = "nonmatch") || $.actions[2] = "GET" }
```

Como usar padrões de filtros para fazer correspondência de termos em eventos de log delimitados por espaços

Escrever padrões de filtros para eventos de log delimitados por espaços

É possível criar padrões de filtros para fazer a correspondência de termos em eventos de log delimitados por espaços Veja a seguir um exemplo de evento de log delimitado por espaços e uma descrição de como escrever a sintaxe para padrões de filtros que correspondam aos termos do evento de log delimitado por espaços.

> **Note**
>
> É possível usar qualquer expressão regular condicional ao criar padrões de filtros para fazer a correspondência de termos em eventos de log delimitados por espaços. Para obter uma lista de expressões regulares compatíveis, consulte Expressões regulares aceitas.

Example: Space-delimited log event

O trecho de código a seguir mostra um evento de log delimitado por espaço que contém sete campos: `ip`, `user`, `username`, `timestamp`, `request`, `status_code` e `bytes`.

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404
 1534
```

> **Note**
>
> Caracteres entre colchetes ("[]") e entre aspas duplas ("") são considerados campos únicos.

Writing filter patterns that match terms in a space-delimited log event

Para criar um padrão de filtro que corresponda a termos em um evento de log delimitado por espaços, coloque o padrão de filtro entre colchetes ("[]") e especifique campos com nomes separados por vírgulas (","). O padrão de filtro a seguir analisa sete campos.

```
[ip=%127\.0\.0\.[1-9]%, user, username, timestamp, request =*.html*, status_code =
 4*, bytes]
```

É possível usar operadores numéricos (**>**, **<**, **=**, **!=**, **>=** ou **<=**) e o asterisco (*) como um curinga ou uma regex para fornecer condições de padrão de filtro. No exemplo de padrão de filtro, `ip` usa uma regex que corresponde ao intervalo de endereços IP 127.0.0.1 a 127.0.0.9, `request` contém um curinga que indica que ele deve extrair um valor com `.html` e `status_code` contém um curinga que afirma que deve extrair um valor começando com `4`.

Se não souber o número de campos que está analisando em um evento de log delimitado por espaço, você poderá usar reticências (…) para fazer referência a qualquer campo sem nome. As reticências podem fazer referência a quantos campos forem necessários. O exemplo a seguir mostra um padrão de filtro com reticências que representam os primeiros quatro campos sem nome mostrados no padrão de filtro de exemplo anterior.

```
[..., request =*.html*, status_code = 4*, bytes]
```

Você também pode usar os operadores lógicos AND (&&) e OR (||) para criar expressões compostas. O padrão de filtro a seguir contém uma expressão composta que indica que o valor de `status_code` deve ser `404` ou `410`.

```
[ip, user, username, timestamp, request =*.html*, status_code = 404 || status_code =
 410, bytes]
```

Fazer a correspondência de termos em eventos de log delimitados por espaços usando correspondência de padrões

É possível usar a correspondência de padrões para criar padrões de filtros delimitados por espaços que correspondam a termos em uma ordem específica. Especifique a ordem de seus termos com indicadores. Use w1 para representar seu primeiro termo e w2 e assim por diante para representar a ordem dos termos subsequentes. Coloque vírgulas (",") entre os termos. Os exemplos a seguir contêm trechos de código que mostram como é possível usar a correspondência de padrões com padrões de filtros delimitados por espaços.

> **Note**
>
> É possível usar qualquer expressão regular condicional ao criar padrões de filtros para fazer a correspondência de termos em eventos de log delimitados por espaços. Para obter uma lista de expressões regulares compatíveis, consulte Expressões regulares aceitas.

Space-delimited evento de log

```
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

O padrão de filtro delimitado por espaços retorna eventos de log em que a primeira palavra nos eventos de log é ERROR.

```
[w1=ERROR, w2]
```

> **Note**
>
> Ao criar filtros de métrica delimitados por espaços que usam correspondência de padrões, é necessário incluir um indicador em branco depois de especificar a ordem dos termos.

Por exemplo, se você criar um padrão de filtro que retorne eventos de logs onde a primeira palavra seja ERROR, inclua um indicador w2 em branco após o termo w1.

Example: Match terms with AND (&&) and OR (||)

É possível usar os operadores lógicos AND ("&&") e OR ("||") para criar padrões de filtros delimitados por espaços que contenham condições. O padrão de filtro a seguir retorna eventos de log onde a primeira palavra nos eventos é ERROR ou WARNING.

```
[w1=ERROR || w1=WARNING, w2]
```

Example: Exclude terms from matches

É possível criar padrões de filtros delimitados por espaços que retornam eventos de log excluindo um ou mais termos. Coloque um símbolo de diferente ("-") antes do termo ou dos termos que você deseja excluir. O trecho de código a seguir mostra um exemplo de um padrão de filtro que retorna eventos de log em que as primeiras palavras não são ERROR e WARNING.

```
[w1!=ERROR && w1!=WARNING, w2]
```

Example: Match the top level item in a resource URI

O trecho de código a seguir mostra um exemplo de um padrão de filtro que corresponde ao item de nível superior em um URI de recurso usando regex.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+$%, response_time]
```

Example: Match the child level item in a resource URI

O trecho de código a seguir mostra um exemplo de um padrão de filtro que corresponde ao item de nível filho em um URI de recurso usando regex.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+/part/[0-9]+$%,
 response_time]
```

Ativar o registro de AWS serviços

Use o catálogo pesquisável para encontrar um AWS serviço e abrir seu guia de configuração de registro. Para obter mais informações sobre os destinos e modelos de permissões para cada serviço, consulte Destinos e permissões de log compatíveis.

Muitos serviços publicam registros somente no CloudWatch Logs, mas outros usam a entrega automática de registros para enviar registros diretamente para o Amazon Simple Storage Service ou o Amazon Data Firehose. A entrega direta é útil quando seu principal requisito é armazenamento ou processamento de longo prazo em um desses destinos.

Mesmo quando você publica registros diretamente no Amazon S3 ou no Firehose, CloudWatch as taxas de entrega se aplicam. Se você enviar registros para o Amazon S3, *AWS_REGION*-S3-Egress-Bytes as cobranças aparecerão no Cost Explorer ou na sua fatura. Se você enviar registros para o Firehose, as *AWS_REGION*-FH-Egress-Bytes cobranças serão exibidas. Para obter mais informações sobre preços de registros vendidos, consulte a guia Registros em CloudWatch Preços da Amazon.

Alguns serviços exigem permissões adicionais antes de poderem entregar os registros. Sem essas permissões, a entrega do registro falha. Na tabela de comparação, os serviços que usam o modelo de permissões original são rotulados como Suportados (permissões V1). Os serviços que usam o modelo atual são rotulados como Suportados (permissões V2). Cada etiqueta está vinculada às políticas necessárias.
<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>
<integration></integration>

<integration></integration>
<integration></integration>
<integration></integration>

<integration></integration>
<integration></integration>
<integration></integration>
</integration-catalog>

Destinos e permissões de registro suportados

A tabela a seguir mostra quais destinos cada AWS serviço oferece suporte para o envio de registros e qual versão de permissões é necessária.

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros de acesso do Amazon API Gateway	Logs fornecidos	Compatível (permissões V1)			
Logs de acesso do Application Load Balancer	Logs fornecidos	[Permissões v2] compatíveis	[Permissões v2] compatíveis	[Permissões v2] compatíveis	
AWS AppSync logs	Logs personalizados	Compatível			
Registros do Amazon Aurora MySQL	Logs personalizados	Compatível			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Amazon Bedrock Registro de bases de conhecimento	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Amazon Bedrock Registro do agente	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Amazon Bedrock AgentCore Runtime	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Amazon Bedrock AgentCore Gateway	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Amazon Bedrock AgentCore Identidade	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Amazon Bedrock AgentCore Memória	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Amazon Bedrock AgentCore Pagamentos	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Amazon Bedrock AgentCore Ferramentas	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)
Registros métricos de qualidade de mídia e registros de mensagens SIP do Amazon Chime	Logs fornecidos	Compatível (permissões V1)			
CloudFront: registros de acesso	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
AWS CloudHSM registros de auditoria	Logs personalizados	Compatível			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
CloudWatch Evidentemente, os registros de eventos de avaliação	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)		
CloudWatch Registros do Internet Monitor	Logs fornecidos		Compatível (permissões V1)		
the section called “Crie uma fonte de entrega para CloudWatch os registros de execução de consultas do Logs Insights”	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
CloudTrail logs	Logs personalizados	Compatível			
AWS CodeBuild logs	Logs personalizados	Compatível			
Amazon CodeWhisperer registros de eventos	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Amazon Cognito logs	Logs fornecidos	Compatível (permissões V1)			
Registros de clientes do Amazon Connect	Logs personalizados	Compatível			
AWS DataSync logs	Logs personalizados	Compatível			
Agente do AWS DevOps logs	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros da Amazon ElastiCache (Redis OSS)	Logs fornecidos	Compatível (permissões V1)		Compatível (permissões V1)	
AWS Elastic Beanstalk logs	Logs personalizados	Compatível			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros do Amazon Elastic Container Service	Logs personalizados	Compatível			
Registros do modo automático do Amazon Elastic Kubernetes Service	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de capacidade do Amazon Elastic Kubernetes Service	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros do plano de controle do Amazon Elastic Kubernetes Service	Logs fornecidos	Compatível			
AWS Elemental MediaPackage registros de acesso	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
AWS Elemental MediaTailor logs	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
AWS Entity Resolution logs	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Amazon EventBridge Registro de tubulações	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Amazon EventBridge ônibus para eventos	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Amazon EventBridge registros de barramento de eventos (personalizados)	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
AWS Fargate logs	Logs personalizados	Compatível			
AWS Fault Injection Service registros de experimentos	Logs fornecidos		Compatível (permissões V1)		

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Amazon FinSpace	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
AWS Global Accelerator registros de fluxo	Logs fornecidos		Compatível (permissões V1)		
AWS Glue registros de trabalho	Logs personalizados	Compatível			
Registros de erros do IAM Identity Center	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de bate-papo do Amazon Interactive Video Service	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
AWS IoT logs	Logs personalizados	Compatível			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
AWS IoT FleetWise logs	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
AWS Lambda logs	Logs fornecidos	Compatível	Compatível	Compatível	
Registros do Amazon Macie	Logs personalizados	Compatível			
AWS Mainframe Modernization	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Registros do Amazon Managed Service para Prometheus	Logs fornecidos	Compatível (permissões V1)			
Registros do agente Amazon MSK	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros do Amazon MSK Connect	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Registros do Amazon MQ	Logs personalizados	Compatível			
AWS Registros do Firewall de Rede	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
AWS Registros do Network Firewall Proxy	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de acesso ao Network Load Balancer	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
OpenSearch logs	Logs personalizados	Compatível			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros OpenSearch de ingestão do Amazon Service	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
AWS PCS logs	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros do conector Amazon Q Business	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de conversas comerciais do Amazon Q	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de eventos do agente Amazon Q in Connect AI	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros rápidos da Amazon	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Logs do Amazon Relational Database Service (Amazon Relational Database Service) do PostgreSQL	Logs personalizados	Compatível			
Registros de consulta do Amazon Route 53 Global Resolver	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de consulta de DNS públicos do Amazon Route 53	Logs fornecidos	Compatível			
Registros de consulta do resolvedor Amazon Route 53	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)		
AWS Registros do RTB Fabric	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
AWS CSPM do Security Hub	Logs fornecidos	Compatível (permissões V2)			
AWS Security Hub	Logs fornecidos	Compatível (permissões V2)			
AWS Registros de fluxo de ataque do Shield Advanced	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros de acesso ao servidor Amazon Simple Storage Service	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Eventos de SageMaker IA da Amazon	Logs fornecidos	Compatível (permissões V1)			

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Eventos para trabalhadores da Amazon SageMaker AI	Logs fornecidos	Compatível (permissões V1)			
AWS Site-to-Site Registros de VPN	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Registros do Amazon Simple Email Service	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
Registros do Amazon Simple Notification Service	Logs personalizados	Compatível			
Registros da política de proteção de dados do Amazon Simple Notification Service	Logs personalizados	Compatível			
Arquivos de feed de dados da EC2 Spot Instance	Logs fornecidos		Compatível (permissões V1)		

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
AWS Step Functions Registros de fluxo de trabalho expresso e fluxo de trabalho padrão	Logs fornecidos	Compatível (permissões V1)			
Registros de auditoria e registros de integridade do Storage Gateway	Logs fornecidos	Compatível (permissões V1)			
AWS Transfer Family logs	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Acesso Verificado pela AWS logs	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Registros de fluxo da Amazon Virtual Private Cloud	Logs fornecidos	Compatível	Compatível (permissões V1)	Compatível (permissões V1)	

Origem do log	Tipo de log	the section called “Registros enviados para CloudWatch Logs”	the section called “Logs enviados ao Amazon S3”	the section called “Logs enviados ao Firehose”	the section called “Traços enviados para X-Ray”
Registros de acesso do Amazon VPC Lattice	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível (permissões V1)	
Amazon VPC Route Server	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	
AWS WAF logs	Logs fornecidos	Compatível (permissões V1)	Compatível (permissões V1)	Compatível	
Amazon WorkMail registros de auditoria	Logs fornecidos	Compatível (permissões V2)	Compatível (permissões V2)	Compatível (permissões V2)	

Registro em log que requer permissões [v1] adicionais

Alguns AWS serviços usam uma infraestrutura comum para enviar seus CloudWatch registros para Logs, Amazon S3 ou Firehose. Para permitir que os AWS serviços listados na tabela anterior enviem seus registros para esses destinos, você deve estar logado como um usuário com determinadas permissões.

Além disso, é necessário conceder permissões AWS para permitir que os registros sejam enviados. AWS pode criar automaticamente essas permissões quando os registros são configurados, ou você mesmo pode criá-las antes de configurar o registro. Para entrega entre contas, você mesmo deve criar manualmente as políticas de permissão.

Se você optar por configurar AWS automaticamente as permissões e as políticas de recursos necessárias quando você ou alguém em sua organização configura o envio de registros pela primeira vez, o usuário que está configurando o envio de registros deverá ter determinadas permissões, conforme explicado posteriormente nesta seção. Se preferir, você pode criar as políticas de recursos, e os usuários que configurarem o envio de logs não precisarão de tantas permissões.

Os tópicos a seguir fornecem mais detalhes sobre cada um desses destinos.

Tópicos

- Registros enviados para CloudWatch Logs
- Logs enviados ao Amazon S3
- Logs enviados ao Firehose

Registros enviados para CloudWatch Logs

> **Important**
>
> Quando você configura os tipos de registro na lista a seguir para serem enviados ao CloudWatch Logs, AWS cria ou altera as políticas de recursos associadas ao grupo de registros que recebe os registros, se necessário. Continue lendo esta seção para ver os detalhes.

Esta seção se aplica quando os tipos de registros listados na tabela da seção anterior são enviados para CloudWatch Logs:

Permissões de usuário

Para poder configurar o envio de qualquer um desses tipos de CloudWatch registros para o Logs pela primeira vez, você precisa estar conectado a uma conta com as seguintes permissões.

- `logs:CreateLogDelivery`

- `logs:PutResourcePolicy`
- `logs:DescribeResourcePolicies`
- `logs:DescribeLogGroups`

> **Note**
>
> Quando você especificar a permissão `logs:DescribeLogGroups`, `logs:DescribeResourcePolicies` ou `logs:PutResourcePolicy`, certifique-se de definir o ARN da linha `Resource` para que use um caractere curinga `*`, em vez de especificar apenas um único nome de grupo de logs. Por exemplo, `"Resource": "arn:aws:logs:us-east-1:111122223333:log-group:*"`.

Se algum desses tipos de registros já estiver sendo enviado para um grupo de CloudWatch registros no Logs, para configurar o envio de outro desses tipos de registros para esse mesmo grupo de registros, você só precisará da `logs:CreateLogDelivery` permissão.

Política de recursos do grupo de logs

O grupo de logs para o qual os logs estão sendo enviados deve ter uma política de recursos que contenha determinadas permissões. Se o grupo de registros atualmente não tiver uma política de recursos e o usuário que configura o registro tiver as `logs:DescribeLogGroups` permissões`logs:PutResourcePolicy`,`logs:DescribeResourcePolicies`, e para o grupo de registros, AWS criará automaticamente a política a seguir quando você começar a enviar os CloudWatch registros para o Logs. Para assinaturas recém-criadas, as políticas de recursos são configuradas no nível do grupo de registros e têm um tamanho máximo de 51.200 bytes. Se uma política de recursos existente em nível de conta já conceder permissões por meio de curingas, uma política separada em nível de grupo de registros não seria criada. Para verificar a política de Group-level recursos de log de um grupo de log específico, use o `describe-resource-policies` comando com o `--resource-arn` parâmetro definido como o ARN do grupo de log e o `--policy-scope` parâmetro definido como. `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:*"
                    ]
                }
            }
        }
    ]
}
```

O limite da política de recursos do grupo de registros é de 51.200 bytes. Quando esse limite é atingido, a AWS não pode adicionar novas permissões. Isso exige que os clientes modifiquem manualmente a política para conceder ao principal `delivery.logs.amazonaws.com` serviço permissões sobre as `logs:PutLogEvents` ações `logs:CreateLogStream` e. Os clientes devem usar um prefixo de nome de grupo de registros com curingas, como, `/aws/vendedlogs/*` e usar esse nome de grupo de registros para a criação futura do Delivery.

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:*"
                    ]
                }
            }
        }
    ]
}
```

Logs enviados ao Amazon S3

Quando você define os registros a serem enviados para o Amazon S3, AWS cria ou altera as políticas de recursos associadas ao bucket do S3 que está recebendo os registros, se necessário.

Os logs publicados diretamente no Amazon S3 são publicados em um bucket especificado por você. Um ou mais arquivos de log são criados a cada cinco minutos no bucket especificado.

Quando você entrega logs a um bucket do Amazon S3 pela primeira vez, o serviço que entrega logs registra o proprietário do bucket para garantir que os logs sejam entregues somente a um bucket pertencente a essa conta. Consequentemente, para alterar o proprietário do bucket do Amazon S3, é necessário recriar ou atualizar a assinatura de log no serviço de origem.

> **Note**
>
> CloudFront usa um modelo de permissões diferente dos outros serviços que enviam registros vendidos para o S3. Para obter mais informações, consulte Permissões necessárias para configurar o registro padrão e acessar seus arquivos de log.
> Além disso, se você usar o mesmo bucket do S3 para registros de CloudFront acesso e outra fonte de log, habilitar a ACL no bucket CloudFront também concederá permissão a todas as outras fontes de log que usam esse bucket.

> **Important**
>
> Se você for enviar logs para um bucket do Amazon S3 e a política do bucket contiver um elemento `NotAction` ou `NotPrincipal`, adicionar automaticamente permissões de entrega de logs ao bucket e criar uma assinatura de log não funcionará. Para criar uma assinatura de logs com sucesso, você precisa adicionar manualmente as permissões de entrega de logs à política do bucket e depois criar a assinatura de logs. Para obter mais informações, consulte as instruções desta seção.
> Se o bucket tiver criptografia do lado do servidor usando uma AWS KMS chave gerenciada pelo cliente, você também deverá adicionar a política de chaves para sua chave gerenciada pelo cliente. Para obter mais informações, consulte Amazon S3.
> Se o bucket de destino tiver SSE-KMS uma chave de bucket ativada, a política de chave KMS gerenciada pelo cliente anexada não funcionará mais conforme o esperado para todas as solicitações. Para obter mais informações, consulte Redução do custo SSE-KMS com as chaves de bucket do Amazon S3.
> Se você estiver usando registros vendidos e criptografia S3 com uma AWS KMS chave gerenciada pelo cliente, deverá usar um AWS KMS ARN de chave totalmente qualificado em vez de um ID de chave ao configurar o bucket. Para obter mais informações, consulte Criptografia do bucket PUT.

Permissões de usuário

Para poder configurar o envio de qualquer um desses tipos de logs ao Amazon S3 pela primeira vez, é necessário conectar-se a uma conta com as permissões a seguir.

- `logs:CreateLogDelivery`
- `S3:GetBucketPolicy`
- `S3:PutBucketPolicy`

Se algum desses tipos de logs já estiver sendo enviado a um bucket do Amazon S3, para configurar o envio de outro desses tipos de logs para esse mesmo bucket, apenas a permissão `logs:CreateLogDelivery` será necessária.

Política de recursos do bucket do S3

O bucket do S3 para o qual os logs estão sendo enviados deve ter uma política de recursos que contenha determinadas permissões. Se o bucket atualmente não tem uma política de recursos e o usuário que está configurando o log tem as permissões `S3:GetBucketPolicy` e `S3:PutBucketPolicy` para o bucket, a AWS cria automaticamente a seguinte política quando você começa a enviar os logs ao 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:*"
                    ]
                }
            }
        }
    ]
}
```

Na política anterior, para `aws:SourceAccount`, especifique a lista de IDS de conta para os quais os logs estão sendo entregues a esse bucket. Para `aws:SourceArn`, especifique a lista de ARNs do recurso que gera os logs, no formulário `arn:aws:logs:`*`source-region`*`:`*`source-account-id`*`:*`.

Se o bucket tiver uma política de recursos, mas ela não contiver a instrução exibida na política anterior, e o usuário configurando o log tiver as permissões `S3:GetBucketPolicy` e `S3:PutBucketPolicy` para o bucket, essa instrução será anexada à política de recursos do bucket.

> **Note**
>
> Em alguns casos, você pode ver `AccessDenied` erros AWS CloudTrail se a `s3:ListBucket` permissão não tiver sido concedida`delivery.logs.amazonaws.com`. Para evitar esses erros em seus CloudTrail registros, você deve conceder a `s3:ListBucket` permissão `delivery.logs.amazonaws.com` e incluir `Condition` os parâmetros mostrados com o conjunto de `s3:GetBucketAcl` permissões na política de bucket anterior. Para simplificar isso, em vez de criar uma nova `Statement`, você pode atualizar `AWSLogDeliveryAclCheck` diretamente para `“Action”: [“s3:GetBucketAcl”, “s3:ListBucket”]`

Criptografia no lado do servidor de bucket do Amazon S3

Você pode proteger os dados em seu bucket do Amazon S3 habilitando a criptografia do lado do servidor com as chaves da S3-managed Amazon SSE-S3 () ou a criptografia do lado do servidor com uma chave armazenada em (). AWS KMS AWS Key Management Service SSE-KMS Para obter mais informações, consulte Proteger dados usando a criptografia no lado do servidor.

Se você escolher SSE-S3, nenhuma configuração adicional será necessária. O Amazon S3 lida com a chave de criptografia.

> **Warning**
>
> Se você escolher SSE-KMS, deverá usar uma chave gerenciada pelo cliente, pois o uso de uma chave AWS gerenciada não é suportado nesse cenário. Se você configurar a criptografia usando uma chave AWS gerenciada, os registros serão entregues em um formato ilegível.

Ao usar uma AWS KMS chave gerenciada pelo cliente, você pode especificar o Amazon Resource Name (ARN) da chave gerenciada pelo cliente ao ativar a criptografia do bucket. Você precisa adicionar o seguinte à política de chaves da chave gerenciada pelo cliente (não à política de bucket para o bucket do S3), para que a conta de entrega de log possa gravar no bucket do S3.

Se você escolher SSE-KMS, deverá usar uma chave gerenciada pelo cliente, pois o uso de uma chave AWS gerenciada não é suportado nesse cenário. Ao usar uma AWS KMS chave gerenciada pelo cliente, você pode especificar o Amazon Resource Name (ARN) da chave gerenciada pelo

cliente ao ativar a criptografia do bucket. Você precisa adicionar o seguinte à política de chaves da chave gerenciada pelo cliente (não à política de bucket para o bucket do S3), para que a conta de entrega de log possa gravar no bucket do S3.

```
{
    "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:*"]
        }
        }
}
```

Para `aws:SourceAccount`, especifique a lista de IDS de conta para os quais os logs estão sendo entregues a esse bucket. Para `aws:SourceArn`, especifique a lista de ARNs do recurso que gera os logs, no formulário `arn:aws:logs:`*`source-region`*`:`*`source-account-id`*`:*`.

Logs enviados ao Firehose

Esta seção se aplica quando os tipos de logs listados na tabela da seção anterior são enviados ao Firehose:

Permissões de usuário

Para poder configurar o envio de qualquer um desses tipos de logs ao Firehose pela primeira vez, você deve fazer login em uma conta com as permissões a seguir.

- `logs:CreateLogDelivery`
- `firehose:TagDeliveryStream`
- `iam:CreateServiceLinkedRole`

Se algum desses tipos de log já estiver sendo enviado ao Firehose, para configurar o envio de outro tipo de log para o Firehose, você só precisará ter as permissões `logs:CreateLogDelivery` e `firehose:TagDeliveryStream`.

Funções do IAM usadas para permissões

Como o Firehose não usa políticas de recursos, AWS usa funções do IAM ao configurar esses registros para serem enviados ao Firehose. AWS cria uma função vinculada ao serviço chamada. AWSServiceRoleForLogDelivery Essa função vinculada ao serviço inclui as permissões a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

Essa função vinculada ao serviço concede permissão para todos os streams de entrega do Firehose que têm a tag definida como. `LogDeliveryEnabled` `true` AWS fornece essa tag ao stream de entrega de destino quando você configura o registro.

Essa função vinculada ao serviço também tem uma política de confiança que permite que a entidade de serviço `delivery.logs.amazonaws.com` assuma a função vinculada ao serviço necessária. Essa política de confiança é a seguinte:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Registro em log que requer permissões [v2] adicionais

Alguns AWS serviços usam um novo método para enviar seus registros. Esse é um método flexível que permite configurar a entrega de registros desses serviços para um ou mais dos seguintes destinos: CloudWatch Logs, Amazon S3 ou Firehose e X-Ray para entrega de rastreamento.

A entrega de um log de trabalho consiste em três elementos:

- Uma `DeliverySource`, que é um objeto lógico que representa os recursos que, de fato, enviam os logs.
- Um `DeliveryDestination`, que é um objeto lógico que representa o destino da entrega de fato .
- Uma `Delivery`, que conecta a origem da entrega ao destino da entrega

Para configurar a entrega de registros entre um AWS serviço compatível e um destino, você deve fazer o seguinte:

- Crie uma fonte de entrega com PutDeliverySource.
- Crie um destino de entrega com PutDeliveryDestination.

- Se você estiver entregando registros entre contas, poderá usar somente destinos Amazon S3 e Firehose. Você deve usar PutDeliveryDestinationPolicyna conta de destino para atribuir uma IAM política ao destino. Essa política autoriza a criação de uma entrega da origem da entrega na conta A para o destino da entrega na conta B. Para entrega entre contas, você mesmo deve criar manualmente as políticas de permissão. Para exemplos de configuração, consulteCross-account exemplo de entrega.
- Crie uma entrega combinando exatamente uma fonte de entrega e um destino de entrega, usando CreateDelivery.

Exemplos de configuração de entrega de registros

Os exemplos a seguir criam a origem e o destino da entrega na mesma AWS conta. Substitua o ARN do recurso de origem e o tipo de registro por valores compatíveis com o serviço que gera os registros. Esses exemplos não exigem uma política de destino de entrega.

Crie uma fonte de entrega para CloudWatch os registros de execução de consultas do Logs Insights

O exemplo a seguir cria uma fonte de entrega para CloudWatch os registros de execução de consultas do Logs Insights. Para obter os requisitos completos `logType` e de recursos do ARN, consulte. PutDeliverySource

> **Note**
>
> O caractere curinga final `log-group:*` é obrigatório porque os registros de execução de consultas não estão vinculados a um grupo de registros. AWS não oferece suporte a um ARN de grupo de log específico para esse tipo de log.

```
aws logs put-delivery-source \
    --name insights-query-logs \
    --resource-arn arn:aws:logs:region:account-id:log-group:* \
    --log-type INSIGHTS_QUERY_LOGS
```

Crie uma fonte de entrega

Crie a fonte de entrega e use um dos exemplos de destino a seguir.

```
aws logs put-delivery-source \
    --name my-delivery-source \
    --resource-arn source-resource-arn \
    --log-type log-type
```

Criar uma entrega para CloudWatch Logs

Crie um destino de entrega para um grupo de registros existente.

```
aws logs put-delivery-destination \
    --name my-cwl-delivery-destination \
    --delivery-destination-configuration \
    "destinationResourceArn=arn:aws:logs:region:account-id:log-group:log-group-name"
```

Crie a entrega.

```
aws logs create-delivery \
    --delivery-source-name my-delivery-source \
    --delivery-destination-arn arn:aws:logs:region:account-id:delivery-destination:my-
cwl-delivery-destination
```

Crie uma entrega para o Amazon S3

Crie um destino de entrega para um bucket existente.

```
aws logs put-delivery-destination \
    --name my-s3-delivery-destination \
    --delivery-destination-configuration \
    "destinationResourceArn=arn:aws:s3:::bucket-name"
```

Crie a entrega.

```
aws logs create-delivery \
    --delivery-source-name my-delivery-source \
    --delivery-destination-arn arn:aws:logs:region:account-id:delivery-destination:my-
s3-delivery-destination
```

Para configurar um prefixo de destino, caminho de sufixo ou Hive-compatible caminho, consulte. Formato de chave de objeto do Amazon S3

Crie uma entrega para Firehose

Crie um destino de entrega para um fluxo DirectPut de entrega existente.

```
aws logs put-delivery-destination \
    --name my-firehose-delivery-destination \
    --delivery-destination-configuration \
    "destinationResourceArn=arn:aws:firehose:region:account-id:deliverystream/delivery-stream-name"
```

Crie a entrega.

```
aws logs create-delivery \
    --delivery-source-name my-delivery-source \
    --delivery-destination-arn arn:aws:logs:region:account-id:delivery-destination:my-firehose-delivery-destination
```

Crie uma entrega para X-Ray

Para rastrear a entrega, use um serviço de origem e um tipo de log que ofereça suporte à entrega para X-Ray. Crie o destino lógico X-Ray de entrega.

```
aws logs put-delivery-destination \
    --name my-xray-delivery-destination \
    --delivery-destination-type XRAY
```

Crie a entrega.

```
aws logs create-delivery \
    --delivery-source-name my-delivery-source \
    --delivery-destination-arn arn:aws:logs:region:account-id:delivery-destination:my-xray-delivery-destination
```

Para verificar a entrega, use o comando a seguir.

```
aws logs describe-deliveries \
    --delivery-source-name-prefix my-delivery-source
```

As seções a seguir fornecem os detalhes das permissões que você precisa ter ao fazer login para configurar a entrega de logs para cada tipo de destino, usando o processo v2. Essas permissões podem ser concedidas a um perfil do IAM com o qual você está conectado.

> **Important**
>
> É sua responsabilidade remover os recursos de entrega de logs após excluir o recurso que gera os logs. Para fazer isso, siga estas etapas.
>
> 1. Exclua o `Delivery` usando a DeleteDeliveryoperação.
> 2. Exclua o `DeliverySource` usando a DeleteDeliverySourceoperação.
> 3. Se o `DeliveryDestination` associado ao `DeliverySource` que você acabou de excluir for usado somente para esse específico`DeliverySource`, você poderá removê-lo usando a DeleteDeliveryDestinationsoperação.

Sumário

- Registros enviados para CloudWatch Logs
  - Permissões de usuário
  - Política de recursos do grupo de logs
- Logs enviados ao Amazon S3
  - Permissões de usuário
  - Política de recursos de bucket do Amazon S3
  - Criptografia no lado do servidor de bucket do Amazon S3
  - Formato de chave de objeto do Amazon S3
- Logs enviados ao Firehose
  - Permissões de usuário
  - Perfis do IAM usados para permissões de recursos
- Traços enviados para X-Ray
  - Permissões de usuário
  - X-Ray política de recursos
  - Habilitar o Transaction Search

Registros enviados para CloudWatch Logs

Para obter um exemplo de AWS CLI, consulte. Criar uma entrega para CloudWatch Logs

Permissões de usuário

Para ativar o envio de CloudWatch registros para o Logs, você precisa estar conectado com as seguintes permissões.

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:*"
            ]
        }
    ]
}
```

Política de recursos do grupo de logs

O grupo de logs para o qual os logs estão sendo enviados deve ter uma política de recursos que contenha determinadas permissões. Se o grupo de registros atualmente não tiver uma política de recursos e o usuário que configura o registro tiver as `logs:DescribeLogGroups` permissões`logs:PutResourcePolicy`,`logs:DescribeResourcePolicies`, e para o grupo de registros, AWS criará automaticamente a política a seguir quando você começar a enviar os CloudWatch registros para o Logs. Para assinaturas recém-criadas, as políticas de recursos são configuradas no nível do grupo de registros e têm um tamanho máximo de 51.200 bytes. Se uma política de recursos existente em nível de conta já conceder permissões por meio de curingas, uma política separada em nível de grupo de registros não seria criada. Para verificar a política de Group-level recursos de log de um grupo de log específico, use o `describe-resource-policies` comando com o `--resource-arn` parâmetro definido como o ARN do grupo de log e o `--policy-scope` parâmetro definido como. `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:*"
                    ]
                }
            }
        }
    ]
}
```

O limite da política de recursos do grupo de registros é de 51.200 bytes. Quando esse limite é atingido, a AWS não pode adicionar novas permissões. Isso exige que os clientes modifiquem manualmente a política para conceder ao principal `delivery.logs.amazonaws.com` serviço permissões sobre as `logs:PutLogEvents` ações `logs:CreateLogStream` e. Os clientes devem usar um prefixo de nome de grupo de registros com curingas, como, `/aws/vendedlogs/*` e usar esse nome de grupo de registros para a criação futura do Delivery.

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:*"
                    ]
                }
            }
        }
    ]
}
```

Logs enviados ao Amazon S3

Para obter um exemplo de AWS CLI, consulte. Crie uma entrega para o Amazon S3

Permissões de usuário

Para permitir o envio de logs ao Amazon S3, é necessário fazer login com as permissões a seguir.

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"
        }
    ]
}
```

Política de recursos de bucket do Amazon S3

O bucket do S3 para o qual os logs estão sendo enviados deve ter uma política de recursos que contenha determinadas permissões. Se o bucket atualmente não tem uma política de recursos e o usuário que está configurando o log tem as permissões `S3:GetBucketPolicy` e `S3:PutBucketPolicy` para o bucket, a AWS cria automaticamente a seguinte política quando você começa a enviar os logs ao 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:*"
```

```
                    ]
                }
            }
        }
    ]
}
```

Na política anterior, para `aws:SourceAccount`, especifique a lista de IDS de conta para os quais os logs estão sendo entregues a esse bucket. Para `aws:SourceArn`, especifique a lista de ARNs do recurso que gera os logs, no formulário `arn:aws:logs:`*`source-region`*`:`*`source-account-id`*`:*`.

Se o bucket tiver uma política de recursos, mas ela não contiver a instrução exibida na política anterior, e o usuário configurando o log tiver as permissões `S3:GetBucketPolicy` e `S3:PutBucketPolicy` para o bucket, essa instrução será anexada à política de recursos do bucket.

> **Note**
>
> Em alguns casos, você pode ver `AccessDenied` erros AWS CloudTrail se a `s3:ListBucket` permissão não tiver sido concedida`delivery.logs.amazonaws.com`. Para evitar esses erros em seus CloudTrail registros, você deve conceder a `s3:ListBucket` permissão `delivery.logs.amazonaws.com` e incluir `Condition` os parâmetros mostrados com o conjunto de `s3:GetBucketAcl` permissões na política de bucket anterior. Para simplificar isso, em vez de criar uma nova `Statement`, você pode atualizar `AWSLogDeliveryAclCheck` diretamente para `“Action”: [“s3:GetBucketAcl”, “s3:ListBucket”]`

Criptografia no lado do servidor de bucket do Amazon S3

Você pode proteger os dados em seu bucket do Amazon S3 habilitando a criptografia no lado do servidor. Você pode usar S3-managed as chaves da Amazon (SSE-S3) ou uma AWS KMS chave armazenada em AWS Key Management Service (SSE-KMS). Para obter mais informações, consulte Proteger dados usando a criptografia no lado do servidor.

Se você escolher SSE-S3, nenhuma configuração adicional será necessária. O Amazon S3 lida com a chave de criptografia.

> **⚠ É necessária uma chave gerenciada pelo cliente**
>
> Se você escolher SSE-KMS, deverá usar uma chave gerenciada pelo cliente. Você não pode usar uma chave AWS gerenciada. Se você configurar a criptografia com uma chave AWS gerenciada, o CloudWatch Logs entregará os registros em um formato ilegível.

Para SSE-KMS, especifique o Amazon Resource Name (ARN) da chave ao ativar a criptografia do bucket. Adicione o seguinte à política de chaves (não à política de bucket do seu bucket do S3), para que a conta de entrega de log possa gravar no seu bucket do S3.

```
{
    "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": ["012345678901"]
        },
        "ArnLike": {
            "aws:SourceArn": ["arn:aws:logs:us-east-1:012345678901:delivery-source:*"]
        }
        }
}
```

Para`aws:SourceAccount`, especifique os IDs da conta cujos registros são entregues a esse bucket. Para`aws:SourceArn`, especifique os ARNs da fonte de entrega no seguinte formato:`arn:aws:logs:`*`source-region`*`:`*`source-account-id`*`:delivery-source:*`.

Formato de chave de objeto do Amazon S3

Para entregas que usam permissões V2, a chave de objeto do Amazon S3 é determinada pelo prefixo de destino, o tipo de log, o caminho do sufixo da entrega e se os caminhos estão habilitados. Hive-compatible O caminho exato definido pelo serviço e as variáveis de sufixo suportadas variam de acordo com o tipo de log.

Prefixo de destino

Um caminho opcional que você anexa ao ARN do bucket ao chamar. PutDeliveryDestination Por exemplo, `.arn:aws:s3:::bucket-name/MyLogPrefix` Os objetos entregues começam com esse prefixo, seguido pelo caminho definido pelo serviço do tipo de registro, como. `AWSLogs/source-account-id/service-name/` Para o registro CloudFront padrão (v2), o prefixo de destino substitui, em vez de preceder, esse caminho padrão.

Caminho do sufixo

Um caminho opcional que você configura para uma entrega individual em seu S3DeliveryConfiguration. Um sufixo pode conter texto estático e variáveis. Para encontrar as variáveis suportadas por um tipo de log, chame DescribeConfigurationTemplatese verifique`allowedSuffixPathFields`. Se você não especificar um caminho de sufixo, o caminho de sufixo padrão do tipo de registro será usado quando houver um disponível.

Hive-compatible caminho

Quando `enableHiveCompatiblePath` é`true`, as variáveis suportadas em um explícito `suffixPath` são renderizadas como. `key=value` Para tipos de registro, como registros de acesso do Application Load Balancer, essa formatação também altera o segmento de conta definido pelo serviço. Por exemplo, `AWSLogs/source-account-id/` torna-se `AWSLogs/aws-account-id=source-account-id/`.

Se você omitir`suffixPath`, a Hive-compatible formatação se aplicará somente quando o caminho do sufixo padrão do tipo de registro for compatível com ela. Por exemplo, os registros de acesso do Application Load Balancer oferecem suporte a essa formatação. Outros tipos de registro, incluindo registros de acesso ao servidor Amazon S3, mantêm seu sufixo padrão original em formato não Hive, incluindo diretórios de recursos e datas.

Para um tipo de registro cujo padrão não oferece suporte à Hive-compatible formatação, especifique um válido `suffixPath` que contenha variáveis compatíveis. Um sufixo explícito substitui o sufixo padrão. Inclua as variáveis de recurso e data de que você precisa, usando o tipo de log `allowedSuffixPathFields` from DescribeConfigurationTemplates.

Os exemplos a seguir mostram o início de uma chave de objeto de log de acesso do Application Load Balancer para a conta em. `111122223333` `us-east-1` Salvo indicação em contrário, os exemplos não pressupõem nenhum prefixo de destino.

Configuração	Início da chave do objeto
Hive-compatible caminho desativado, sufixo omitido	`AWSLogs/111122223333/elasticloadbalancing/us-east-1/2026/09/10/`
Hive-compatible caminho ativado, sufixo omitido	`AWSLogs/aws-account-id=111122223333/elasticloadbalancing/region=us-east-1/year=2026/month=09/day=10/`
Hive-compatible caminho ativado, sufixo `myFolder/{yyyy}/{MM}/{dd}`	`AWSLogs/aws-account-id=111122223333/elasticloadbalancing/myFolder/year=2026/month=09/day=10/`
Hive-compatible caminho ativado, sufixo `myFolder/`	`AWSLogs/111122223333/elasticloadbalancing/myFolder/`
Prefixo de destino`MyLogPrefix` , Hive-compatible caminho desativado, sufixo omitido	`MyLogPrefix/AWSLogs/111122223333/elasticloadbalancing/us-east-1/2026/09/10/`

⚠ Caminhos de sufixo estático e permissões de bucket

Para tipos de registro, como registros de acesso do Application Load Balancer, um `suffixPath` que contém somente texto estático não permite a Hive-compatible formatação do prefixo da conta definido pelo serviço. Por exemplo, `myFolder/` usa o prefixo de conta simples mostrado na tabela. A entrega falhará se o CloudWatch Logs tentar gravar com o prefixo de conta simples e a política de bucket permitir somente o Hive-compatible prefixo. Quando você cria uma entrega do Application Load Balancer com `enableHiveCompatiblePath` definido como`true`, a política de bucket criada automaticamente concede o `s3:PutObject` Hive-compatible prefixo. Inclua pelo menos

uma variável compatível do tipo de registro `allowedSuffixPathFields` em `suffixPath` ou omita `suffixPath` para um tipo de registro cujo caminho de sufixo padrão suporte Hive-compatible formatação.
CloudFront o registro padrão (v2) formata seu prefixo padrão independentemente das variáveis em. `suffixPath`

> **Note**
>
> CloudFront documenta seu comportamento de caminho de registro padrão (v2) e exemplos em Enviar registros para o Amazon S3.

> **Note**
>
> A atualização do caminho ou da Hive-compatible configuração do sufixo afeta as entregas subsequentes após a propagação da configuração. Os objetos existentes não são movidos. Você não pode alterar o ARN de um destino, incluindo seu prefixo, enquanto as entregas ativas fazem referência a esse destino.
> A política de bucket deve considerar `s3:PutObject` o prefixo resultante. Ao gerenciar a política de bucket, mantenha as `aws:SourceAccount` `aws:SourceArn` condições mostradas na política de bucket do Amazon S3 e conceda acesso somente ao prefixo exigido. Política de recursos de bucket do Amazon S3

Logs enviados ao Firehose

Para obter um exemplo de AWS CLI, consulte. Crie uma entrega para Firehose

Permissões de usuário

Para permitir o envio de logs ao Firehose, você primeiro deve fazer login com as permissões a seguir.

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"
        }
    ]
}
```

Perfis do IAM usados para permissões de recursos

Como o Firehose não usa políticas de recursos, AWS usa funções do IAM ao configurar esses registros para serem enviados ao Firehose. AWS cria uma função vinculada ao serviço chamada. AWSServiceRoleForLogDelivery Essa função vinculada ao serviço inclui as permissões a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

Essa função vinculada ao serviço concede permissão para todos os streams de entrega do Firehose que têm a tag definida como. `LogDeliveryEnabled` `true` AWS fornece essa tag ao stream de entrega de destino quando você configura o registro.

Essa função vinculada ao serviço também tem uma política de confiança que permite que a entidade de serviço `delivery.logs.amazonaws.com` assuma a função vinculada ao serviço necessária. Essa política de confiança é a seguinte:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Traços enviados para X-Ray

Para obter um exemplo de AWS CLI, consulte. Crie uma entrega para X-Ray

Permissões de usuário

Para ativar o envio de rastreamentos para AWS X-Ray, você deve estar conectado com as seguintes permissões.

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": "*"
      }
  ]
```

```
}
```

X-Ray política de recursos

A conta de destino para a qual os rastreamentos estão sendo enviados deve ter uma política de recursos que contenha determinadas permissões. Quando o usuário que configura o rastreamento tem `xray:ListResourcePolicies` permissões `xray:PutResourcePolicy` e permissões na conta, cria AWS automaticamente a política de recursos quando você começa a enviar rastreamentos para X-Ray. A política criada depende do serviço de origem:

Amazon Bedrock AgentCore recursos

AWS cria uma política de recursos por tipo de recurso. A política usa padrões curinga com escopo limitado ao limite da conta, abrangendo todos os recursos do mesmo tipo de Amazon Bedrock AgentCore recurso na conta. Por exemplo, se um recurso de Amazon Bedrock AgentCorememória estiver habilitado para entrega de rastreamento, a política cobrirá todos os recursos de memória dessa conta, incluindo quaisquer recursos de memória criados no futuro.

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:*"
        }
      }
    }
  ]
}
```

Outros AWS serviços

Para outros serviços que oferecem suporte à entrega de rastreamento, AWS cria uma política de recursos com escopo específico para o recurso de origem.

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"
        }
      }
    }
  ]
}
```

Habilitar o Transaction Search

Para ativar o envio de rastreamentos para X-Ray, você deve ativar a pesquisa de transações.

Service-specific permissões

Além das permissões específicas do destino listadas nas seções anteriores, alguns serviços exigem autorização explícita para os clientes enviarem logs de seus recursos, como uma camada adicional de segurança. Essa política autoriza a `AllowVendedLogDeliveryForResource` ação de recursos que vendem registros dentro desse serviço. Para esses serviços, use a política a seguir e substitua o namespace do serviço (*`service`*), a Região (*`region`*), o ID da AWS conta (*`account-id`*) e o tipo de recurso (*`resource-type`*) pelos valores apropriados para seu recurso. Nem todo serviço que vende registros usa essa ação. Confirme as permissões e os valores necessários, inclusive se a `AllowVendedLogDeliveryForResource` ação se aplica, na documentação de registros vendidos do serviço ao qual você está concedendo acesso. O exemplo a seguir mostra a política para registros de vendas do 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/*"
        }
    ]
}
```

Console-specific permissões

Além das permissões listadas nas seções anteriores, se você for configurar a entrega de logs usando o console em vez de APIs, precisará das seguintes permissões adicionais:

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": [
                "*"
            ]
        }
    ]
}
```

Cross-account exemplo de entrega

Neste exemplo, duas contas estão envolvidas. A conta com o recurso gerador de registros é Conta A, ID:*123456789012*, e a conta com o recurso que consome registros é Conta B, ID:. *111122223333*

A conta A deseja fornecer registros da base de Amazon Bedrock conhecimento em sua conta com o ARN arn:aws:bedrock: ::knowledge-base/. *us-east-1 123456789012 kb-12345678*

Neste exemplo, a conta A precisa das seguintes permissões:

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:*"
            ]
        }
    ]
}
```

Criar origem de entrega

Para começar, a conta A cria uma origem de entrega com sua base de conhecimentos do Bedrock:

```
aws logs put-delivery-source --name my-delivery-source --log-type APPLICATION_LOGS --
resource-arn arn:aws:bedrock:region:AAAAAAAAAAAA:knowledge-base/XXXXXXXXXX
```

Em seguida, a conta B deve criar o destino da entrega usando um dos fluxos abaixo:

- Configurar entrega para um bucket do Amazon S3
- Configurar entrega em um fluxo do Firehose

Configurar entrega para um bucket do Amazon S3

A conta B deseja receber os logs no bucket do S3 com o ARN arn:aws:s3: ::amzn-s3-demo-bucket. Neste exemplo, a conta B precisará das seguintes permissões:

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:*"
        }
    ]
}
```

O bucket precisará das seguintes permissões em sua política de 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"
                    ]
                }
            }
        }
    ]
}
```

Se o bucket estiver criptografado com SSE-KMS, certifique-se de que a política de AWS KMS chaves tenha as permissões apropriadas. Por exemplo, se a chave do KMS for `arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`, use o seguinte:

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"
                    ]
                }
            }
        }
    ]
}
```

A conta B pode então criar um destino de entrega tendo o bucket do S3 como recurso de destino:

```
aws logs put-delivery-destination --name my-s3-delivery-destination --delivery-
destination-configuration "destinationResourceArn=arn:aws:s3:::amzn-s3-demo-bucket"
```

Em seguida, a conta B cria uma política de destino de entrega em seu destino de entrega recém-criado, o que dará à conta A permissão para criar uma entrega de logs. A política que será adicionada ao destino de entregas recém-criado é a seguinte:

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"
        }
    ]
}
```

Essa política será salva no computador da conta B como `destination-policy-s3.json`. Para anexar esse recurso, a conta B executará o seguinte comando:

```
aws logs put-delivery-destination-policy --delivery-destination-name my-s3-delivery-
destination --delivery-destination-policy file://destination-policy-s3.json
```

Por fim, a conta A cria a entrega, que vincula a origem da entrega na conta A ao destino da entrega na conta 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
```

Configurar entrega em um fluxo do Firehose

Neste exemplo, a conta B quer receber logs em seu fluxo do Firehose. O stream do Firehose tem o seguinte ARN e está configurado para usar o tipo de stream de entrega: DirectPut

`arn:aws:firehose:us-east-1:111122223333:deliverystream/log-delivery-stream`

Neste exemplo, a conta B precisa das seguintes permissões:

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:*"
        }
    ]
}
```

O fluxo do Firehose deve ter a tag `LogDeliveryEnabled` definida como `true`.

A conta B pode então criar um destino de entrega tendo o fluxo do Firehose como recurso de destino:

```
aws logs put-delivery-destination --name my-fh-delivery-destination --delivery-
destination-configuration
 "destinationResourceArn=arn:aws:firehose:region:BBBBBBBBBBBB:deliverystream/X"
```

Em seguida, a conta B cria uma política de destino de entrega em seu destino de entrega recém-criado, o que dará à conta A permissão para criar uma entrega de logs. A política que será adicionada ao destino de entrega recém-criado é a seguinte:

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"
        }
    ]
}
```

Essa política será salva no computador da conta B como `destination-policy-fh.json`. Para anexar esse recurso, a conta B executa o seguinte comando:

```
aws logs put-delivery-destination-policy --delivery-destination-name my-fh-delivery-
destination --delivery-destination-policy file://destination-policy-fh.json
```

Por fim, a conta A cria a entrega, que vincula a origem da entrega na conta A ao destino da entrega na conta 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
```

Cross-service prevenção delegada confusa

O problema do “confused deputy” é um problema de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executar a ação. Em AWS, a falsificação de identidade entre serviços pode resultar em um problema confuso de delegado. Cross-service a representação pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado de modo a usar suas permissões para atuar nos recursos de outro cliente de uma forma na qual ele não deveria ter permissão para acessar. Para evitar isso, AWS fornece ferramentas que ajudam você a proteger seus dados para todos os serviços com diretores de serviços que receberam acesso aos recursos em sua conta.

Recomendamos usar as chaves de contexto de condição `aws:SourceArn`aws:SourceAccountaws:SourceOrgID,, e `aws:SourceOrgPaths`global nas políticas de recursos para limitar as permissões que o CloudWatch Logs concede a outro serviço ao recurso. Use `aws:SourceArn` se quiser associar apenas um recurso ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços. Use `aws:SourceOrgID` se quiser permitir que qualquer recurso de qualquer conta de uma organização seja associado ao uso entre serviços. Use `aws:SourceOrgPaths` para associar qualquer recurso de contas em um AWS Organizations caminho ao uso entre serviços. Para obter mais informações sobre como usar e entender os caminhos, consulte Compreender o caminho da AWS Organizations entidade.

A maneira mais eficaz de se proteger contra o problema do substituto confuso é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso. Se você não souber o ARN completo do recurso ou especificar vários recursos, use a chave de condição de contexto global `aws:SourceArn` com caracteres curinga (`*`) para as partes desconhecidas do ARN. Por exemplo, .`arn:aws:`*servicename*`:*:`*123456789012*`:*`

Se o valor do `aws:SourceArn` não contiver o ID da conta, como um ARN de bucket do Amazon S3, você deverá usar ambos, a `aws:SourceAccount` e o `aws:SourceArn` para limitar as permissões.

Para se proteger do problema de "confused deputy" em grande escala, use a chave de contexto de condição global `aws:SourceOrgID` ou `aws:SourceOrgPaths` com o ID ou o caminho da

organização do recurso nas políticas baseadas em recursos. As políticas que incluem a chave `aws:SourceOrgID` ou `aws:SourceOrgPaths` incluem automaticamente as contas corretas e você não tem que atualizar manualmente as políticas quando adiciona, remove ou move contas na organização.

As políticas documentadas para conceder acesso aos CloudWatch Logs para gravar dados no Kinesis Data Streams e Etapa 1: criar um destino no Firehose Etapa 2: Criar um destino e mostram como você pode usar a chave de contexto de condição global para ajudar `aws:SourceArn` a evitar o confuso problema secundário.

Automatize a ativação de registros com regras de ativação de telemetria

Você pode usar as regras de ativação de telemetria para configurar automaticamente a coleta de registros para seus recursos. AWS As regras ajudam você a padronizar a coleta de registros em sua organização ou contas e a garantir uma cobertura de monitoramento consistente.

As regras de capacitação permitem que você:

- Ative automaticamente o registro de recursos novos e existentes que correspondam ao escopo da sua regra
- Aplique regras no nível da organização, unidade organizacional (OU) ou conta individual
- Filtre quais recursos uma regra afeta usando tags

Atualmente, as regras de habilitação oferecem suporte às seguintes AWS fontes de telemetria: Amazon VPC Flow Logs, Logs, Route 53 Resolver Query AWS WAF Logs, NLB Access Logs, Amazon EKS Control Plane Logs, Data CloudTrail and Management Events, Amazon Bedrock Logs, Amazon AgentCore EC2 Detailed Metrics, AWS Security Hub, Amazon Bedrock Gateway, Amazon AgentCore Bedrock Memory e Distribution. AgentCore CloudFront

Para obter detalhes completos sobre as regras de habilitação, incluindo comportamento de regras, gerenciamento de regras, solução de problemas e considerações específicas do serviço, consulte Regras de habilitação de telemetria no Guia do usuário da Amazon. CloudWatch

Hierarquia de avaliação das regras

As regras de capacitação são avaliadas hierarquicamente: primeiro as regras organizacionais, depois as regras e, em seguida, as OU-level regras no nível da conta. As regras em níveis mais altos

fornecem a telemetria básica. Lower-level as regras podem adicionar telemetria adicional, mas não podem reduzi-la. Se existirem regras conflitantes no mesmo escopo, nenhuma será aplicada até que o conflito seja resolvido.

Criação de uma regra de capacitação

Para criar uma regra de habilitação de telemetria

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Configuração de telemetria.
3. Escolha a guia Regras de habilitação.
4. Escolha Adicionar regra.
5. Especifique o nome da regra, o escopo (organização, OU ou conta), a fonte de dados e o tipo de telemetria.
6. Opcionalmente, adicione tags para filtrar quais recursos a regra afeta.
7. Selecione Criar regra.

Ativar o registro de fontes de terceiros

CloudWatch O Logs suporta a ingestão de registros de fontes de terceiros por meio de integrações diretas de API e integrações de buckets do Amazon S3. Você também pode receber descobertas adicionais de segurança de terceiros por meio do AWS Security Hub CSPM.

Integrações diretas de terceiros

CloudWatch O Logs fornece integrações diretas com as seguintes fontes terceirizadas. Essas integrações usam conexões diretas de API ou integrações de bucket do Amazon S3 para ingerir registros em registros: CloudWatch

- CrowdStrike Falcão
- Microsoft Office 365
- Okta Auth0
- Microsoft Entra ID
- NGFW da Palo Alto Networks
- Microsoft Windows Events
- Wiz
- Zscaler Internet Access
- Okta SSO
- SentinelOne Segurança de terminais
- GitHub Registros de auditoria
- ServiceNow CMDB
- Cisco Umbrella

Para obter instruções de configuração, consulte o guia de configuração de integração de terceiros no Guia CloudWatch do usuário da Amazon.

Fontes adicionais via AWS Security Hub CSPM

Além das integrações diretas, parceiros de segurança terceirizados enviam as descobertas para o CSPM do AWS Security Hub, que são então disponibilizadas como fontes de dados no Logs.

CloudWatch A tabela a seguir lista as integrações de parceiros CSPM do Security Hub e seu tipo de integração.

Para habilitar as descobertas do CSPM do Security Hub como fonte de dados no CloudWatch Logs, crie uma regra de habilitação de telemetria para o Security AWS Hub no console. CloudWatch A regra de habilitação é configurada para CloudWatch ingerir automaticamente as descobertas do CSPM do Security Hub em um grupo de registros gerenciados. Para step-by-step obter instruções, consulte as regras de habilitação de telemetria no Guia do usuário da Amazon. CloudWatch

Parceiro	Integração
3 CORESec — NTA	Envia descobertas por meio do Security Hub CSPM
Lógica de alertas — Gerenciamento de SIEMless ameaças	Envia descobertas por meio do Security Hub CSPM
Aqua Security — Plataforma de segurança nativa em nuvem	Envia descobertas por meio do Security Hub CSPM
Aqua Security — Kube-bench	Envia descobertas por meio do Security Hub CSPM
Armadura — Armadura em qualquer lugar	Envia descobertas por meio do Security Hub CSPM
AttackIQ	Envia descobertas por meio do Security Hub CSPM
Barracuda Networks — Guardião da segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
BigID — BigID Enterprise	Envia descobertas por meio do Security Hub CSPM
Hexágono azul	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Check Point — CloudGuard IaaS	Envia descobertas por meio do Security Hub CSPM
Check Point — CloudGuard Gestão da postura	Envia descobertas por meio do Security Hub CSPM
Clarity — Dome	Envia descobertas por meio do Security Hub CSPM
Segurança de armazenamento em nuvem — Antivírus para Amazon S3	Envia descobertas por meio do Security Hub CSPM
Segurança de contraste — avaliação de contraste	Envia descobertas por meio do Security Hub CSPM
CrowdStrike — CrowdStrike Falcão	Envia descobertas por meio do Security Hub CSPM
CyberArk — Análise privilegiada de ameaças	Envia descobertas por meio do Security Hub CSPM
Teorema dos dados	Envia descobertas por meio do Security Hub CSPM
Drata	Envia descobertas por meio do Security Hub CSPM
Forcepoint — CASB	Envia descobertas por meio do Security Hub CSPM
Forcepoint — Gateway de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
Forcepoint — DLP	Envia descobertas por meio do Security Hub CSPM
Forcepoint — NGFW	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Fuga	Envia descobertas por meio do Security Hub CSPM
Guardicore — Central	Envia descobertas por meio do Security Hub CSPM
HackerOne — Inteligência de vulnerabilidade	Envia descobertas por meio do Security Hub CSPM
JFrog — Raio X	Envia descobertas por meio do Security Hub CSPM
Juniper Networks — Firewall vSRX de próxima geração	Envia descobertas por meio do Security Hub CSPM
k9 Security — Analisador de acesso	Envia descobertas por meio do Security Hub CSPM
Rendas	Envia descobertas por meio do Security Hub CSPM
McAfee — MVISION CNAPP	Envia descobertas por meio do Security Hub CSPM
NETSCOUT — Investigador cibernético	Envia descobertas por meio do Security Hub CSPM
Orca — Plataforma de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM
Palo Alto Networks — Computação em nuvem Prisma	Envia descobertas por meio do Security Hub CSPM
Palo Alto Networks — Prisma Cloud Enterprise	Envia descobertas por meio do Security Hub CSPM
Plerion — Plataforma de segurança na nuvem	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Gatuno	Envia descobertas por meio do Security Hub CSPM
Qualys — Gerenciamento de vulnerabilidades	Envia descobertas por meio do Security Hub CSPM
Rapid7 — InsightVM	Envia descobertas por meio do Security Hub CSPM
SentinelOne	Envia descobertas por meio do Security Hub CSPM
Snyk	Envia descobertas por meio do Security Hub CSPM
Segurança Sonrai — Sonrai Dig	Envia descobertas por meio do Security Hub CSPM
Sophos — Proteção de servidor	Envia descobertas por meio do Security Hub CSPM
StackRox — Segurança do Kubernetes	Envia descobertas por meio do Security Hub CSPM
Sumo Logic — Análise de dados de máquinas	Envia descobertas por meio do Security Hub CSPM
Symantec — Proteção da carga de trabalho na nuvem	Envia descobertas por meio do Security Hub CSPM
Tenable.io	Envia descobertas por meio do Security Hub CSPM
Trend Micro: Cloud One	Envia descobertas por meio do Security Hub CSPM
Vectra — Cognito Detect	Envia descobertas por meio do Security Hub CSPM

Parceiro	Integração
Wiz	Envia descobertas por meio do Security Hub CSPM
Caveonix — Caveonix Cloud	Envia e recebe descobertas por meio do Security Hub CSPM
Cloud Custodian	Envia e recebe descobertas por meio do Security Hub CSPM
DisruptOps	Envia e recebe descobertas por meio do Security Hub CSPM
Leão	Envia e recebe descobertas por meio do Security Hub CSPM
Turbot	Envia e recebe descobertas por meio do Security Hub CSPM

Note

Essa lista reflete as integrações de parceiros do Security Hub que enviam descobertas no momento da redação. Como o AWS Security Hub adiciona regularmente novas integrações de parceiros, consulte Integrações de produtos de terceiros com o CSPM do Security Hub no Guia do Usuário do AWS Security Hub para obter a maior up-to-date lista de parceiros disponíveis.

Exportar dados de log para o Amazon S3

Este capítulo fornece informações para que você possa exportar dados de log dos grupos de logs para um bucket do Amazon S3 para processamento personalizado e análise ou para carregar em outros sistemas. Você pode exportar para um bucket do S3 na mesma conta ou em uma conta diferente.

Você pode fazer o seguinte:

- Exporte dados de log para buckets do S3 que são criptografados por SSE-KMS in AWS Key Management Service ()AWS KMS
- Exportar dados de log para buckets do S3 com o S3 Object Lock ativado com um período de retenção

Recomendamos não exportar regularmente para o Amazon S3 como forma de arquivar continuamente seus logs. Para esse caso de uso, recomendamos que você use assinaturas. Para obter mais informações sobre assinaturas, consulte Real-time processamento de dados de registro com assinaturas.

Para iniciar o processo de exportação, você deve criar um bucket do S3 para armazenar os dados de log exportados. Você pode armazenar os arquivos exportados em seu bucket do Amazon S3 e definir regras de ciclo de vida do S3 para arquivar ou excluir arquivos exportados automaticamente.

Você pode exportar para buckets do S3 criptografados com AES-256 ou com. SSE-KMS A exportação para buckets criptografados com não DSSE-KMS é suportada.

Você pode exportar logs de vários grupos de log ou vários intervalos de tempo para o mesmo bucket do S3. Para organizar os dados exportados, especifique um prefixo para cada tarefa de exportação que será usado como o prefixo-chave do Amazon S3 para todos os objetos exportados. Por exemplo, `prod/app-logs/2026-01-03/` ou `log-group-name/backup/`

> **Note**
>
> Time-based a classificação em partes dos dados de log dentro de um arquivo exportado não é garantida. Você pode classificar os dados exportados do campo de log usando os utilitários do Linux. Por exemplo, o comando de utilitário a seguir classifica os eventos em todos os arquivos `.gz` em uma única pasta.

```
find . -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
```

O comando de utilitário a seguir classifica os arquivos .gz de várias subpastas.

```
find ./*/ -type f -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
```

Além disso, você pode usar outro comando `stdout` para canalizar a saída classificada para outro arquivo e salvá-la.

Pode levar até 12 horas para os dados de log se tornarem disponíveis para exportação. O tempo limite para exportar de dados de log é de 24 horas. Se suas tarefas de exportação estiverem atingindo o tempo limite, reduza o intervalo de tempo quando criar a tarefa de exportação.

Para obter informações sobre análise quase em tempo real de dados de log, consulte Análise de dados de registro com o CloudWatch Logs Insights ou Real-time processamento de dados de registro com assinaturas.

Conteúdo

- Conceitos
- Exportação de dados de log para o Amazon S3 usando o console.
- Exporte dados de log para o Amazon S3 usando o AWS CLI
- Descrever as tarefas de exportação (CLI)
- Cancelar uma tarefa de exportação (CLI)

Conceitos

Antes de começar, familiarize-se com os seguintes conceitos de exportação:

nome do grupo de logs

O nome do grupo de logs associado a uma tarefa de exportação. Os dados de log neste grupo serão exportados para o bucket especificado do S3.

de (timestamp)

Um timestamp obrigatório expresso como o número de milissegundos desde 1º de janeiro de 1970 00:00:00 UTC. Todos os eventos de log do grupo de logs que foram ingeridos a partir dessa hora serão exportados.

a (timestamp)

Um timestamp obrigatório expresso como o número de milissegundos desde 1º de janeiro de 1970 00:00:00 UTC. Todos os eventos de log no grupo de logs que foram ingeridos antes desse período serão exportados.

bucket de destino

O nome do bucket do S3 associado a uma tarefa de exportação. Esse bucket é usado para exportar os dados de log do grupo de logs especificado.

prefixo de destino

Um atributo opcional que é usado como o prefixo de chave do Amazon S3 para todos os objetos exportados. Isso ajuda a criar uma organização do tipo pasta em seu bucket.

Exportação de dados de log para o Amazon S3 usando o console.

Nos exemplos a seguir, você usa o CloudWatch console da Amazon para exportar todos os dados de um grupo de CloudWatch logs do Amazon Logs chamado `my-log-group` para um bucket do Amazon S3 chamado. `amzn-s3-demo-bucket`

Há suporte para a exportação de dados de log para buckets do S3 que são criptografados pelo SSE-KMS . A exportação para buckets criptografados com não DSSE-KMS é suportada.

Os detalhes de como configurar a exportação dependem se o bucket do Amazon S3 para o qual você deseja exportar está na mesma conta que os logs que estão sendo exportados ou em uma conta diferente.

Tópicos

- Same-account exportação (console)
- Cross-account exportação (console)

Same-account exportação (console)

Se o bucket do Amazon S3 estiver na mesma conta dos logs que estão sendo exportados, use as instruções nesta seção.

Tópicos

- Crie um bucket do Amazon S3 (console)
- Configurar permissões de acesso (console)
- Definir permissões em um bucket do Amazon S3 (console)
- (Opcional) Exportação para um bucket Amazon S3 de destino criptografado SSE-KMS com (console)
- Criar uma tarefa de exportação (console)

Crie um bucket do Amazon S3 (console)

Recomendamos que você use um bucket criado especificamente para o CloudWatch Logs. No entanto, se você desejar usar um bucket existente, vá para a etapa 2.

> **Note**
>
> O bucket do Amazon S3 deve residir na mesma região que os dados de log a serem exportados. CloudWatch O Logs não suporta a exportação de dados para buckets do Amazon S3 em uma região diferente.

Como criar um bucket do Amazon S3

1. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Se necessário, altere a região da . Na barra de navegação, escolha a região em que seus CloudWatch registros residem.
3. Escolha Criar bucket.
4. Para Bucket Name (Nome do bucket), digite um nome para o bucket.
5. Em Região, selecione a região em que seus dados de CloudWatch registros residem.
6. Escolha Criar.

Configurar permissões de acesso (console)

Para criar a tarefa de exportação, você precisará estar conectado com a função do `AmazonS3ReadOnlyAccess` IAM e com as seguintes permissões:

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Definir permissões em um bucket do Amazon S3 (console)

Por padrão, todos os buckets e objetos do Amazon S3 são privados. Somente o proprietário do recurso, Conta da AWS aquele que criou o bucket, pode acessar o bucket e quaisquer objetos que ele contenha. No entanto, o proprietário do recurso pode optar por conceder permissões de acesso a outros recursos e usuários ao criar uma política de acesso padrão.

Quando você define a política, é recomendável incluir uma string gerada aleatoriamente como o prefixo para o bucket, para que apenas os streams de log desejados sejam exportados para o bucket.

> **Important**
>
> Para tornar as exportações para buckets do Amazon S3 mais seguras, agora exigimos que você especifique a lista de contas de origem que têm permissão para exportar dados de log para seu bucket do S3.
> No exemplo a seguir, a lista de IDs de conta na chave da `aws:SourceAccount` seriam as contas das quais um usuário pode exportar dados de log para o bucket do Amazon S3.
> A chave `aws:SourceArn` seria o recurso para o qual a ação está sendo realizada. Você pode restringir isso a um grupo de logs específico ou usar um curinga, como mostrado neste exemplo.
> Recomendamos que você inclua também o ID da conta na qual o bucket do S3 foi criado para permitir a exportação dentro da mesma conta.

Para definir permissões em um bucket do Amazon S3

1. No console do Amazon S3, escolha o bucket que você criou.
2. Escolha Permissions (Permissões), Bucket policy (Política de bucket).
3. No Bucket Policy Editor (Editor de política do bucket), adicione a política a seguir. Altere `amzn-s3-demo-bucket` para o nome do bucket do S3. Certifique-se de especificar o endpoint correto da região como `us-west-1` para a Entidade 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. Escolha Salvar para definir a política que você acabou de adicionar como política de acesso em seu bucket. Essa política permite que a CloudWatch Logs exporte dados de log para seu bucket

do Amazon S3. O proprietário do bucket tem permissões completas sobre todos os objetos exportados.

> **Warning**
>
> Se o bucket existente já tiver uma ou mais políticas anexadas, adicione as instruções de acesso de CloudWatch registros a essa política ou políticas. Recomendamos avaliar o conjunto resultante de permissões para ter certeza de que elas são apropriadas para os usuários que acessarão o bucket.

(Opcional) Exportação para um bucket Amazon S3 de destino criptografado SSE-KMS com (console)

Essa etapa é necessária somente se você estiver exportando para um bucket do Amazon S3 que usa criptografia do lado do servidor com. AWS KMS keys Essa criptografia é conhecida como SSE-KMS.

Para exportar para um bucket criptografado com SSE-KMS

1. Abra o AWS KMS console em https://console.aws.amazon.com/kms.
2. Para alterar o Região da AWS, use o seletor de região no canto superior direito da página.
3. Na barra de navegação esquerdo, escolha Customer managed keys (Chaves gerenciadas pelo cliente).

   Escolha Create Key (Criar chave).
4. Para Key type (Tipo de chave), escolha Symmetric (Simétrica).
5. Em Key usage (Uso da chave), escolha Encrypt and decrypt (Criptografar e descriptografar) e, em seguida, escolha Next (Avançar).
6. Em Add labels (Adicionar rótulos), insira um alias para a chave e, opcionalmente, adicione uma descrição ou tags. Escolha Próximo.
7. Em Key administrators (Administradores de chaves), selecione quem pode administrar essa chave e escolha Next (Avançar).
8. Em Define key usage permissions (Definir permissões de uso da chave), não faça alterações e escolha Next (Avançar).
9. Revise as configurações e escolha Finish (Concluir).

10. De volta à página Customer managed keys (Chaves gerenciadas pelo cliente), escolha o nome da chave que você acabou de criar.
11. Na guia Key Policy (Política de chaves), selecione Switch to policy view (Alternar para visualização de política).
12. Na seção Key policy (Política de chaves), escolha Edit (Editar).
13. Adicione a declaração a seguir à lista de declarações de política de chaves. Ao fazer isso, *Region* substitua pela região dos seus registros e *account-ARN* substitua pelo ARN da conta que possui a chave 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. Escolha Salvar alterações.
15. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
16. Encontre o bucket que você criou no Crie um bucket S3 (CLI) e escolha o nome do bucket.
17. Escolha a guia Properties (Propriedades). Em seguida, em Default encryption (Criptografia padrão), escolha Edit (Editar).
18. Em Server-side Criptografia, escolha Habilitar.
19. Em Tipo de criptografia, escolha AWS Key Management Service chave (SSE-KMS).
20. Escolha Escolher entre suas AWS KMS chaves e encontre a chave que você criou.
21. Para Bucket key (Chave do bucket), escolha Enable (Habilitar).
22. Escolha Salvar alterações.

Criar uma tarefa de exportação (console)

Neste procedimento, você cria a tarefa de exportação para exportar registros de um grupo de registros.

Para exportar dados para o Amazon S3 usando o console CloudWatch

1. Faça login com permissões suficientes, conforme documentado em Configurar permissões de acesso (console).
2. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
3. No painel de navegação, escolha Grupos de logs.
4. Na tela Grupos de logs, escolha o nome do grupo de logs.
5. Escolha Actions (Ações), Export to Amazon S3 (Exportar para o Amazon S3).
6. Na tela Export data to Amazon S3 (Exportar dados para o Amazon S3), em Define data export (Definir exportação de dados), defina o período dos dados a serem exportados usando From (De) e To (Até).
7. Se o seu grupo de logs tiver vários streams de log, você poderá fornecer um prefixo de stream de logs para limitar os dados do grupo de logs para um stream específico. Escolha Advanced (Avançado) e, depois, em Stream prefix (Prefixo do stream), digite o prefixo do stream de logs.
8. Em Escolher bucket do S3, escolha a conta associada ao bucket do S3.

9. Em S3 bucket name (Nome do bucket do S3), escolha um bucket do Amazon S3.
10. Em Prefixo do bucket do S3, insira a string gerada aleatoriamente que você especificou na política do bucket.
11. Escolha Export (Exportar) para exportar seus dados de log para o Amazon S3.
12. Para visualizar o status dos dados de log exportados para o Amazon S3, escolha Actions (Ações), View all exports to Amazon S3 (Visualizar todas as exportações para o Amazon S3).

Cross-account exportação (console)

Se o bucket do Amazon S3 estiver em uma conta diferente da conta dos logs que estão sendo exportados, use as instruções nesta seção.

Tópicos

- Crie um bucket do Amazon S3 para exportação entre contas (console)
- Configurar permissões de acesso para exportação entre contas (console)
- Defina permissões em um bucket do S3 para exportação entre contas (console)
- (Opcional) Exportação para um bucket Amazon S3 de destino SSE-KMS criptografado para exportação entre contas (console)
- Crie uma tarefa de exportação para exportação entre contas (console)

Crie um bucket do Amazon S3 para exportação entre contas (console)

Recomendamos que você use um bucket criado especificamente para o CloudWatch Logs. No entanto, se quiser usar um bucket existente, você pode pular esse procedimento.

> **Note**
>
> O bucket do Amazon S3 deve residir na mesma região que os dados de log a serem exportados. CloudWatch O Logs não suporta a exportação de dados para buckets do Amazon S3 em uma região diferente.

Como criar um bucket do Amazon S3

1. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.

2. Se necessário, altere a região da . Na barra de navegação, escolha a região em que seus CloudWatch registros residem.
3. Escolha Criar bucket.
4. Para Bucket Name (Nome do bucket), digite um nome para o bucket.
5. Em Região, selecione a região em que seus dados de CloudWatch registros residem.
6. Escolha Criar.

Configurar permissões de acesso para exportação entre contas (console)

Primeiro, você deve criar uma nova política do IAM para permitir que CloudWatch os Logs tenham a `s3:PutObject` ação para o bucket Amazon S3 de destino na conta de destino.

Além da `s3:PutObject` ação, as ações adicionais incluídas na política dependem se o bucket de destino usa AWS KMS criptografia ou se as ACLs estão habilitadas usando a configuração de propriedade de objetos do S3.

- Se estiver usando a criptografia KMS, adicione as ações `kms:GenerateDataKey` e `kms:Decrypt` ao recurso chave
- Se as ACLs estiverem habilitadas no bucket, adicione a ação `s3:PutObjectAcl` ao recurso do bucket.

Altere `amzn-s3-demo-bucket` para o nome do seu bucket S3 de destino nas políticas a seguir.

Para criar uma política do IAM para exportar registros para um bucket do Amazon S3

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Selecione Criar política.
4. Na seção Editor de políticas, escolha JSON.
5. Se o bucket de destino não usar AWS KMS criptografia, cole a política a seguir no editor.

   JSON

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

```
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

Se o bucket de destino usar AWS KMS criptografia, cole a política a seguir no 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"
    }
  ]
}
```

Se as ACLs estiverem habilitadas no bucket de destino, adicione s3: PutObjectAcl a s3: PutObject Action block nas políticas acima.

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:PutObject",
               "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

6. Escolha Próximo.
7. Insira um nome de política. Você usará esse nome para vincular a política ao seu perfil do IAM.
8. Escolha Criar política para salvar a nova política.

Para criar uma tarefa de exportação, você deve estar conectado com uma função do IAM que tenha a política `AmazonS3ReadOnlyAccess` gerenciada anexada, a política do IAM criada acima e também com as seguintes permissões:

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .

- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.

- Usuários do IAM:

- Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
- (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Defina permissões em um bucket do S3 para exportação entre contas (console)

Por padrão, todos os buckets e objetos do S3 são privados. Somente o proprietário do recurso, Conta da AWS aquele que criou o bucket, pode acessar o bucket e quaisquer objetos que ele contenha. No entanto, o proprietário do recurso pode optar por conceder permissões de acesso a outros recursos e usuários ao criar uma política de acesso padrão.

Quando você define a política, é recomendável incluir uma string gerada aleatoriamente como o prefixo para o bucket, para que apenas os streams de log desejados sejam exportados para o bucket.

> **Important**
>
> Para tornar as exportações para buckets S3 mais seguras, agora exigimos que você especifique a lista de contas de origem que têm permissão para exportar dados de log para seu bucket S3.
> No exemplo a seguir, a lista de IDs de conta na chave da `aws:SourceAccount` seriam as contas das quais um usuário pode exportar dados de log para o bucket do S3. A chave `aws:SourceArn` seria o recurso para o qual a ação está sendo realizada. Você pode restringir isso a um grupo de logs específico ou usar um curinga, como mostrado neste exemplo.
> Recomendamos que você inclua também o ID da conta na qual o bucket do S3 foi criado para permitir a exportação dentro da mesma conta.

Para definir permissões em um bucket do Amazon S3

1. No console do Amazon S3, escolha o bucket que você criou.
2. Escolha Permissions (Permissões), Bucket policy (Política de bucket).

3. No Bucket Policy Editor (Editor de política do bucket), adicione a política a seguir. Altere `amzn-s3-demo-bucket` para o nome do bucket do S3. Certifique-se de especificar o endpoint correto da região como `us-east-1` para a Entidade 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. Escolha Salvar para definir a política que você acabou de adicionar como política de acesso em seu bucket. Essa política permite que o CloudWatch Logs exporte dados de log para seu bucket do S3. O proprietário do bucket tem permissões completas sobre todos os objetos exportados.

> **Warning**
>
> Se o bucket existente já tiver uma ou mais políticas anexadas, adicione as instruções de acesso de CloudWatch registros a essa política ou políticas. Recomendamos avaliar o conjunto resultante de permissões para ter certeza de que elas são apropriadas para os usuários que acessarão o bucket.

(Opcional) Exportação para um bucket Amazon S3 de destino SSE-KMS criptografado para exportação entre contas (console)

Esse procedimento é necessário somente se você estiver exportando para um bucket do S3 que usa criptografia do lado do servidor com. AWS KMS keys Essa criptografia é conhecida como SSE-KMS.

Para exportar para um bucket criptografado com SSE-KMS

1. Abra o AWS KMS console em https://console.aws.amazon.com/kms.
2. Para alterar o Região da AWS, use o seletor de região no canto superior direito da página.
3. Na barra de navegação esquerdo, escolha Customer managed keys (Chaves gerenciadas pelo cliente).

   Escolha Create Key (Criar chave).
4. Para Key type (Tipo de chave), escolha Symmetric (Simétrica).
5. Em Key usage (Uso da chave), escolha Encrypt and decrypt (Criptografar e descriptografar) e, em seguida, escolha Next (Avançar).
6. Em Add labels (Adicionar rótulos), insira um alias para a chave e, opcionalmente, adicione uma descrição ou tags. Escolha Próximo.
7. Em Key administrators (Administradores de chaves), selecione quem pode administrar essa chave e escolha Next (Avançar).
8. Em Define key usage permissions (Definir permissões de uso da chave), não faça alterações e escolha Next (Avançar).
9. Revise as configurações e escolha Finish (Concluir).
10. De volta à página Customer managed keys (Chaves gerenciadas pelo cliente), escolha o nome da chave que você acabou de criar.
11. Na guia Key Policy (Política de chaves), selecione Switch to policy view (Alternar para visualização de política).
12. Na seção Key policy (Política de chaves), escolha Edit (Editar).
13. Adicione a declaração a seguir à lista de declarações de política de chaves. Ao fazer isso, *us-east-1* substitua pela região dos seus registros, *account-ARN* pelo ARN da conta que possui a chave KMS, *123456789012* pelo número da conta que possui a chave KMS, *key_id* pelo ID da chave kms-e pela função usada para criar a tarefa *role_name* de exportação.

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. Escolha Salvar alterações.
15. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
16. Encontre o bucket que você criou no Crie um bucket S3 (CLI) e escolha o nome do bucket.
17. Escolha a guia Properties (Propriedades). Em seguida, em Default encryption (Criptografia padrão), escolha Edit (Editar).
18. Em Server-side Criptografia, escolha Habilitar.
19. Em Tipo de criptografia, escolha AWS Key Management Service chave (SSE-KMS).
20. Escolha Escolher entre suas AWS KMS chaves e encontre a chave que você criou.
21. Para Bucket key (Chave do bucket), escolha Enable (Habilitar).
22. Escolha Salvar alterações.

Crie uma tarefa de exportação para exportação entre contas (console)

Neste procedimento, você cria a tarefa de exportação para exportar registros de um grupo de registros.

Para exportar dados para o Amazon S3 usando o console CloudWatch

1. Faça login com permissões suficientes, conforme documentado em Configurar permissões de acesso (console).
2. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
3. No painel de navegação, escolha Grupos de logs.
4. Na tela Grupos de logs, escolha o nome do grupo de logs.
5. Escolha Actions (Ações), Export to Amazon S3 (Exportar para o Amazon S3).
6. Na tela Export data to Amazon S3 (Exportar dados para o Amazon S3), em Define data export (Definir exportação de dados), defina o período dos dados a serem exportados usando From (De) e To (Até).

7. Se o seu grupo de logs tiver vários streams de log, você poderá fornecer um prefixo de stream de logs para limitar os dados do grupo de logs para um stream específico. Escolha Advanced (Avançado) e, depois, em Stream prefix (Prefixo do stream), digite o prefixo do stream de logs.
8. Em Escolher bucket do S3, escolha a conta associada ao bucket do S3.
9. Em Nome do bucket do S3, escolha um bucket do S3.
10. Em Prefixo do bucket do S3, insira a string gerada aleatoriamente que você especificou na política do bucket.
11. Escolha Export (Exportar) para exportar seus dados de log para o Amazon S3.
12. Para visualizar o status dos dados de log exportados para o Amazon S3, escolha Actions (Ações), View all exports to Amazon S3 (Visualizar todas as exportações para o Amazon S3).

Exporte dados de log para o Amazon S3 usando o AWS CLI

No exemplo a seguir, você usa uma tarefa de exportação para exportar todos os dados de um grupo de CloudWatch logs de registros chamado `my-log-group` para um bucket do Amazon S3 chamado. `amzn-s3-demo-bucket` Este exemplo pressupõe que você já tenha criado um grupo de logs denominado `my-log-group`.

Há suporte para a exportação de dados de log para buckets do S3 que são criptografados pelo AWS KMS . A exportação para buckets criptografados com não DSSE-KMS é suportada.

Os detalhes de como configurar a exportação dependem se o bucket do Amazon S3 para o qual você deseja exportar está na mesma conta que os logs que estão sendo exportados ou em uma conta diferente.

Tópicos

- Same-account exportação (CLI)
- Cross-account exportação (CLI)

Same-account exportação (CLI)

Se o bucket do Amazon S3 estiver na mesma conta dos logs que estão sendo exportados, use as instruções nesta seção.

Tópicos

- Crie um bucket S3 (CLI)

- Configurar permissões de acesso (CLI)
- Definir permissões em um bucket do S3 (CLI)
- (Opcional) Exportação para um bucket Amazon S3 de destino criptografado com SSE-KMS (CLI)
- Criar uma tarefa de exportação (CLI)

Crie um bucket S3 (CLI)

Recomendamos que você use um bucket criado especificamente para o CloudWatch Logs. No entanto, se quiser usar um bucket existente, você pode pular esse procedimento.

> **Note**
>
> O bucket do S3 deve residir na mesma Região que os dados de log a serem exportados. CloudWatch O Logs não oferece suporte à exportação de dados para buckets do S3 em uma região diferente.

Para criar um bucket S3 usando o AWS CLI

Em um prompt de comando, execute o seguinte comando create-bucket, em que `LocationConstraint` é a região onde você está exportando dados de log.

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-2
```

A seguir está um exemplo de saída.

```
{
    "Location": "/amzn-s3-demo-bucket"
}
```

Configurar permissões de acesso (CLI)

Para criar a tarefa de exportação posteriormente, você precisará estar conectado com a função do `AmazonS3ReadOnlyAccess` IAM e com as seguintes permissões:

- `logs:CreateExportTask`

- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Definir permissões em um bucket do S3 (CLI)

Por padrão, todos os buckets e objetos do S3 são privados. Somente o proprietário do recurso e a conta que criou o bucket podem acessar o bucket e todos os objetos que ele contém. No entanto, o proprietário do recurso pode optar por conceder permissões de acesso a outros recursos e usuários ao criar uma política de acesso padrão.

> **Important**
>
> Para tornar as exportações para buckets S3 mais seguras, agora exigimos que você especifique a lista de contas de origem que têm permissão para exportar dados de log para seu bucket S3.
> No exemplo a seguir, a lista de IDs de conta na chave da `aws:SourceAccount` seriam as contas das quais um usuário pode exportar dados de log para o bucket do S3. A chave

`aws:SourceArn` seria o recurso para o qual a ação está sendo realizada. Você pode restringir isso a um grupo de logs específico ou usar um curinga, como mostrado neste exemplo.
Recomendamos que você inclua também o ID da conta na qual o bucket do S3 foi criado para permitir a exportação dentro da mesma conta.

Para definir permissões em um bucket do S3.

1. Crie um arquivo denominado `policy.json` e adicione a seguinte política de acesso, alterando `amzn-s3-demo-bucket` para o nome do bucket do S3 e `Principal` para o endpoint da região na qual você está exportando dados de log, como, por exemplo, `us-east-1`. Use um editor de texto para criar este arquivo de política. Não use o console do 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. Defina a política que você acabou de adicionar como política de acesso ao seu bucket usando o comando put-bucket-policy. Essa política permite que o CloudWatch Logs exporte dados de log para seu bucket do S3. O proprietário do bucket terá permissões completas sobre todos os objetos exportados.

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://
policy.json
```

> **Warning**
>
> Se o bucket existente já tiver uma ou mais políticas anexadas, adicione as instruções de acesso de CloudWatch registros a essa política ou políticas. Recomendamos avaliar o conjunto resultante de permissões para ter certeza de que elas são apropriadas para os usuários que acessarão o bucket.

(Opcional) Exportação para um bucket Amazon S3 de destino criptografado com SSE-KMS (CLI)

Esse procedimento é necessário somente se você estiver exportando para um bucket do S3 que usa criptografia do lado do servidor com. AWS KMS keys Essa criptografia é conhecida como SSE-KMS.

Para exportar para um bucket criptografado com SSE-KMS

1. Use um editor de texto para criar um arquivo chamado `key_policy.json` e adicione a seguinte política de acesso. Ao adicionar a política, faça as seguintes alterações:

   - *`Region`*Substitua pela região dos seus registros.
   - *`account-ARN`*Substitua pelo ARN da conta que possui a chave 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. Digite o comando:

```
aws kms create-key --policy file://key_policy.json
```

A seguir está um exemplo de saída deste comando:

```
{
    "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. Use um editor de texto para criar um arquivo chamado `bucketencryption.json` com o seguinte conteúdo:

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
```

```
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

4. Insira o comando a seguir, *amzn-s3-demo-bucket* substituindo-o pelo nome do bucket para o qual você está exportando registros.

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-
encryption-configuration file://bucketencryption.json
```

Se o comando não retornar um erro, o processo foi bem-sucedido.

Criar uma tarefa de exportação (CLI)

Use o comando a seguir para criar a tarefa de exportação. Depois de criada, a tarefa de exportação pode levar de alguns segundos a algumas horas, dependendo do tamanho dos dados a serem exportados.

Para exportar dados para o Amazon S3 usando o AWS CLI

1. Faça login com permissões suficientes, conforme documentado em Configurar permissões de acesso (CLI).
2. Em um prompt de comando, use o seguinte comando create-export-task para criar a tarefa de exportação.

```
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"
```

A seguir está um exemplo de saída.

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

Cross-account exportação (CLI)

Se o bucket do Amazon S3 estiver em uma conta diferente da conta dos logs que estão sendo exportados, use as instruções nesta seção.

Tópicos

- Crie um bucket S3 para exportação entre contas (CLI)
- Configurar permissões de acesso para exportação entre contas (CLI)
- Defina permissões em um bucket do S3 para exportação entre contas (CLI)
- (Opcional) Exportação para um bucket Amazon S3 de destino criptografado para exportação entre SSE-KMS contas (CLI)
- Crie uma tarefa de exportação para exportação entre contas (CLI)

Crie um bucket S3 para exportação entre contas (CLI)

Recomendamos que você use um bucket criado especificamente para o CloudWatch Logs. No entanto, se você desejar usar um bucket existente, vá para a etapa 2.

> **Note**
>
> O bucket do S3 deve residir na mesma Região que os dados de log a serem exportados. CloudWatch O Logs não oferece suporte à exportação de dados para buckets do S3 em uma região diferente.

Para criar um bucket S3 usando o AWS CLI

Em um prompt de comando, execute o seguinte comando create-bucket, em que `LocationConstraint` é a região onde você está exportando dados de log.

```
aws s3api create-bucket --bucket amzn-s3-demo-bucket --create-bucket-configuration
 LocationConstraint=us-east-2
```

A seguir está um exemplo de saída.

```
{
    "Location": "/amzn-s3-demo-bucket"
}
```

Configurar permissões de acesso para exportação entre contas (CLI)

Primeiro, você deve criar uma nova política do IAM para permitir que CloudWatch os Logs tenham a `s3:PutObject` ação para o bucket Amazon S3 de destino na conta de destino.

Além da `s3:PutObject` ação, as ações adicionais incluídas na política dependem se o bucket de destino usa AWS KMS criptografia ou se as ACLs estão habilitadas usando a configuração de propriedade de objetos do S3.

- Se estiver usando a criptografia KMS, adicione as ações `kms:GenerateDataKey` e `kms:Decrypt` ao recurso chave
- Se as ACLs estiverem habilitadas no bucket, adicione a ação `s3:PutObjectAcl` ao recurso do bucket.

Altere `amzn-s3-demo-bucket` para o nome do seu bucket S3 de destino nas políticas a seguir.

A política que deve ser criada depende de o bucket de destino usar criptografia do AWS KMS . Se não usar AWS KMS criptografia, crie uma política com o conteúdo a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

Se o bucket de destino usar AWS KMS criptografia, crie uma política com o conteúdo a seguir.

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"
        }
    ]
}
```

Se as ACLs estiverem habilitadas no bucket de destino, adicione s3: PutObjectAcl a s3: PutObject Action block nas políticas acima.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:PutObject",
               "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

Para criar uma tarefa de exportação, você deve estar conectado com uma função do IAM que tenha a política `AmazonS3ReadOnlyAccess` gerenciada anexada, a política do IAM criada acima e também com as seguintes permissões:

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Defina permissões em um bucket do S3 para exportação entre contas (CLI)

Por padrão, todos os buckets e objetos do S3 são privados. Somente o proprietário do recurso e a conta que criou o bucket podem acessar o bucket e todos os objetos que ele contém. No entanto, o proprietário do recurso pode optar por conceder permissões de acesso a outros recursos e usuários ao criar uma política de acesso padrão.

> **Important**
>
> Para tornar as exportações para buckets S3 mais seguras, agora exigimos que você especifique a lista de contas de origem que têm permissão para exportar dados de log para seu bucket S3.

No exemplo a seguir, a lista de IDs de conta na chave da `aws:SourceAccount` seriam as contas das quais um usuário pode exportar dados de log para o bucket do S3. A chave `aws:SourceArn` seria o recurso para o qual a ação está sendo realizada. Você pode restringir isso a um grupo de logs específico ou usar um curinga, como mostrado neste exemplo.
Recomendamos que você inclua também o ID da conta na qual o bucket do S3 foi criado para permitir a exportação dentro da mesma conta.

Para definir permissões em um bucket do S3.

1. Crie um arquivo chamado `policy.json` e adicione a seguinte política de acesso, alterando `amzn-s3-demo-bucket` para o nome do seu bucket S3 de destino, `Principal` para o endpoint da região para a qual você está exportando dados de log, como. `us-west-1` Use um editor de texto para criar este arquivo de política. Não use o console do 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. Defina a política que você acabou de adicionar como política de acesso ao seu bucket usando o comando put-bucket-policy. Essa política permite que o CloudWatch Logs exporte dados de log para seu bucket do S3. O proprietário do bucket terá permissões completas sobre todos os objetos exportados.

```
aws s3api put-bucket-policy --bucket amzn-s3-demo-bucket --policy file://
policy.json
```

> **Warning**
>
> Se o bucket existente já tiver uma ou mais políticas anexadas, adicione as instruções de acesso de CloudWatch registros a essa política ou políticas. Recomendamos avaliar o conjunto resultante de permissões para ter certeza de que elas são apropriadas para os usuários que acessarão o bucket.

(Opcional) Exportação para um bucket Amazon S3 de destino criptografado para exportação entre SSE-KMS contas (CLI)

Esse procedimento é necessário somente se você estiver exportando para um bucket do S3 que usa criptografia do lado do servidor com. AWS KMS keys Essa criptografia é conhecida como SSE-KMS.

Para exportar para um bucket criptografado com SSE-KMS

1. Use um editor de texto para criar um arquivo chamado `key_policy.json` e adicione a seguinte política de acesso. Ao adicionar a política, faça as seguintes alterações:

   - *us-east-1*Substitua pela região dos seus registros.
   - *account-ARN*Substitua pelo ARN da conta que possui a chave KMS.
   - *123456789012*Substitua pelo número da conta que possui a chave KMS.
   - *key_id*com o ID da chave kms.
   - *role_name*com a função usada para criar a tarefa de exportação.

   JSON

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

```
            "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. Digite o comando:

```
aws kms create-key --policy file://key_policy.json
```

A seguir está um exemplo de saída deste comando:

```
{
    "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
    }
```

3. Use um editor de texto para criar um arquivo chamado `bucketencryption.json` com o seguinte conteúdo:

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

4. Insira o comando a seguir, *amzn-s3-demo-bucket* substituindo-o pelo nome do bucket para o qual você está exportando registros.

```
aws s3api put-bucket-encryption --bucket amzn-s3-demo-bucket --server-side-
encryption-configuration file://bucketencryption.json
```

Se o comando não retornar um erro, o processo foi bem-sucedido.

Crie uma tarefa de exportação para exportação entre contas (CLI)

Use o comando a seguir para criar a tarefa de exportação. Depois de criada, a tarefa de exportação pode levar de alguns segundos a algumas horas, dependendo do tamanho dos dados a serem exportados.

Para exportar dados para o Amazon S3 usando o AWS CLI

1. Faça login com permissões suficientes, conforme documentado em Configurar permissões de acesso (CLI).
2. Em um prompt de comando, use o seguinte comando create-export-task para criar a tarefa de exportação.

```
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"
```

A seguir está um exemplo de saída.

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

Descrever as tarefas de exportação (CLI)

Depois de criar uma tarefa de exportação, você pode obter o status atual da tarefa.

Para descrever as tarefas de exportação usando o AWS CLI

No prompt de comando, use o seguinte comando describe-export-tasks.

```
aws logs --profile CWLExportUser describe-export-tasks --task-id
 "cda45419-90ea-4db5-9833-aade86253e66"
```

A seguir está um exemplo de saída.

```
{
   "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
   }]
}
```

Você pode usar o comando `describe-export-tasks` de três maneiras diferentes:

- Sem filtros - lista todas as suas tarefas de exportação, na ordem inversa de criação.
- Filtrar com base no ID de tarefa - lista a tarefa de exportação, se houver, com o ID especificado.
- Filtrar com base no status da tarefa - lista as tarefas de exportação com o status especificado.

Por exemplo, use o comando a seguir para filtrar com base no status `FAILED`.

```
aws logs --profile CWLExportUser describe-export-tasks --status-code "FAILED"
```

A seguir está um exemplo de saída.

```
{
   "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
   }]
}
```

Cancelar uma tarefa de exportação (CLI)

Você pode cancelar uma tarefa de exportação se estiver no estado `PENDING` ou `RUNNING`.

Para cancelar uma tarefa de exportação usando o AWS CLI

No prompt de comando, use o seguinte comando cancel-export-task:

```
aws logs --profile CWLExportUser cancel-export-task --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

Você pode usar o comando describe-export-tasks para verificar se a tarefa foi cancelada com êxito.

Streaming de dados de CloudWatch registros para o Amazon OpenSearch Service

Você pode configurar um grupo de CloudWatch logs no Amazon Logs, para que você possa transmitir dados para o seu cluster do Amazon OpenSearch Service quase em tempo real. Para obter mais informações, consulte Real-time processamento de dados de registro com assinaturas.

> **Note**
>
> O streaming para o OpenSearch serviço é suportado somente para grupos de registros na classe de log Standard. Para obter mais informações sobre classes de logs, consulte Classes de logs.

Dependendo da quantidade de dados de log transmitidos no fluxo, considere definir um limite de simultaneidade ao nível da função. Para obter mais informações, consulte Escalabilidade de funções do Lambda.

> **Note**
>
> Como o streaming de grandes quantidades de dados do CloudWatch Logs para o OpenSearch Serviço pode resultar em altas cobranças de uso, recomendamos que você crie um orçamento no Gerenciamento de Faturamento e Custos da AWS console. Para obter mais informações, consulte Gerenciando seus custos com AWS orçamentos.

Esta seção descreve os pré-requisitos que você deve preencher antes de inscrever um grupo de registros no Serviço. OpenSearch Também descreve como inscrever um grupo de registros no OpenSearch Serviço.

Pré-requisitos

Antes de começar, crie um domínio OpenSearch de serviço. O domínio pode ter acesso público ou acesso à VPC, mas você não poderá modificar o tipo de acesso depois que o domínio for criado. Talvez você queira revisar as configurações do domínio de OpenSearch serviço posteriormente e modificar a configuração do cluster com base na quantidade de dados que o cluster processará.

Para obter instruções sobre como criar um domínio, consulte Criação OpenSearch de domínios de serviço.

Para obter mais informações sobre o OpenSearch serviço, consulte o Amazon OpenSearch Service Developer Guide.

Inscrever um grupo de registros no OpenSearch Serviço

Você pode usar o CloudWatch console para inscrever um grupo de registros no OpenSearch Serviço.

Para inscrever um grupo de registros no OpenSearch Serviço

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Grupos de logs.
3. Selecione o nome do grupo de logs.
4. Escolha Ações, Filtros de assinatura, Criar filtro OpenSearch de assinatura do Amazon Service.
5. Escolha se deseja fazer streaming para um cluster nessa conta ou em outra conta.
   - Se você escolheu essa conta, selecione o domínio que criou na etapa anterior.
   - Caso tenha escolhido outra conta, forneça o ARN do domínio e o endpoint.
6. Para a função de execução do Lambda IAM, escolha a função do IAM que o Lambda deve usar ao executar chamadas. OpenSearch

   A função do IAM escolhida deve atender a estes requisitos:

   - Ela deve possuir `lambda.amazonaws.com` na relação de confiança.
   - Ela deve incluir a seguinte política:

     JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOpenSearchStreamingAccess",
            "Action": [
                "es:*"
            ],
```

```
            "Effect": "Allow",
            "Resource": "arn:aws:es:us-
east-1:123456789012:domain/cloudwatch-logs/*"
        }
    ]
}
```

- Se o domínio OpenSearch de serviço de destino usar acesso à VPC, a função deverá ter a AWSLambdaVPCAccessExecutionRolepolítica anexada. Essa política gerenciada pela Amazon concede ao Lambda acesso à VPC do cliente, permitindo que a Lambda grave no endpoint na VPC. OpenSearch

7. Em Log format, escolha um formato de log.
8. Em Subscription filter pattern (Padrão de filtro de assinatura), digite os termos ou o padrão a ser localizado nos eventos de log. Isso garante que você envie somente os dados de seu interesse para o seu OpenSearch cluster. Para obter mais informações, consulte Criar métricas de eventos de log usando filtros.
9. (Opcional) Em Select log data to test (Selecionar dados de log para testar), selecione um fluxo de logs e escolha Test pattern (Testar padrão) para verificar se o filtro de pesquisa está retornando os resultados esperados.
10. Selecione Start streaming (Iniciar transmissão).

Exemplos de código para CloudWatch registros usando AWS SDKs

Os exemplos de código a seguir mostram como usar o CloudWatch Logs com um kit AWS de desenvolvimento de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Exemplos básicos de CloudWatch registros usando AWS SDKs
  - Olá, CloudWatch Logs
  - Ações para CloudWatch registros usando AWS SDKs
    - Use AssociateKmsKeycom um AWS SDK
    - Use CancelExportTaskcom um AWS SDK
    - Use CreateExportTaskcom um AWS SDK
    - Use CreateLogGroupcom um AWS SDK ou CLI
    - Use CreateLogStreamcom um AWS SDK ou CLI
    - Use DeleteLogGroupcom um AWS SDK ou CLI
    - Use DeleteSubscriptionFiltercom um AWS SDK
    - Use DeleteSyslogConfigurationcom um AWS SDK
    - Use DescribeExportTaskscom um AWS SDK
    - Use DescribeLogGroupscom um AWS SDK ou CLI
    - Use DescribeLogStreamscom um AWS SDK ou CLI
    - Use DescribeSubscriptionFilterscom um AWS SDK
    - Use GetLogEventscom um AWS SDK ou CLI

    - Use GetQueryResultscom um AWS SDK
    - Use ListSyslogConfigurationscom um AWS SDK
    - Use PutResourcePolicycom um AWS SDK
    - Use PutSubscriptionFiltercom um AWS SDK
    - Use PutSyslogConfigurationcom um AWS SDK
    - Use StartLiveTailcom um AWS SDK ou CLI
    - Use StartQuerycom um AWS SDK
- Cenários para CloudWatch registros usando AWS SDKs
    - Configurar a conectividade do serviço de contêiner
    - Como criar sua primeira função com tecnologia sem servidor
    - Aprenda noções básicas sobre a ingestão de syslog do CloudWatch Logs usando um AWS SDK
    - Use CloudWatch Logs para executar uma consulta grande
    - Usar eventos programados para invocar uma função do Lambda

Exemplos básicos de CloudWatch registros usando AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon CloudWatch Logs com AWS SDKs.

Exemplos

- Olá, CloudWatch Logs
- Ações para CloudWatch registros usando AWS SDKs
    - Use AssociateKmsKeycom um AWS SDK
    - Use CancelExportTaskcom um AWS SDK
    - Use CreateExportTaskcom um AWS SDK
    - Use CreateLogGroupcom um AWS SDK ou CLI
    - Use CreateLogStreamcom um AWS SDK ou CLI
    - Use DeleteLogGroupcom um AWS SDK ou CLI
    - Use DeleteSubscriptionFiltercom um AWS SDK
    - Use DeleteSyslogConfigurationcom um AWS SDK
    - Use DescribeExportTaskscom um AWS SDK

- Use DescribeLogGroupscom um AWS SDK ou CLI
- Use DescribeLogStreamscom um AWS SDK ou CLI
- Use DescribeSubscriptionFilterscom um AWS SDK
- Use GetLogEventscom um AWS SDK ou CLI
- Use GetQueryResultscom um AWS SDK
- Use ListSyslogConfigurationscom um AWS SDK
- Use PutResourcePolicycom um AWS SDK
- Use PutSubscriptionFiltercom um AWS SDK
- Use PutSyslogConfigurationcom um AWS SDK
- Use StartLiveTailcom um AWS SDK ou CLI
- Use StartQuerycom um AWS SDK

Olá, CloudWatch Logs

O exemplo de código a seguir mostra como começar a usar o CloudWatch Logs.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
def hello_cloudwatch_logs() -> None:
    """
    Verifies connectivity to CloudWatch Logs by calling DescribeLogGroups
    and displaying the first page of log groups with name, ARN, and
    creation time.
    """
    logs_client = boto3.client("logs")

    try:
        response = logs_client.describe_log_groups()
        log_groups = response.get("logGroups", list())
```

```
        if not log_groups:
            print("No log groups found in this Region for your account.")
        else:
            print(f"Found {len(log_groups)} log group(s):\n")
            for lg in log_groups:
                name = lg.get("logGroupName", "N/A")
                arn = lg.get("arn", "N/A")
                creation_ms = lg.get("creationTime", 0)
                creation_dt = datetime.fromtimestamp(
                    creation_ms / 1000, tz=timezone.utc
                )
                print(f"  Name: {name}")
                print(f"  ARN:  {arn}")
                print(f"  Created: {creation_dt.strftime('%Y-%m-%d %H:%M:%S
UTC')}")
                print()

    except ClientError as error:
        logger.error(
            "Failed to describe log groups: %s",
            error.response["Error"]["Message"],
        )
        raise
```

- Para obter detalhes da API, consulte a DescribeLogGroupsReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Ações para CloudWatch registros usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do CloudWatch Logs com AWS SDKs. Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Esses trechos chamam a API CloudWatch Logs e são trechos de código de programas maiores que devem ser executados em contexto. É possível ver as ações em contexto em Cenários para CloudWatch registros usando AWS SDKs.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência da API Amazon CloudWatch Logs.

Exemplos

- Use AssociateKmsKeycom um AWS SDK
- Use CancelExportTaskcom um AWS SDK
- Use CreateExportTaskcom um AWS SDK
- Use CreateLogGroupcom um AWS SDK ou CLI
- Use CreateLogStreamcom um AWS SDK ou CLI
- Use DeleteLogGroupcom um AWS SDK ou CLI
- Use DeleteSubscriptionFiltercom um AWS SDK
- Use DeleteSyslogConfigurationcom um AWS SDK
- Use DescribeExportTaskscom um AWS SDK
- Use DescribeLogGroupscom um AWS SDK ou CLI
- Use DescribeLogStreamscom um AWS SDK ou CLI
- Use DescribeSubscriptionFilterscom um AWS SDK
- Use GetLogEventscom um AWS SDK ou CLI
- Use GetQueryResultscom um AWS SDK
- Use ListSyslogConfigurationscom um AWS SDK
- Use PutResourcePolicycom um AWS SDK
- Use PutSubscriptionFiltercom um AWS SDK
- Use PutSyslogConfigurationcom um AWS SDK
- Use StartLiveTailcom um AWS SDK ou CLI
- Use StartQuerycom um AWS SDK

Use **`AssociateKmsKey`**com um AWS SDK

O código de exemplo a seguir mostra como usar `AssociateKmsKey`.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    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}.");
            }
        }
    }
```

- Para obter detalhes da API, consulte AssociateKmsKeya Referência AWS SDK para .NET da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CancelExportTask`**com um AWS SDK

O código de exemplo a seguir mostra como usar `CancelExportTask`.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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.");
        }
    }
}
```

- Para obter detalhes da API, consulte CancelExportTaska Referência AWS SDK para .NET da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CreateExportTask`**com um AWS SDK

O código de exemplo a seguir mostra como usar `CreateExportTask`.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    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.");
            }
        }
    }
```

- Para obter detalhes da API, consulte CreateExportTaska Referência AWS SDK para .NET da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CreateLogGroup`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateLogGroup`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Configurar a conectividade do serviço de contêiner
- Aprenda as noções básicas sobre a ingestão de syslog

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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.");
            }
        }
    }
```

- Para obter detalhes da API, consulte CreateLogGroupa Referência AWS SDK para .NET da API.

CLI

AWS CLI

O seguinte comando cria um grupo de logs chamado `my-logs`:

```
aws logs create-log-group --log-group-name my-logs
```

- Para obter detalhes da API, consulte CreateLogGroupna Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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();
```

- Para obter detalhes da API, consulte CreateLogGroupa Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CreateLogStream`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateLogStream`.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    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.");
            }
        }
    }
```

- Para obter detalhes da API, consulte CreateLogStreama Referência AWS SDK para .NET da API.

CLI

AWS CLI

O seguinte comando cria um fluxo de logs `20150601` no grupo de logs `my-logs`:

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- Para obter detalhes da API, consulte CreateLogStreamna Referência de AWS CLI Comandos.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DeleteLogGroup`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DeleteLogGroup`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Configurar a conectividade do serviço de contêiner
- Como criar sua primeira função com tecnologia sem servidor
- Aprenda as noções básicas sobre a ingestão de syslog

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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}.");
            }
        }
    }
```

- Para obter detalhes da API, consulte DeleteLogGroupa Referência AWS SDK para .NET da API.

CLI

AWS CLI

O seguinte comando exclui um grupo de logs chamado `my-logs`:

```
aws logs delete-log-group --log-group-name my-logs
```

- Para obter detalhes da API, consulte DeleteLogGroupna Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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();
```

- Para obter detalhes da API, consulte DeleteLogGroupa Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DeleteSubscriptionFilter`**com um AWS SDK

Os exemplos de código a seguir mostram como usar o `DeleteSubscriptionFilter`.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Inclua os arquivos necessários.

```
#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/DeleteSubscriptionFilterRequest.h>
#include <iostream>
```

Excluir o filtro de assinatura.

```
        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;
        }
```

- Para obter detalhes da API, consulte DeleteSubscriptionFiltera Referência AWS SDK para C ++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte DeleteSubscriptionFiltera Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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();
```

- Para obter detalhes da API, consulte DeleteSubscriptionFiltera Referência AWS SDK para JavaScript da API.

SDK para JavaScript (v2)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da 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);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes da API, consulte DeleteSubscriptionFiltera Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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")
    }
}
```

- Para obter detalhes da API, consulte a DeleteSubscriptionFilterreferência da API AWS SDK for Kotlin.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DeleteSyslogConfiguration`**com um AWS SDK

O código de exemplo a seguir mostra como usar `DeleteSyslogConfiguration`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Aprenda as noções básicas sobre a ingestão de syslog

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    def delete_syslog_configuration(
        self, log_group_identifier: str, vpc_endpoint_id: str
    ) -> None:
        """
        Deletes a syslog configuration for a log group. After deletion, syslog
        data is no longer ingested through the specified VPC endpoint.

        :param log_group_identifier: The name or ARN of the log group.
```

```
        :param vpc_endpoint_id: The ID of the VPC endpoint associated with the
            syslog configuration.
        :raises ClientError: If the syslog configuration does not exist (in which
            case the error is logged but not re-raised during cleanup).
        """
        try:
            self.logs_client.delete_syslog_configuration(
                logGroupIdentifier=log_group_identifier,
                vpcEndpointId=vpc_endpoint_id,
            )
            logger.info(
                "Deleted syslog configuration for log group '%s' and VPC endpoint
'%s'.",
                log_group_identifier,
                vpc_endpoint_id,
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.info(
                    "Syslog configuration does not exist or was already deleted:
%s",
                    error.response["Error"]["Message"],
                )
            else:
                logger.error(
                    "Failed to delete syslog configuration: %s",
                    error.response["Error"]["Message"],
                )
                raise
```

- Para obter detalhes da API, consulte a DeleteSyslogConfigurationReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DescribeExportTasks`**com um AWS SDK

O código de exemplo a seguir mostra como usar `DescribeExportTasks`.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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);
        }
    }
```

- Para obter detalhes da API, consulte DescribeExportTasksa Referência AWS SDK para .NET da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DescribeLogGroups`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeLogGroups`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Aprenda as noções básicas sobre a ingestão de syslog

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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);
        }
    }
```

- Para obter detalhes da API, consulte DescribeLogGroupsa Referência AWS SDK para .NET da API.

CLI

AWS CLI

O seguinte comando descreve um grupo de logs chamado `my-logs`:

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

Saída:

```
{
    "logGroups": [
        {
            "storedBytes": 0,
            "metricFilterCount": 0,
            "creationTime": 1433189500783,
            "logGroupName": "my-logs",
            "retentionInDays": 5,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"
        }
    ]
}
```

- Para obter detalhes da API, consulte DescribeLogGroupsna Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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;
};
```

- Para obter detalhes da API, consulte DescribeLogGroupsa Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DescribeLogStreams`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DescribeLogStreams`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Como criar sua primeira função com tecnologia sem servidor

CLI

AWS CLI

O comando a seguir mostra todos os fluxos de log começando com o prefixo `2015` no grupo de logs `my-logs`:

```
aws logs describe-log-streams --log-group-name my-logs --log-stream-name-
prefix 2015
```

Saída:

```
{
    "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
        }
    ]
```

```
}
```

- Para obter detalhes da API, consulte DescribeLogStreamsna Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Pesquisa fluxos de log em um grupo de logs especificado que correspondam a um determinado prefixo.

```
/**
 * 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
        }
    }
}
```

Imprime metadados sobre o fluxo de logs mais recente em um grupo de logs especificado.

```
/**
 * 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());
        }
    }
}
```

- Para obter detalhes da API, consulte DescribeLogStreamsa Referência AWS SDK for Java 2.x da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DescribeSubscriptionFilters`**com um AWS SDK

Os exemplos de código a seguir mostram como usar o `DescribeSubscriptionFilters`.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Inclua os arquivos necessários.

```
#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>
```

Liste os filtros de assinatura.

```
        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();
        }
```

- Para obter detalhes da API, consulte DescribeSubscriptionFilters a Referência AWS SDK para C++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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");
    }
}
```

- Para obter detalhes da API, consulte DescribeSubscriptionFilters a Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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();
```

- Para obter detalhes da API, consulte DescribeSubscriptionFilters a Referência AWS SDK para JavaScript da API.

SDK para JavaScript (v2)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da 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);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes da API, consulte DescribeSubscriptionFilterssa Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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}")
        }
    }
}
```

- Para obter detalhes da API, consulte a DescribeSubscriptionFiltersreferência da API AWS SDK for Kotlin.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`GetLogEvents`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `GetLogEvents`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Como criar sua primeira função com tecnologia sem servidor

CLI

AWS CLI

O seguinte comando recupera eventos de logs de um fluxo de logs chamado `20150601` no grupo de logs `my-logs`:

```
aws logs get-log-events --log-group-name my-logs --log-stream-name 20150601
```

Saída:

```
{
    "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"
}
```

- Para obter detalhes da API, consulte GetLogEventsna Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte GetLogEventsa Referência AWS SDK for Java 2.x da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`GetQueryResults`**com um AWS SDK

Os exemplos de código a seguir mostram como usar o `GetQueryResults`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Executar uma consulta grande

.NET

SDK para .NET (v4)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   /// <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;
       }
   }
```

- Para obter detalhes da API, consulte GetQueryResultsa Referência AWS SDK para .NET da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
  return this.client.send(new GetQueryResultsCommand({ queryId }));
}
```

- Para obter detalhes da API, consulte GetQueryResultsa Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    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", [])
```

- Para obter detalhes da API, consulte a GetQueryResultsReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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.
```

- Para obter detalhes da API, consulte a GetQueryResultsreferência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ListSyslogConfigurations`**com um AWS SDK

O código de exemplo a seguir mostra como usar `ListSyslogConfigurations`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Aprenda as noções básicas sobre a ingestão de syslog

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    def list_syslog_configurations(
        self,
        log_group_identifier: Optional[str] = None,
```

```
        vpc_endpoint_id: Optional[str] = None,
    ) -> list[dict[str, Any]]:
        """
        Lists syslog configurations, optionally filtered by log group or VPC
endpoint.

        Handles pagination by following nextToken until all results are returned.

        :param log_group_identifier: Optional log group name or ARN to filter by.
        :param vpc_endpoint_id: Optional VPC endpoint ID to filter by.
        :return: A list of syslog configuration dictionaries.
        :raises ClientError: If a filter parameter is invalid.
        """
        try:
            configurations = list()
            params = dict()
            if log_group_identifier is not None:
                params["logGroupIdentifier"] = log_group_identifier
            if vpc_endpoint_id is not None:
                params["vpcEndpointId"] = vpc_endpoint_id

            while True:
                response = self.logs_client.list_syslog_configurations(**params)
                configurations.extend(response.get("syslogConfigurations",
list()))
                next_token = response.get("nextToken", None)
                if next_token is None:
                    break
                params["nextToken"] = next_token

            logger.info("Listed %d syslog configuration(s).",
len(configurations))
            return configurations
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidParameterException":
                logger.error(
                    "Invalid filter parameter: %s. Ensure the log group
identifier "
                    "or VPC endpoint ID is correctly formatted.",
                    error.response["Error"]["Message"],
                )
            else:
                logger.error(
                    "Failed to list syslog configurations: %s",
```

```
                    error.response["Error"]["Message"],
                )
            raise
```

- Para obter detalhes da API, consulte a ListSyslogConfigurationsReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`PutResourcePolicy`**com um AWS SDK

O código de exemplo a seguir mostra como usar `PutResourcePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Aprenda as noções básicas sobre a ingestão de syslog

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    def put_resource_policy(
        self,
        log_group_arn: str,
        vpc_endpoint_id: str,
    ) -> dict[str, Any]:
        """
        Creates or updates a resource-scoped policy on a log group that grants
        the syslog.logs.amazonaws.com service principal permission to call
```

```
        logs:PutLogEvents and logs:CreateLogStream.

        A resource-scoped policy (one attached to a specific log group) is
        specified with resourceArn and policyDocument. The PutResourcePolicy
        API does not allow policyName to be combined with resourceArn.

        :param log_group_arn: The ARN of the log group (with trailing :*).
        :param vpc_endpoint_id: The VPC endpoint ID to scope the condition.
        :return: The resource policy response.
        """
        policy_document = json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "SyslogIngestPermissions",
                        "Effect": "Allow",
                        "Principal": {"Service": "syslog.logs.amazonaws.com"},
                        "Action": [
                            "logs:PutLogEvents",
                            "logs:CreateLogStream",
                        ],
                        "Resource": log_group_arn,
                        "Condition": {
                            "StringEquals": {"aws:sourceVpce": vpc_endpoint_id}
                        },
                    }
                ],
            }
        )
        try:
            response = self.logs_client.put_resource_policy(
                policyDocument=policy_document,
                resourceArn=log_group_arn,
            )
            logger.info("Put resource policy on log group '%s'.", log_group_arn)
            return response
        except ClientError as error:
            logger.error(
                "Failed to put resource policy: %s",
                error.response["Error"]["Message"],
            )
            raise
```

- Para obter detalhes da API, consulte a PutResourcePolicyReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`PutSubscriptionFilter`**com um AWS SDK

Os exemplos de código a seguir mostram como usar o `PutSubscriptionFilter`.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Inclua os arquivos necessários.

```
#include <aws/core/Aws.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/PutSubscriptionFilterRequest.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

Crie o filtro de assinatura.

```
        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;
        }
```

- Para obter detalhes da API, consulte PutSubscriptionFiltera Referência AWS SDK para C++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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(
                    "Successfully created CloudWatch logs subscription filter
 %s",
                    filter);

        } catch (CloudWatchLogsException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte PutSubscriptionFiltera Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
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();
```

- Para obter detalhes da API, consulte PutSubscriptionFiltera Referência AWS SDK para JavaScript da API.

SDK para JavaScript (v2)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da 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);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes da API, consulte PutSubscriptionFiltera Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **PutSyslogConfiguration**com um AWS SDK

O código de exemplo a seguir mostra como usar `PutSyslogConfiguration`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Aprenda as noções básicas sobre a ingestão de syslog

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    def put_syslog_configuration(
        self, log_group_identifier: str, vpc_endpoint_id: str
    ) -> None:
        """
        Creates or updates a syslog configuration for a log group. This enables
        ingestion of syslog data through the specified VPC endpoint.

        :param log_group_identifier: The name or ARN of the log group to
associate
            with the syslog configuration.
        :param vpc_endpoint_id: The ID of the VPC endpoint to use for syslog
            ingestion.
        :raises ClientError: If the log group or VPC endpoint does not exist.
        """
        try:
            self.logs_client.put_syslog_configuration(
                logGroupIdentifier=log_group_identifier,
                vpcEndpointId=vpc_endpoint_id,
```

```
            )
            logger.info(
                "Created syslog configuration for log group '%s' with VPC
 endpoint '%s'.",
                log_group_identifier,
                vpc_endpoint_id,
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "ResourceNotFoundException":
                logger.error(
                    "The specified log group or VPC endpoint does not exist: %s",
                    error.response["Error"]["Message"],
                )
            else:
                logger.error(
                    "Failed to put syslog configuration: %s",
                    error.response["Error"]["Message"],
                )
            raise
```

- Para obter detalhes da API, consulte a PutSyslogConfigurationReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`StartLiveTail`**com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `StartLiveTail`.

.NET

SDK para .NET

Inclua os arquivos necessários.

```
using Amazon;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
```

Inicie a sessão do 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;
            }
```

Você pode lidar com os eventos da sessão do Live Tail de duas maneiras:

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

- Para obter detalhes da API, consulte StartLiveTaila Referência AWS SDK para .NET da API.

CLI

AWS CLI

O comando a seguir inicia uma sessão do Live Tail em um grupo de registros chamado`my-logs`:

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs
```

O comando a seguir inicia uma sessão do Live Tail em um grupo de registros chamado `my-logs` no modo interativo:

```
aws logs start-live-tail --log-group-identifiers arn:aws:logs:us-
east-1:111111222222:log-group:my-logs --mode interactive
```

No modo interativo, você pode destacar até cinco termos nos registros com cauda. Os códigos de severidade são destacados por padrão. Pressione h para entrar no modo de destaque e, em seguida, digite os termos a serem destacados, um por vez, e pressione enter. Pressione `c` para limpar o (s) termo (s) destacado (s). Pressione `t` para alternar a formatação entre o texto. JSON/Plain Pressione `Esc` para sair da sessão do Live Tail. Pressione down as teclasup/para rolar para cima ou para baixo entre os eventos de registro, use`Ctrl + u`/`Ctrl + d`para rolar mais rápido. Pressione q para rolar até os últimos eventos do registro.

- Para obter detalhes da API, consulte StartLiveTailna Referência de AWS CLI Comandos.

Go

SDK para Go V2

Inclua os arquivos necessários.

```
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"
)
```

Gerencie os eventos da sessão do 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 occured during streaming: %v", err)
   } else if event == nil {
    log.Println("Stream is Closed")
    return
   } else {
    log.Fatalf("Unknown event type: %T", e)
   }
  }
 }
}
```

Inicie a sessão do 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)
```

Interrompa a sessão do Live Tail após um período decorrido.

```
// Close the stream (which ends the session) after a timeout
time.Sleep(10 * time.Second)
stream.Close()
log.Println("Event stream closed")
```

- Para obter detalhes da API, consulte StartLiveTaila Referência AWS SDK para Go da API.

Java

SDK para Java 2.x

Inclua os arquivos necessários.

```
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;
```

Gerencie os eventos da sessão do 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();
    }
```

Inicie a sessão do 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));
```

Interrompa a sessão do Live Tail após um período decorrido.

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

- Para obter detalhes da API, consulte StartLiveTaila Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

Inclua os arquivos necessários.

```
import { CloudWatchLogsClient, StartLiveTailCommand } from "@aws-sdk/client-
cloudwatch-logs";
```

Gerencie os eventos da sessão do 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)
    }
}
```

Inicie a sessão do 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);
    }
```

Interrompa a sessão do Live Tail após um período decorrido.

```
    /* Set a timeout to close the client. This will stop the Live Tail session.
*/
    setTimeout(function() {
        console.log("Client timeout");
        client.destroy();
      }, 10000);
```

- Para obter detalhes da API, consulte StartLiveTaila Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

Inclua os arquivos necessários.

```
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
```

Inicie a sessão do 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 unsubcribe 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)
    }
```

- Para obter detalhes da API, consulte a StartLiveTailreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

Inclua os arquivos necessários.

```
import boto3
import time
from datetime import datetime
```

Inicie a sessão do 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}]
{log}'.format(date=datetime.fromtimestamp(log_event['timestamp']/1000),log=log_event['me
            else:
                # On-stream exceptions are captured here
                raise RuntimeError(str(event))
    except Exception as e:
        print(e)
```

- Para obter detalhes da API, consulte a StartLiveTailReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`StartQuery`**com um AWS SDK

Os exemplos de código a seguir mostram como usar o `StartQuery`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Executar uma consulta grande

.NET

SDK para .NET (v4)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <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;
        }
    }
```

- Para obter detalhes da API, consulte StartQuerya Referência AWS SDK para .NET da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/**
 * 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;
  }
}
```

- Para obter detalhes da API, consulte StartQuerya Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    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}")
```

- Para obter detalhes da API, consulte a StartQueryReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   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.
```

- Para obter detalhes da API, consulte a StartQueryreferência da API AWS SDK for SAP ABAP.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para CloudWatch registros usando AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns em CloudWatch Logs com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções no CloudWatch Logs ou combinadas com outras Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, onde podem ser encontradas instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Configurar a conectividade do serviço de contêiner
- Como criar sua primeira função com tecnologia sem servidor
- Aprenda noções básicas sobre a ingestão de syslog do CloudWatch Logs usando um AWS SDK
- Use CloudWatch Logs para executar uma consulta grande
- Usar eventos programados para invocar uma função do Lambda

Configurar a conectividade do serviço de contêiner

O exemplo de código a seguir mostra como:

- Criar a infraestrutura da VPC.
- Configurar registro em log
- Criar o cluster do ECS.
- Configurar perfis do IAM.
- Criar o serviço com o Service Connect.
- Verificar a implantação
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/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 "$0"
main "$@"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI .
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition
  - DescribeClusters
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks

- DescribeVpcs
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListNamespaces
- ListServices
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Aguardar (Amazon EC2)
- Aguardar (Amazon ECS)

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar sua primeira função com tecnologia sem servidor

O exemplo de código a seguir mostra como:

- Criar um perfil do IAM para o Lambda.
- Criar código da função.
- Criar uma função do Lambda
- Testar uma função do Lambda.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/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 ${TIMEOUT}
 seconds."
            return 1
        fi
        sleep "$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."
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI .
  - AttachRolePolicy
  - CreateFunction

- CreateRole
- DeleteFunction
- DeleteLogGroup
- DeleteRole
- DescribeLogStreams
- DetachRolePolicy
- GetFunction
- GetFunctionConfiguration
- GetLogEvents
- Invoke

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Aprenda noções básicas sobre a ingestão de syslog do CloudWatch Logs usando um AWS SDK

O exemplo de código a seguir mostra como:

- Implante uma CloudFormation pilha que provisione um endpoint VPC syslog.
- Crie um grupo de registros e adicione uma política de recursos para o serviço syslog.
- Crie uma configuração de syslog com. PutSyslogConfiguration
- Liste as configurações do syslog com e sem filtros.
- Limpe excluindo a configuração do syslog, o grupo de registros e a pilha.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
class SyslogIngestionScenario:
    """Interactive scenario demonstrating CloudWatch Logs syslog ingestion."""

    STACK_NAME = "syslog-demo-stack"

    def __init__(
        self,
        cloudwatch_logs_wrapper: CloudWatchLogsWrapper,
        cf_client: boto3.client,
    ) -> None:
        """
        :param cloudwatch_logs_wrapper: A CloudWatchLogsWrapper instance.
        :param cf_client: A Boto3 CloudFormation client.
        """
        self.wrapper = cloudwatch_logs_wrapper
        self.cf_client = cf_client
        self.vpc_endpoint_id: Optional[str] = None
        self.log_group_name: Optional[str] = None
        self.log_group_arn: Optional[str] = None
        self.stack_deployed = False

    # ------------------------------------------------------------------
    # Setup phase
    # ------------------------------------------------------------------
    def deploy_cfn_stack(self) -> str:
        """
        Deploys the CloudFormation stack that provisions the VPC endpoint
        prerequisite. Waits for CREATE_COMPLETE and returns the VpcEndpointId
        output.

        :return: The VPC endpoint ID.
        """
        print("\n" + "=" * 68)
        print("Step 1: Deploy the CloudFormation prerequisite stack")
        print("=" * 68)
        print(
            f"\nDeploying stack '{self.STACK_NAME}' to create a VPC, subnet, "
            "security group, and syslog VPC endpoint.\n"
            "This can take several minutes..."
        )

        try:
```

```
            self.cf_client.create_stack(
                StackName=self.STACK_NAME,
                TemplateBody=_load_cfn_template(),
                Capabilities=["CAPABILITY_IAM"],
            )
            # Only this run's own stack should be torn down in cleanup. Mark it
            # as soon as creation is initiated so a stack that fails partway
            # (e.g. ROLLBACK) is still cleaned up.
            self.stack_deployed = True
        except ClientError as error:
            if error.response["Error"]["Code"] == "AlreadyExistsException":
                # The stack pre-existed this run. Reuse it but do NOT mark it
                # for cleanup — deleting a stack we did not create could destroy
                # infrastructure the user set up themselves.
                print(
                    f"Stack '{self.STACK_NAME}' already exists. Reusing it; it "
                    "will not be deleted during cleanup."
                )
            else:
                raise

        waiter = self.cf_client.get_waiter("stack_create_complete")
        try:
            waiter.wait(
                StackName=self.STACK_NAME,
                WaiterConfig={"Delay": 30, "MaxAttempts": 40},
            )
        except WaiterError:
            # The waiter fails both when the stack genuinely failed
            # (CREATE_FAILED, ROLLBACK_COMPLETE, ...) and, harmlessly, when the
            # stack already existed in a completed state before this run.
            # Inspect the actual status so a real failure surfaces clearly
            # instead of as a confusing "missing outputs" error later.
            status = self.cf_client.describe_stacks(StackName=self.STACK_NAME)[
                "Stacks"
            ][0]["StackStatus"]
            if status not in ("CREATE_COMPLETE", "UPDATE_COMPLETE"):
                raise RuntimeError(
                    f"CloudFormation stack '{self.STACK_NAME}' did not reach a "
                    f"successful state (status: {status}). Check the stack events
"
                    "in the CloudFormation console for the failure reason."
                )
```

```
        response = self.cf_client.describe_stacks(StackName=self.STACK_NAME)
        outputs = response["Stacks"][0].get("Outputs", list())
        vpc_endpoint_id = None
        for output in outputs:
            if output["OutputKey"] == "VpcEndpointId":
                vpc_endpoint_id = output["OutputValue"]
                break

        if vpc_endpoint_id is None:
            raise RuntimeError(
                "VpcEndpointId not found in stack outputs. "
                "Check the CloudFormation stack for errors."
            )

        self.vpc_endpoint_id = vpc_endpoint_id
        print(f"\nStack deployed. VPC Endpoint ID: {vpc_endpoint_id}")
        return vpc_endpoint_id

    def create_and_verify_log_group(self) -> None:
        """
        Prompts the user for a log group name, creates it, and verifies it
        with DescribeLogGroups.
        """
        print("\n" + "=" * 68)
        print("Step 2: Create and verify a log group")
        print("=" * 68)

        name_pattern = re.compile(r"^[a-zA-Z0-9_\-/.#]{1,512}$")

        while True:
            log_group_name = q.ask(
                "\nEnter a name for the syslog log group (e.g. /syslog/demo): "
            )
            if name_pattern.match(log_group_name):
                break
            print("Invalid name. Use 1-512 characters: a-z, A-Z, 0-9, " "_,
-, /, ., #")

        self.log_group_name = log_group_name
        self.wrapper.create_log_group(log_group_name)

        log_groups = self.wrapper.describe_log_groups(
            log_group_name_prefix=log_group_name
        )
```

```
        for lg in log_groups:
            if lg.get("logGroupName") == log_group_name:
                creation_ms = lg.get("creationTime", 0)
                creation_dt = datetime.fromtimestamp(
                    creation_ms / 1000, tz=timezone.utc
                )
                self.log_group_arn = lg.get("logGroupArn", "")
                print(f"\nLog group verified:")
                print(f"  Name:    {lg['logGroupName']}")
                print(f"  ARN:     {lg.get('arn', 'N/A')}")
                print(f"  Created: {creation_dt.strftime('%Y-%m-%d %H:%M:%S
UTC')}")
                return

        print(f"\nWARNING: Log group '{log_group_name}' not found in describe
results.")

    def add_resource_policy(self) -> None:
        """
        Adds a resource policy to the log group so the syslog service can
        write to it.
        """
        print("\n" + "=" * 68)
        print("Step 3: Add a resource policy for syslog ingestion")
        print("=" * 68)

        if self.log_group_arn is None or self.vpc_endpoint_id is None:
            print(
                "WARNING: Cannot add resource policy - missing log group ARN or
VPC endpoint."
            )
            return

        self.wrapper.put_resource_policy(
            log_group_arn=self.log_group_arn,
            vpc_endpoint_id=self.vpc_endpoint_id,
        )
        print(
            "\nResource policy created. The "
            "syslog.logs.amazonaws.com service principal can now call "
            "logs:PutLogEvents and logs:CreateLogStream on the log group."
        )

    # ---------------------------------------------------------------------
```

```
    # Create syslog configuration phase
    # ------------------------------------------------------------------
    def create_syslog_configuration(self) -> None:
        """
        Creates a syslog configuration that associates the VPC endpoint with
        the log group.
        """
        print("\n" + "=" * 68)
        print("Step 4: Create the syslog configuration")
        print("=" * 68)

        print(
            "\nAssociating the VPC endpoint with the log group so that syslog "
            "messages arriving at the endpoint are stored in the log group."
        )
        print(
            "CloudWatch Logs parses common syslog formats (RFC 5424, "
            "RFC 3164, Cisco FTD/ASA) and extracts fields such as "
            "facility, severity, hostname, and appName.\n"
        )

        self.wrapper.put_syslog_configuration(
            log_group_identifier=self.log_group_name,
            vpc_endpoint_id=self.vpc_endpoint_id,
        )

        print(
            f"\nSyslog ingestion is now enabled for log group "
            f"'{self.log_group_name}' through VPC endpoint "
            f"'{self.vpc_endpoint_id}'."
        )
        print("\nSupported transport options:")
        print("  - TCP with TLS on port 6514 (recommended)")
        print("  - Plaintext TCP on port 1514")
        print("  - UDP on port 514")

    # ------------------------------------------------------------------
    # List syslog configurations phase
    # ------------------------------------------------------------------
    def _display_configurations(self, configs: list) -> None:
        """Helper to display a list of syslog configurations."""
        if not configs:
            print("  (no configurations found)")
            return
```

```
        for cfg in configs:
            created_ms = cfg.get("createdAt", 0)
            created_dt = datetime.fromtimestamp(created_ms / 1000,
tz=timezone.utc)
            print(f"  Log Group ARN:    {cfg.get('logGroupArn', 'N/A')}")
            print(f"  VPC Endpoint ID:  {cfg.get('vpcEndpointId', 'N/A')}")
            print(f"  Source Type:      {cfg.get('sourceType', 'N/A')}")
            print(f"  Created At:       {created_dt.strftime('%Y-%m-%d %H:%M:%S
UTC')}")
            print()

    def list_all_configurations(self) -> None:
        """Lists all syslog configurations in the account (no filter)."""
        print("\n" + "=" * 68)
        print("Step 5a: List all syslog configurations (no filter)")
        print("=" * 68)

        configs = self.wrapper.list_syslog_configurations()
        print(f"\nAll syslog configurations ({len(configs)} found):\n")
        self._display_configurations(configs)

    def list_configurations_by_log_group(self) -> None:
        """Lists syslog configurations filtered by the scenario's log group."""
        print("\n" + "=" * 68)
        print("Step 5b: List configurations filtered by log group")
        print("=" * 68)

        configs = self.wrapper.list_syslog_configurations(
            log_group_identifier=self.log_group_name,
        )
        print(
            f"\nConfigurations for log group '{self.log_group_name}' "
            f"({len(configs)} found):\n"
        )
        self._display_configurations(configs)

    def list_configurations_by_vpc_endpoint(self) -> None:
        """Lists syslog configurations filtered by VPC endpoint."""
        print("\n" + "=" * 68)
        print("Step 5c: List configurations filtered by VPC endpoint")
        print("=" * 68)

        configs = self.wrapper.list_syslog_configurations(
            vpc_endpoint_id=self.vpc_endpoint_id,
```

```
        )
        print(
            f"\nConfigurations for VPC endpoint '{self.vpc_endpoint_id}' "
            f"({len(configs)} found):\n"
        )
        self._display_configurations(configs)

    # ------------------------------------------------------------------
    # Cleanup phase
    # ------------------------------------------------------------------
    def cleanup(self, prompt: bool = True) -> None:
        """
        Cleans up the resources created during the scenario. Tolerates
        resources that were never created or are already gone, so it is safe to
        call after a partial failure.

        :param prompt: When True (the normal end-of-scenario path), ask the user
            to confirm before deleting. When False (best-effort cleanup after an
            error), skip the prompt and delete whatever was created.
        """
        print("\n" + "=" * 68)
        print("Cleanup")
        print("=" * 68)

        if prompt:
            do_cleanup = q.ask(
                "\nDo you want to delete all resources created during this "
                "scenario? (y/n) ",
                q.is_yesno,
            )
            # q.is_yesno converts the answer to a bool (True when the user
            # answered 'y'); check it directly rather than calling string
methods.
            if not do_cleanup:
                print("Skipping cleanup. Resources remain in your account.")
                return
        else:
            print(
                "\nAn error occurred; attempting best-effort cleanup of any "
                "resources that were created..."
            )

        # Delete syslog configuration
        if self.log_group_name and self.vpc_endpoint_id:
```

```
            print(
                f"\nDeleting syslog configuration for log group "
                f"'{self.log_group_name}'..."
            )
            self.wrapper.delete_syslog_configuration(
                log_group_identifier=self.log_group_name,
                vpc_endpoint_id=self.vpc_endpoint_id,
            )
            print(
                "  After deletion, syslog data is no longer ingested through "
                "the VPC endpoint into this log group."
            )

            # Verify deletion
            configs = self.wrapper.list_syslog_configurations(
                log_group_identifier=self.log_group_name,
            )
            if not configs:
                print("  Verified: configuration has been removed.")
            else:
                print(
                    f"  WARNING: Still found {len(configs)} configuration(s). "
                    "They may take a moment to be fully removed."
                )

        # Delete log group
        if self.log_group_name:
            print(f"\nDeleting log group '{self.log_group_name}'...")
            self.wrapper.delete_log_group(self.log_group_name)
            print(
                "  The log group and all archived log events have been "
                "permanently deleted."
            )

        # Delete CloudFormation stack
        if self.stack_deployed:
            print(f"\nDeleting CloudFormation stack '{self.STACK_NAME}'...")
            try:
                self.cf_client.delete_stack(StackName=self.STACK_NAME)
                waiter = self.cf_client.get_waiter("stack_delete_complete")
                waiter.wait(
                    StackName=self.STACK_NAME,
                    WaiterConfig={"Delay": 30, "MaxAttempts": 40},
                )
```

```
                print("  Stack deleted successfully.")
            except (ClientError, WaiterError) as error:
                logger.warning("Stack deletion issue: %s", error)
                print(
                    f"  WARNING: Stack deletion may still be in progress. "
                    f"Check the CloudFormation console."
                )

    # ------------------------------------------------------------------
    # Run the full scenario
    # ------------------------------------------------------------------
    def run_scenario(self) -> None:
        """Runs all phases of the syslog ingestion scenario."""
        print("\n" + "=" * 68)
        print("CloudWatch Logs Syslog Ingestion Scenario")
        print("=" * 68)
        print(
            "\nThis scenario demonstrates CloudWatch Logs managed syslog "
            "ingestion, which lets you route syslog data from firewalls, "
            "routers, switches, and Linux servers directly into CloudWatch "
            "Logs through a VPC endpoint - without running a separate "
            "collection tier.\n"
        )

        succeeded = False
        try:
            # Setup
            self.deploy_cfn_stack()
            self.create_and_verify_log_group()
            self.add_resource_policy()

            # Create syslog configuration
            self.create_syslog_configuration()

            q.ask("\nPress Enter to continue to listing configurations...")

            # List syslog configurations
            self.list_all_configurations()
            self.list_configurations_by_log_group()
            self.list_configurations_by_vpc_endpoint()
            succeeded = True

            # On success, prompt before cleaning up.
            self.cleanup(prompt=True)
```

```
        finally:
            # On failure, run best-effort cleanup without prompting so that
            # partially-created resources are not left behind.
            if not succeeded:
                self.cleanup(prompt=False)

        print("\n" + "=" * 68)
        print("Scenario complete!")
        print("=" * 68)
        print(
            "\nYou have successfully demonstrated CloudWatch Logs managed "
            "syslog ingestion:\n"
            "  - Deployed a CloudFormation stack with a syslog VPC endpoint.\n"
            "  - Created a log group and authorized the syslog service.\n"
            "  - Associated a syslog configuration with PutSyslogConfiguration.
\n"
            "  - Listed configurations with and without filters.\n"
            "  - Cleaned up all resources.\n"
        )
```

Crie uma classe que envolva as operações de CloudWatch registros.

```
class CloudWatchLogsWrapper:
    """Encapsulates Amazon CloudWatch Logs actions."""

    def __init__(self, logs_client: boto3.client) -> None:
        """
        Initializes the CloudWatchLogsWrapper with a CloudWatch Logs client.

        :param logs_client: A Boto3 CloudWatch Logs client.
        """
        self.logs_client = logs_client

    @classmethod
    def from_client(cls) -> "CloudWatchLogsWrapper":
        """
        Creates a CloudWatchLogsWrapper instance with a default CloudWatch Logs
 client.

        :return: An instance of CloudWatchLogsWrapper.
```

```
        """
        logs_client = boto3.client("logs")
        return cls(logs_client)
```

- Para ver detalhes da API, consulte os tópicos a seguir na Referência da API do SDK da AWS para Python (Boto3).
  - CreateLogGroup
  - DeleteLogGroup
  - DeleteSyslogConfiguration
  - DescribeLogGroups
  - ListSyslogConfigurations
  - PutResourcePolicy
  - PutSyslogConfiguration

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use CloudWatch Logs para executar uma consulta grande

Os exemplos de código a seguir mostram como usar o CloudWatch Logs para consultar mais de 10.000 registros.

.NET

SDK para .NET (v4)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Esse é o fluxo de trabalho principal que demonstra o grande cenário de consultas.

```
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.IsNullOrWhiteSpace(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($"Enter a name for the CloudFormation stack (press
Enter for default '{_stackName}'): ");
            string? input = Console.ReadLine();
            if (!string.IsNullOrWhiteSpace(input))
            {
                var regex = "[a-zA-Z][-a-zA-Z0-9]*";
                if (!Regex.IsMatch(input, regex))
                {
                    Console.WriteLine($"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.IsNullOrWhiteSpace(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.IsNullOrWhiteSpace(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();
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para .NET .
  - GetQueryResults
  - StartQuery

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Esse é o ponto de entrada.

```
// 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}`,
);
```

Essa é uma classe que divide as consultas em várias etapas, se necessário.

```
// 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;
      },
    );
  }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript .
  - GetQueryResults
  - StartQuery

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Esse arquivo invoca um módulo de exemplo para gerenciar CloudWatch consultas com mais de 10.000 resultados.

```
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()
```

Este módulo processa CloudWatch consultas com mais de 10.000 resultados.

```
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", [])
```

- Para ver detalhes da API, consulte os tópicos a seguir na Referência da API do SDK da AWS para Python (Boto3).
  - GetQueryResults
  - StartQuery

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar eventos programados para invocar uma função do Lambda

Os exemplos de código a seguir mostram como criar uma AWS Lambda função invocada por um evento EventBridge agendado pela Amazon.

Java

SDK para Java 2.x

Mostra como criar um evento EventBridge programado pela Amazon que invoca uma AWS Lambda função. Configure EventBridge para usar uma expressão cron para agendar quando a função Lambda é invocada. Neste exemplo, você cria uma função do Lambda usando a API de runtime de Java do Lambda. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma aplicação que envia uma mensagem de texto móvel para seus funcionários que os parabeniza na data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Serviços usados neste exemplo

- CloudWatch Registros
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

JavaScript

SDK para JavaScript (v3)

Mostra como criar um evento EventBridge programado pela Amazon que invoca uma AWS Lambda função. Configure EventBridge para usar uma expressão cron para agendar quando a função Lambda é invocada. Neste exemplo, você cria uma função Lambda usando a API de tempo de execução do JavaScript Lambda. Este exemplo invoca AWS serviços diferentes para realizar um caso de uso específico. Este exemplo mostra como criar uma aplicação que envia uma mensagem de texto móvel para seus funcionários que os parabeniza na data de aniversário de um ano.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Esse exemplo também está disponível no Guia do desenvolvedor do AWS SDK para JavaScript v3.

Serviços usados neste exemplo

- CloudWatch Registros
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Python

SDK para Python (Boto3)

Este exemplo mostra como registrar uma AWS Lambda função como alvo de um EventBridge evento programado da Amazon. O manipulador do Lambda grava uma mensagem amigável e os dados completos do evento no Amazon CloudWatch Logs para recuperação posterior.

- Implanta uma função do Lambda.
- Cria um evento EventBridge agendado e torna a função Lambda o alvo.
- Concede permissão para permitir a EventBridge invocação da função Lambda.
- Imprime os dados mais recentes do CloudWatch Logs para mostrar o resultado das invocações programadas.
- Limpa todos os recursos criados durante a demonstração.

Este exemplo é melhor visualizado em GitHub. Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Serviços usados neste exemplo

- CloudWatch Registros
- DynamoDB
- EventBridge
- Lambda
- Amazon SNS

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando o CloudWatch Logs com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criptografe tabelas de pesquisa em CloudWatch registros usando AWS Key Management Service

Os dados da tabela de pesquisa são sempre criptografados nos CloudWatch registros. Por padrão, o CloudWatch Logs usa criptografia do lado do servidor com o Galois/Counter Modo Padrão de Criptografia Avançada (AES-GCM) de 256 bits para criptografar dados da tabela de pesquisa em repouso. Como alternativa, você pode usar o AWS Key Management Service para essa criptografia. Se você fizer isso, a criptografia será feita usando uma AWS KMS chave. O uso da criptografia AWS KMS é habilitado no nível da tabela de pesquisa, associando uma chave KMS a uma tabela de pesquisa, seja quando você cria a tabela de pesquisa ou quando a atualiza.

> **Important**
>
> CloudWatch O Logs suporta somente chaves KMS simétricas. Não use uma chave assimétrica para criptografar os dados em suas tabelas de pesquisa. Para obter mais informações, consulte Usar chaves simétricas e assimétricas.

Depois de associar uma chave KMS a uma tabela de pesquisa, todos os dados armazenados na tabela de pesquisa são criptografados usando essa chave. CloudWatch O Logs descriptografa esses dados sempre que solicitados. CloudWatch Os registros devem ter permissões para a chave KMS sempre que dados criptografados forem solicitados.

Se, posteriormente, você desassociar uma chave KMS de uma tabela de pesquisa, o CloudWatch Logs criptografará os dados usando o método de criptografia padrão do CloudWatch Logs. No entanto, se a chave for desativada ou excluída antes de você desassociá-la, o CloudWatch Logs não conseguirá ler os dados que foram criptografados com essa chave.

Para obter informações gerais sobre como o CloudWatch Logs usa AWS KMS para criptografar dados de registro, consulteCriptografe dados de registro no CloudWatch Logs usando AWS Key Management Service.

Como o CloudWatch Logs usa AWS KMS para tabelas de pesquisa

CloudWatch O Logs usa criptografia de AWS KMS envelope para proteger os dados da tabela de pesquisa. Quando você associa uma chave KMS a uma tabela de pesquisa, o CloudWatch

Logs envia uma `GenerateDataKey` solicitação para AWS KMS. AWS KMS gera uma chave de criptografia de dados (DEK) exclusiva e retorna uma cópia em texto simples e uma cópia criptografada da DEK. CloudWatch O Logs usa a DEK de texto simples para criptografar os dados da tabela de pesquisa e, em seguida, armazena a DEK criptografada junto com os dados criptografados. O DEK de texto simples não é armazenado e é descartado da memória após o uso.

Quando o CloudWatch Logs precisa ler os dados da tabela de pesquisa, ele envia uma `Decrypt` solicitação AWS KMS com a DEK criptografada. AWS KMS descriptografa a DEK e retorna a DEK em texto simples para o CloudWatch Logs, que a usa para descriptografar os dados da tabela de pesquisa.

CloudWatch O Logs usa o seguinte contexto de criptografia ao fazer solicitações para AWS KMS:

```
{
    "aws:logs:arn": "arn:aws:logs:region:account-id:lookup-table:lookup-table-name"
}
```

Você pode usar esse contexto de criptografia nas políticas do IAM e nas políticas de AWS KMS chaves para controlar o acesso à chave KMS. Para obter mais informações, consulte AWS KMS chaves e contexto de criptografia.

Permissões obrigatórias

Para usar a AWS KMS criptografia com tabelas de pesquisa, o diretor do IAM deve ter as seguintes AWS KMS permissões na chave KMS:

- `kms:Decrypt`
- `kms:GenerateDataKey`

A `kms:Decrypt` permissão é necessária ao chamar `GetLookupTable` uma tabela de pesquisa criptografada com uma chave KMS, para que o CloudWatch Logs possa descriptografar os dados em seu nome. A `kms:Decrypt` permissão também é necessária na chave (a chave KMS usada para criptografar a tabela de pesquisa) ao chamar `StartQuery` com uma consulta que usa o `lookup` comando em uma tabela de pesquisa criptografada. A `kms:GenerateDataKey` permissão é necessária ao chamar `CreateLookupTable` ou `UpdateLookupTable` com uma chave KMS, para que o CloudWatch Logs possa gerar uma chave de criptografia de dados para criptografar os dados da tabela de pesquisa.

Além disso, o serviço de CloudWatch registros deve ter permissão para usar a chave KMS. Você concede essas permissões adicionando uma declaração de política à política de chaves do KMS, conforme descrito na seção a seguir.

Etapa 1: criar uma AWS KMS chave

Para criar uma chave KMS simétrica, use o seguinte comando create-key:

```
aws kms create-key
```

A saída contém o ID de chave e o nome do recurso da Amazon (ARN) da chave. A seguir está um exemplo de saída:

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

Grave o ARN da chave. Você precisa dele nas etapas a seguir.

Etapa 2: definir permissões na chave do KMS

Por padrão, todas AWS KMS as chaves são privadas. Somente o proprietário do recurso pode usá-la para criptografar e descriptografar dados. Você deve conceder permissão principal ao serviço de

CloudWatch registros para usar a chave e também conceder permissão à função de chamada para usar a chave.

Primeiro, salve a política padrão para sua chave KMS `policy.json` usando o seguinte get-key-policycomando:

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

Abra o `policy.json` arquivo em um editor de texto e adicione a declaração a seguir para conceder permissão principal ao serviço de CloudWatch registros para usar a chave. Este exemplo usa uma `Condition` seção que corresponde ao contexto de criptografia para restringir a chave a uma tabela de pesquisa específica.

```
{
    "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"
        }
    }
}
```

Em seguida, adicione permissões à função que chamará os CloudWatch Logs `CreateLookupTable` ou `UpdateLookupTable` APIs. CloudWatch O Logs é usado `kms:ViaService` para fazer chamadas AWS KMS em nome do cliente. Para maiores informações, veja 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"
            ]
        }
    }
}
```

Por fim, adicione a política atualizada usando o seguinte put-key-policycomando:

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

Etapa 3: associar uma chave KMS a uma tabela de pesquisa

Você pode associar uma chave KMS a uma tabela de pesquisa ao criá-la usando a `CreateLookupTable` API ou atualizar uma tabela de pesquisa existente usando a `UpdateLookupTable` API. Ambos APIs fazem parte do AWSLogsConfigService.

Para associar a chave KMS a uma tabela de pesquisa ao criá-la

Use a `CreateLookupTable` API e especifique o `kmsKeyArn` parâmetro com o ARN da sua chave KMS:

```
aws logs create-lookup-table \
    --lookup-table-name my-lookup-table \
    --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```

Para associar a chave KMS a uma tabela de pesquisa existente

Use a `UpdateLookupTable` API e especifique o `kmsKeyArn` parâmetro com o ARN da sua chave KMS:

```
aws logs update-lookup-table \
    --lookup-table-name my-lookup-table \
    --kms-key-arn "arn:aws:kms:region:account-id:key/key-id"
```

Considerações

- Depois de associar ou desassociar uma chave KMS de uma tabela de pesquisa, pode levar até cinco minutos para que a operação entre em vigor.
- Se você revogar o acesso do CloudWatch Logs a uma chave associada ou excluir uma chave KMS associada, os dados da tabela de pesquisa criptografada nos CloudWatch Logs não poderão mais ser recuperados.
- Para realizar as etapas deste tópico, você deve ter as seguintes permissões:`kms:CreateKey`,, `kms:GetKeyPolicy``kms:PutKeyPolicy`, e as permissões de CloudWatch registros apropriadas para chamar `CreateLookupTable` ou`UpdateLookupTable`.

Segurança no Amazon CloudWatch Logs

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade aplicáveis WorkSpaces, consulte AWS Serviços no escopo do programa de conformidade AWS .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade dos dados, os requisitos da empresa e as leis e os regulamentos aplicáveis

Essa documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o Amazon CloudWatch Logs. Ele mostra como configurar o Amazon CloudWatch Logs para atender aos seus objetivos de segurança e conformidade. Você também aprenderá a usar outros AWS serviços que ajudam a monitorar e proteger seus recursos do CloudWatch Logs.

Conteúdo

- Proteção de dados no Amazon CloudWatch Logs
- Gerenciamento de identidade e acesso para Amazon CloudWatch Logs
- Validação de conformidade para Amazon CloudWatch Logs
- Resiliência no Amazon Logs CloudWatch
- Segurança da infraestrutura no Amazon CloudWatch Logs
- Usando CloudWatch registros com endpoints VPC de interface

Proteção de dados no Amazon CloudWatch Logs

> **Note**
>
> Além das informações a seguir sobre proteção geral de dados no AWS, o CloudWatch Logs também permite que você proteja dados confidenciais em eventos de log mascarando-os. Para obter mais informações, consulte Ajude a proteger dados de log confidenciais com mascaramento.

O modelo de responsabilidade AWS compartilhada O modelo se aplica à proteção de dados no Amazon CloudWatch Logs. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre privacidade de dados, consulte Perguntas frequentes sobre privacidade de dados . Para obter informações sobre proteção de dados na Europa, consulte o Centro de Regulamento Geral sobre a Proteção de Dados (RGPD).

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com Centro de Identidade do AWS IAM ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS os recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como trabalhar com CloudTrail trilhas no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.

- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com CloudWatch Logs ou outros Serviços da AWS usando o console, a API ou AWS os SDKs. AWS CLI Quaisquer dados inseridos em tags ou em campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Criptografia em repouso

CloudWatch O Logs protege os dados em repouso usando criptografia. Todos os grupos de logs são criptografados. Por padrão, o serviço de CloudWatch registros gerencia a criptografia do lado do servidor e usa a criptografia do lado do servidor com o Galois/Counter Modo Padrão de Criptografia Avançada de 256 bits () AES-GCM para criptografar dados de log em repouso.

Se você quiser gerenciar as chaves usadas para criptografar e descriptografar seus registros, use chaves. AWS KMS Para obter mais informações, consulte Criptografe dados de registro no CloudWatch Logs usando AWS Key Management Service.

Criptografia em trânsito

CloudWatch O Logs usa criptografia de ponta a ponta dos dados em trânsito. O serviço CloudWatch Logs gerencia as chaves de criptografia do lado do servidor.

Gerenciamento de identidade e acesso para Amazon CloudWatch Logs

O acesso ao Amazon CloudWatch Logs exige credenciais que AWS possam ser usadas para autenticar suas solicitações. Essas credenciais devem ter permissões para acessar AWS recursos, como recuperar dados de CloudWatch registros sobre seus recursos de nuvem. As seções a seguir fornecem detalhes sobre como você pode usar AWS Identity and Access Management (IAM) e CloudWatch registros para ajudar a proteger seus recursos controlando quem pode acessá-los:

- Autenticação
- Controle de acesso

Autenticação

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Controle de acesso

Você pode ter credenciais válidas para autenticar suas solicitações, mas, a menos que tenha permissões, não poderá criar nem acessar CloudWatch os recursos do Logs. Por exemplo, você deve ter permissões para criar streams de log, criar grupos de logs, etc.

As seções a seguir descrevem como gerenciar permissões para CloudWatch registros. Recomendamos que você leia a visão geral primeiro.

- Visão geral do gerenciamento de permissões de acesso aos seus recursos do CloudWatch Logs
- Usando políticas baseadas em identidade (políticas do IAM) para registros CloudWatch
- CloudWatch Referência de permissões de registros

Visão geral do gerenciamento de permissões de acesso aos seus recursos do CloudWatch Logs

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em Centro de Identidade do AWS IAM:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criando um perfil para um provedor de identidades de terceiros (federação) no Guia do Usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Tópicos

- CloudWatch Registra recursos e operações
- Entendendo a propriedade de recursos
- Gerenciar acesso aos recursos da
- Especificar elementos da política: ações, efeitos e entidades principais
- Especificar condições em uma política

CloudWatch Registra recursos e operações

Em CloudWatch Logs, os principais recursos são grupos de log, fluxos de log e destinos. CloudWatch Os registros não oferecem suporte a sub-recursos (outros recursos para uso com o recurso principal).

Esses recursos e sub-recursos têm Nomes de recursos da Amazon (ARNs) exclusivos associados a eles, conforme mostrado na tabela a seguir.

Tipo de atributo	Formato ARN
Grupo de registros	arn:aws:logs: ::log-group: *region account-id log_group_name*
Stream de log	arn:aws:logs: ::log-group ::log-stream: *region account-id log_group_name log-stream-name*
Destino	arn:aws:logs: ::destino: *region account-id destination_name*

Para obter mais informações sobre ARNs, consulte ARNs no Manual do usuário do IAM. Para obter informações sobre CloudWatch Logs ARNs, consulte Amazon Resource Names (ARNs) em. Referência geral da Amazon Web Services Para ver um exemplo de política que abrange CloudWatch registros, consulteUsando políticas baseadas em identidade (políticas do IAM) para registros CloudWatch.

CloudWatch O Logs fornece um conjunto de operações para trabalhar com os recursos do CloudWatch Logs. Para ver uma lista das operações disponíveis, consulte CloudWatch Referência de permissões de registros.

Entendendo a propriedade de recursos

A AWS conta possui os recursos que são criados na conta, independentemente de quem criou os recursos. Especificamente, o proprietário do recurso é a AWS conta da entidade principal (ou seja, a conta raiz, um usuário ou uma função do IAM) que autentica a solicitação de criação do recurso. Os seguintes exemplos mostram como isso funciona:

- Se você usar as credenciais da conta raiz da sua AWS conta para criar um grupo de registros, sua AWS conta será a proprietária do recurso CloudWatch Registros.
- Se você criar um usuário em sua AWS conta e conceder permissões para criar recursos de CloudWatch registros para esse usuário, o usuário poderá criar recursos de CloudWatch registros. No entanto, sua AWS conta, à qual o usuário pertence, possui os recursos do CloudWatch Logs.

- Se você criar uma função do IAM em sua AWS conta com permissões para criar recursos de CloudWatch registros, qualquer pessoa que possa assumir a função poderá criar recursos de CloudWatch registros. Sua AWS conta, à qual a função pertence, é proprietária dos recursos do CloudWatch Logs.

Gerenciar acesso aos recursos da

Uma política de permissões descreve quem tem acesso a quê. A seção a seguir explica as opções disponíveis para a criação de políticas de permissões.

> **Note**
>
> Esta seção discute o uso do IAM no contexto dos CloudWatch registros. Não são fornecidas informações detalhadas sobre o serviço IAM. Para ver a documentação completa do IAM, consulte What is IAM? no IAM User Guide. Para obter informações sobre a sintaxe e as descrições da política do IAM, consulte a IAM policy reference no IAM User Guide.

As políticas vinculadas a uma identidade do IAM são chamadas de políticas baseadas em identidade (políticas do IAM) e as políticas vinculadas a um recurso são chamadas de políticas baseadas em recursos. CloudWatch O Logs oferece suporte a políticas baseadas em identidade e políticas baseadas em recursos para destinos, que são usadas para permitir assinaturas entre contas. Para obter mais informações, consulte Cross-account Assinaturas entre regiões.

Tópicos

- Permissões de grupo de logs e Contributor Insights
- Resource-based políticas

Permissões de grupo de logs e Contributor Insights

O Contributor Insights é um recurso CloudWatch que permite analisar dados de grupos de registros e criar séries temporais que exibem dados do colaborador. É possível ver métricas sobre os principais colaboradores, o número total de colaboradores exclusivos e o uso deles. Para obter mais informações, consulte Usando o Contributor Insights para analisar High-Cardinality dados.

As operações do Contributor Insights usam o `cloudwatch:` namespace IAM. CloudWatch As permissões de registros (`logs:`) não são avaliadas quando o Contributor Insights processa dados do grupo de registros. Para obter informações detalhadas sobre o modelo de permissões, as chaves

de condição e as melhores práticas do Contributor Insights, consulte Uso de chaves condicionais para limitar o acesso dos usuários do Contributor Insights aos grupos de log no Guia do usuário da Amazon CloudWatch .

Se seus grupos de registros contiverem dados confidenciais, use as políticas de proteção de dados do CloudWatch Logs para mascarar esses dados antes que o Contributor Insights ou qualquer outro recurso possa processá-los. Para obter mais informações, consulte Ajude a proteger dados de log confidenciais com mascaramento.

Resource-based políticas

CloudWatch O Logs oferece suporte a políticas baseadas em recursos para destinos, que você pode usar para ativar assinaturas entre contas. Para obter mais informações, consulte Etapa 1: criar um destino. Os destinos podem ser criados usando a PutDestination API e você pode adicionar uma política de recursos ao destino usando a PutDestinationPolicy API. O exemplo a seguir permite que outra conta da AWS com o ID da conta 111122223333 assine seus grupos de logs no destino `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"
    }
  ]
}
```

Especificar elementos da política: ações, efeitos e entidades principais

Para cada recurso do CloudWatch Logs, o serviço define um conjunto de operações de API. Para conceder permissões para essas operações de API, o CloudWatch Logs define um conjunto de

ações que você pode especificar em uma política. Para que possam ser executadas, algumas operações da API podem exigir permissões para mais de uma ação. Para obter mais informações sobre os recursos e operações da API, consulte CloudWatch Registra recursos e operações e CloudWatch Referência de permissões de registros.

Estes são os elementos de política básicos:

- Recurso: use um nome de recurso da Amazon (ARN) para identificar o recurso ao qual a política se aplica. Para obter mais informações, consulte CloudWatch Registra recursos e operações.
- Ação: você usa palavras-chave de ação para identificar operações de recursos que deseja permitir ou negar. Por exemplo, a permissão `logs.DescribeLogGroups` permite que o usuário execute a operação `DescribeLogGroups`.
- Efeito: você especifica o efeito (permitir ou negar) quando o usuário solicita a ação específica. Se você não conceder (permitir) explicitamente acesso a um recurso, o acesso estará implicitamente negado. Você também pode negar explicitamente o acesso a um recurso, para ter certeza de que um usuário não consiga acessá-lo, mesmo que uma política diferente conceda acesso.
- Entidade principal: em políticas baseadas em identidade (políticas do IAM), o usuário ao qual a política é anexada é a entidade principal implícita. Para políticas baseadas em recursos, você especifica o usuário, a conta, o serviço ou outra entidade para a qual deseja receber permissões (aplica-se somente às políticas baseadas em recursos). CloudWatch O Logs oferece suporte a políticas baseadas em recursos para destinos.

Para saber mais sobre a sintaxe e as descrições da política do IAM, consulte a Referência de política do AWS IAM no Guia do usuário do IAM.

Para ver uma tabela que mostra todas as ações da API CloudWatch Logs e os recursos aos quais elas se aplicam, consulteCloudWatch Referência de permissões de registros.

Especificar condições em uma política

Ao conceder permissões, você pode usar a linguagem da política de acesso para especificar as condições quando uma política deve entrar em vigor. Por exemplo, é recomendável aplicar uma política somente após uma data específica. Para obter mais informações sobre como especificar condições em uma linguagem de política, consulte Condition no Guia do usuário do IAM.

Para expressar condições, você usa chaves de condição predefinidas. Para obter uma lista de chaves de contexto suportadas por cada AWS serviço e uma lista de chaves AWS de política

abrangentes, consulte Ações, recursos e chaves de condição para AWS serviços e chaves de contexto de condição AWS global.

> **Note**
>
> Você pode usar tags para controlar o acesso aos recursos do CloudWatch Logs, incluindo grupos de registros e destinos. O acesso aos fluxos de log é controlado no nível do grupo de log, devido à relação hierárquica entre grupos de log e fluxos de log. Para obter mais informações sobre como usar etiquetas para controlar o acesso, consulte Controlar acesso a recursos da Amazon Web Services usando etiquetas de recursos.

Usando políticas baseadas em identidade (políticas do IAM) para registros CloudWatch

Este tópico fornece exemplos de políticas baseadas em identidade em que um administrador de conta pode anexar políticas de permissões a identidades do IAM (ou seja, usuários, grupos e funções).

> **Important**
>
> Recomendamos que você primeiro revise os tópicos introdutórios que explicam os conceitos básicos e as opções disponíveis para gerenciar o acesso aos seus recursos do CloudWatch Logs. Para obter mais informações, consulte Visão geral do gerenciamento de permissões de acesso aos seus recursos do CloudWatch Logs.

Este tópico abrange o seguinte:

- Permissões necessárias para usar o CloudWatch console
- AWS políticas gerenciadas (predefinidas) para registros CloudWatch
- Exemplos de política gerenciada pelo cliente

Veja a seguir um exemplo de política de permissões:

JSON

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

Essa política tem uma instrução que concede permissões para criar grupos e streams de log para fazer upload de eventos de log para streams de log e listar detalhes sobre streams de log.

O caractere curinga (*) no `Resource` valor concede permissão para as ações listadas em qualquer recurso do CloudWatch Logs.

- Para limitar as permissões a ações específicas do grupo de registros (por exemplo,,`DescribeLogStreams`)`CreateLogGroup`, substitua o caractere curinga pelo ARN do grupo de registros— `arn:aws:logs:`*`us-west-2`*`:`*`123456789012`*`:log-group:`*`MyLogGroup`*
- Para limitar as permissões a ações específicas do fluxo de registros (por exemplo,`CreateLogStream`,`PutLogEvents`), substitua o caractere curinga pelo ARN do fluxo de registros— `arn:aws:logs:`*`us-west-2`*`:`*`123456789012`*`:log-group:`*`MyLogGroup`*`:*` (corresponde a todos os fluxos) ou `arn:aws:logs:`*`us-west-2`*`:`*`123456789012`*`:log-group:`*`MyLogGroup`*`:log-stream:`*`MyStream`* (fluxo específico)

Veja a referência de autorização de serviço para saber qual tipo de recurso cada API exige.

Permissões necessárias para usar o CloudWatch console

Para que um usuário trabalhe com CloudWatch registros no CloudWatch console, esse usuário deve ter um conjunto mínimo de permissões que permita ao usuário descrever outros AWS recursos em sua AWS conta. Para usar o CloudWatch Logs no CloudWatch console, você precisa ter permissões dos seguintes serviços:

- CloudWatch
- CloudWatch Registros
- OpenSearch Serviço
- IAM
- Kinesis
- Lambda
- Amazon S3

Se você criar uma política do IAM que seja mais restritiva que as permissões mínimas necessárias, o console do não funcionará como pretendido para os usuários com essa política do IAM. Para garantir que esses usuários ainda possam usar o CloudWatch console, anexe também a política `CloudWatchReadOnlyAccess` gerenciada ao usuário, conforme descrito emAWS políticas gerenciadas (predefinidas) para registros CloudWatch.

Você não precisa conceder permissões mínimas de console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a CloudWatch Logs API.

O conjunto completo de permissões necessárias para trabalhar com o CloudWatch console para um usuário que não está usando o console para gerenciar assinaturas de registro é:

- cloudwatch: GetMetricData
- cloudwatch: ListMetrics
- registros: CancelExportTask
- registros: CreateExportTask
- registros: CreateLogGroup
- registros: CreateLogStream
- registros: DeleteLogGroup

- registros: DeleteLogStream
- registros: DeleteMetricFilter
- registros: DeleteQueryDefinition
- registros: DeleteRetentionPolicy
- registros: DeleteSubscriptionFilter
- registros: DescribeExportTasks
- registros: DescribeLogGroups
- registros: DescribeLogStreams
- registros: DescribeMetricFilters
- registros: DescribeQueryDefinitions
- registros: DescribeQueries
- registros: DescribeSubscriptionFilters
- registros: FilterLogEvents
- registros: GetLogEvents
- registros: GetLogGroupFields
- registros: GetLogRecord
- registros: GetQueryResults
- registros: PutMetricFilter
- registros: PutQueryDefinition
- registros: PutRetentionPolicy
- registros: StartQuery
- registros: StopQuery
- registros: PutSubscriptionFilter
- registros: TestMetricFilter

Para um usuário que também usará o console para gerenciar assinaturas de log, as seguintes permissões também são necessárias:

- Sim: DescribeElasticsearchDomain
- Sim: ListDomainNames

- Eu sou: AttachRolePolicy
- Eu sou: CreateRole
- Eu sou: GetPolicy
- Eu sou: GetPolicyVersion
- Eu sou: GetRole
- Eu sou: ListAttachedRolePolicies
- Eu sou: ListRoles
- cinesia: DescribeStreams
- cinesia: ListStreams
- lambda: AddPermission
- lambda: CreateFunction
- lambda: GetFunctionConfiguration
- lambda: ListAliases
- lambda: ListFunctions
- lambda: ListVersionsByFunction
- lambda: RemovePermission
- s3: ListBuckets

AWS políticas gerenciadas (predefinidas) para registros CloudWatch

AWS aborda muitos casos de uso comuns fornecendo políticas de IAM autônomas que são criadas e administradas pela AWS. As políticas gerenciadas concedem permissões necessárias para casos de uso comuns, de maneira que você possa evitar a necessidade de investigar quais permissões são necessárias. Para obter mais informações, consulte Políticas gerenciadas pela AWS no Guia do usuário do IAM.

As seguintes políticas AWS gerenciadas, que você pode associar a usuários e funções em sua conta, são específicas dos CloudWatch registros:

- CloudWatchLogsFullAccess— Concede acesso total aos CloudWatch registros.
- CloudWatchLogsReadOnlyAccess— Concede acesso somente para CloudWatch leitura aos registros.

CloudWatchLogsFullAccess

A CloudWatchLogsFullAccess política concede acesso total aos CloudWatch registros. A política inclui as `cloudwatch:GenerateQueryResultsSummary` permissões `cloudwatch:GenerateQuery` e, para que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de uma solicitação em linguagem natural. Para ver o conteúdo completo da política, consulte o Guia CloudWatchLogsFullAccess de referência de políticas AWS gerenciadas.

CloudWatchLogsReadOnlyAccess

A CloudWatchLogsReadOnlyAccess política concede acesso somente para CloudWatch leitura aos registros. Ela inclui as `cloudwatch:GenerateQueryResultsSummary` permissões `cloudwatch:GenerateQuery` e, para que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de uma solicitação em linguagem natural. Para ver o conteúdo completo da política, consulte o Guia CloudWatchLogsReadOnlyAccess de referência de políticas AWS gerenciadas.

CloudWatchOpenSearchDashboardsFullAccess

A CloudWatchOpenSearchDashboardsFullAccess política concede acesso para criar, gerenciar e excluir integrações com o OpenSearch Service e para criar painéis de exclusão e gerenciamento de registros vendidos nessas integrações.

Para ver o conteúdo completo da política, consulte o Guia CloudWatchOpenSearchDashboardsFullAccess de referência de políticas AWS gerenciadas.

CloudWatchOpenSearchDashboardAccess

A CloudWatchOpenSearchDashboardAccess política concede acesso para visualizar painéis de registros vendidos criados com Amazon OpenSearch Service análises.

Para ver o conteúdo completo da política, consulte o Guia CloudWatchOpenSearchDashboardAccess de referência de políticas AWS gerenciadas.

CloudWatchLogsCrossAccountSharingConfiguration

A CloudWatchLogsCrossAccountSharingConfiguration política concede acesso para criar, gerenciar e visualizar links do Observability Access Manager para compartilhar recursos de CloudWatch registros entre contas. Para obter mais informações, consulte Observabilidade entre contas do CloudWatch .

Para ver o conteúdo completo da política, consulte o Guia CloudWatchLogsCrossAccountSharingConfiguration de referência de políticas AWS gerenciadas.

CloudWatchLogsAPIKeyAccess

A CloudWatchLogsAPIKeyAccess política permite a autenticação da chave da API CloudWatch Logs e a ingestão criptografada de registros. Essa política concede permissões para autenticar usando tokens portadores e gravar eventos de registro nos CloudWatch registros, com AWS KMS permissões adicionais para descriptografar e gerar chaves de dados quando os registros são criptografados.

Essa política concede as seguintes permissões:

- `logs`— Permite que os diretores se autentiquem por meio de tokens portadores de chaves da API e gravem eventos de registro em fluxos de registros. CloudWatch
- `kms`— Permite que os diretores leiam os metadados das AWS KMS chaves, gerem chaves de dados para criptografia e descriptografem os dados. Essas permissões oferecem suporte a CloudWatch registros criptografados, permitindo que o serviço criptografe dados de registro usando chaves gerenciadas pelo cliente AWS KMS . O acesso é restrito às operações chamadas por meio do serviço CloudWatch Logs.

Para visualizar mais detalhes sobre a política, inclusive a versão mais recente do documento de política JSON, consulte CloudWatchLogsAPIKeyAccess no AWS Managed Policy Reference Guide.

CloudWatch Registra atualizações em AWS políticas gerenciadas

Veja detalhes sobre atualizações nas políticas AWS gerenciadas do CloudWatch Logs desde que esse serviço começou a rastrear essas alterações. Para receber alertas automáticos sobre alterações nesta página, assine o feed RSS na página de histórico do Documento de CloudWatch Registros.

Alteração	Descrição	Data
CloudWatchLogsAPIKeyAccess – Nova política.	CloudWatch Os registros adicionaram uma nova política gerenciada CloudWatchLogsAPIKeyAccess.	17 de fevereiro de 2026

Alteração	Descrição	Data
	Essa política permite a autenticação da chave da API CloudWatch Logs e a ingestão criptografada de registros , concedendo permissões para autenticar usando tokens portadores e gravar eventos de log nos Logs. CloudWatch	
CloudWatchLogsFullAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram permissões CloudWatchLogsFullAccess a. Permissões para ações de administração de observabi lidade foram adicionadas para possibilitar o acesso somente leitura aos pipelines de telemetria e às integrações de tabelas do S3.	02 de dezembro de 2025
CloudWatchLogsRead OnlyAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram permissõe s CloudWatchLogsRead OnlyAccess a. Permissões para ações de administração de observabi lidade foram adicionadas para possibilitar o acesso somente leitura aos pipelines de telemetria e às integrações de tabelas do S3.	02 de dezembro de 2025

Alteração	Descrição	Data
CloudWatchLogsFullAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram permissões CloudWatchLogsFullAccess a. As permissões para `cloudwatch:GenerateQueryResultsSummary` foram adicionadas para permitir a geração de um resumo em linguagem natural dos resultados da consulta.	20 de maio de 2025
CloudWatchLogsReadOnlyAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram permissões CloudWatchLogsReadOnlyAccess a. As permissões para `cloudwatch:GenerateQueryResultsSummary` foram adicionadas para permitir a geração de um resumo em linguagem natural dos resultados da consulta.	20 de maio de 2025

Alteração	Descrição	Data
CloudWatchLogsFullAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram permissões CloudWatchLogsFullAccess a. Permissões para Amazon OpenSearch Service e IAM foram adicionadas para permitir a integração do CloudWatch Logs com o OpenSearch Service para alguns recursos.	1.º de dezembro de 2024
CloudWatchOpenSearchDashboardsFullAccess— Nova política de IAM.	CloudWatch Os registros adicionaram uma nova política de IAM, CloudWatchOpenSearchDashboardsFullAccess .- Essa política concede acesso para criar, gerenciar e excluir integrações com o OpenSearch Service e para criar, gerenciar e excluir painéis de log vendidos nessas integrações.	1.º de dezembro de 2024

Alteração	Descrição	Data
CloudWatchOpenSearchDashboardAccess— Nova política de IAM.	CloudWatch Os registros adicionaram uma nova política de IAM, CloudWatchOpenSearchDashboardAccess .- Essa política concede acesso para visualizar painéis de registros vendidos fornecidos por. Amazon OpenSearch Service	1.º de dezembro de 2024
CloudWatchLogsFullAccess: atualizar para uma política existente.	CloudWatch Os registros adicionaram uma permissão CloudWatchLogsFullAccess a. A `cloudwatch:GenerateQuery` permissão foi adicionada para que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de um prompt de linguagem natural.	27 de novembro de 2023

Alteração	Descrição	Data
CloudWatchLogsReadOnlyAccess: atualizar para uma política existente.	CloudWatch adicionou uma permissão para CloudWatchLogsReadOnlyAccess. A `cloudwatch:GenerateQuery` permissão foi adicionada para que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de um prompt de linguagem natural.	27 de novembro de 2023
CloudWatchLogsReadOnlyAccess – atualização para uma política existente	CloudWatch Os registros adicionaram permissões CloudWatchLogsReadOnlyAccess a. As `logs:StopLiveTail` permissões `logs:StartLiveTail` e foram adicionadas para que os usuários com essa política possam usar o console para iniciar e interromper as sessões ao vivo do CloudWatch Logs. Para obter mais informações, consulte Usar o Live Tail para visualizar registros quase em tempo real.	6 de junho de 2023

Alteração	Descrição	Data
CloudWatchLogsCrossAccountSharingConfiguration – Nova política	CloudWatch Os registros adicionaram uma nova política para permitir que você gerencie links de observabilidade CloudWatch entre contas que compartilham grupos de registros de CloudWatch registros. Para obter mais informações, consulte CloudWatch observabilidade entre contas	27 de novembro de 2022
CloudWatchLogsReadOnlyAccess – atualização para uma política existente	CloudWatch Os registros adicionaram permissões CloudWatchLogsReadOnlyAccess a. As `oam:ListAttachedLinks` permissões `oam:ListSinks` e foram adicionadas para que os usuários com essa política possam usar o console para visualizar dados compartilhados de contas de origem com CloudWatch observabilidade entre contas.	27 de novembro de 2022

Exemplos de política gerenciada pelo cliente

Você pode criar suas próprias políticas personalizadas do IAM para permitir permissões para ações e recursos do CloudWatch Logs. Você pode anexar essas políticas personalizadas a usuários ou grupos do que exijam essas permissões.

Nesta seção, você pode encontrar exemplos de políticas de usuário que concedem permissões para várias ações do CloudWatch Logs. Essas políticas funcionam quando você está usando a API CloudWatch Logs, AWS os SDKs ou o. AWS CLI

Exemplos

- Exemplo 1: Permitir acesso total aos CloudWatch registros
- Exemplo 2: Permitir acesso somente para leitura aos registros CloudWatch
- Exemplo 3: Permitir acesso a um grupo de logs /fluxo de log

Exemplo 1: Permitir acesso total aos CloudWatch registros

A política a seguir permite que um usuário acesse todas as ações do CloudWatch Logs.

JSON

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

Exemplo 2: Permitir acesso somente para leitura aos registros CloudWatch

AWS fornece uma CloudWatchLogsReadOnlyAccess política que permite acesso somente para leitura aos dados do CloudWatch Logs. Esta política inclui as seguintes permissões.

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": "*"
        }
    ]
}
```

Exemplo 3: Permitir acesso a um grupo de logs /fluxo de log

CloudWatch O Logs tem dois tipos de recursos com diferentes formatos de ARN:

- Grupo de registros: `arn:aws:logs:`*`region`*`:`*`account`*`:log-group:`*`LogGroupName`*
- Transmissão de log: `arn:aws:logs:`*`region`*`:`*`account`*`:log-group:`*`LogGroupName`*`:log-stream:`*`StreamName`*

Ao escrever políticas do IAM, o formato ARN que você usa deve corresponder ao tipo de recurso que a API autoriza. Veja a referência de autorização de serviço para saber qual tipo de recurso cada API exige.

Exemplo 3a: Permitir acesso a ações em nível de grupo de registros em um grupo de registros específico

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

```
        "logs:PutRetentionPolicy",
        "logs:PutSubscriptionFilter",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:us-west-2:123456789012:log-group:SampleLogGroupName"
      }
   ]
}
```

Essas ações autorizam o tipo `log-group` de recurso. O formato ARN padrão (sem `:*`) é suportado.

Exemplo 3b: Permitir acesso a ações em nível de fluxo de log em um grupo de log específico

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:GetLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:us-west-2:123456789012:log-group:SampleLogGroupName:*"
      }
   ]
}
```

Essas ações autorizam o tipo `log-stream` de recurso. O `:*` sufixo é necessário para corresponder a todos os fluxos de registros dentro do grupo de registros ou especificar um fluxo específico com. `:log-stream:`*`StreamName`*

Exemplo 3c: política combinada para ações de grupo de registros e fluxo de registros

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
        "Action": [
          "logs:DeleteLogGroup",
          "logs:PutRetentionPolicy",
```

```
          "logs:DescribeLogStreams",
          "logs:FilterLogEvents"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:logs:us-west-2:123456789012:log-group:SampleLogGroupName"
      },
      {
        "Action": [
          "logs:CreateLogStream",
          "logs:PutLogEvents",
          "logs:GetLogEvents"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:logs:us-west-2:123456789012:log-
group:SampleLogGroupName:*"
      }
   ]
}
```

Usar etiquetas e políticas do IAM para controle no nível do grupo de logs

Você pode conceder aos usuários acesso a determinados grupos de logs enquanto os impede de acessar outros grupos de logs. Para fazer isso, etiquete seus grupos de logs e use políticas do IAM que fazem referência a essas etiquetas. Para aplicar tags a um grupo de logs, você precisa ter a permissão `logs:TagResource` ou `logs:TagLogGroup`. Isso se aplica se você estiver atribuindo tags ao grupo de logs ao criá-lo. ou atribuí-los mais tarde.

Para obter mais informações sobre como marcar grupos de logs, consulte Marque grupos de registros no Amazon CloudWatch Logs.

Ao etiquetar grupos de logs, você pode conceder uma política do IAM a um usuário para permitir o acesso a apenas os grupos de logs com determinada etiqueta. Por exemplo, a instrução de política a seguir concede acesso a apenas grupos de logs com o valor `Green` para a chave de tag `Team`.

JSON

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

```
                "logs:*"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/Team": "Green"
                }
            }
        }
    ]
}
```

As operações StopLiveTail da API StopQuery e da API não interagem com AWS os recursos no sentido tradicional. Elas não retornam dados, não colocam dados nem modificam recursos. Em vez disso, eles operam somente em uma determinada sessão final em tempo real ou em uma determinada consulta do CloudWatch Logs Insights, que não são categorizadas como recursos. Como resultado, ao especificar o campo `Resource` nas políticas do IAM para essas operações, será necessário definir o valor do campo `Resource` como `*`, como no exemplo a seguir.

JSON

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

Para obter mais informações sobre como usar instruções de política do IAM, consulte Controlar o acesso usando políticas no Manual do usuário do IAM.

CloudWatch Referência de permissões de registros

Ao configurar o Controle de acesso e escrever políticas de permissões que podem ser anexadas a uma identidade do IAM (políticas baseadas em identidade), você pode usar a tabela a seguir como referência. A tabela lista cada operação da CloudWatch Logs API e as ações correspondentes para as quais você pode conceder permissões para realizar a ação. Você especifica as ações no campo `Action` das políticas. Para o `Resource` campo, você pode especificar o ARN de um grupo de logs ou stream de logs, ou especificar `*` para representar todos os recursos do CloudWatch Logs.

Você pode usar chaves AWS de condição abrangentes em suas políticas de CloudWatch registros para expressar condições. Para obter uma lista completa AWS de chaves abrangentes, consulte Chaves de contexto de condição AWS globais e do IAM no Guia do usuário do IAM.

> **Note**
>
> Para especificar uma ação, use o prefixo `logs:` seguido do nome da operação da API. Por exemplo:`logs:CreateLogGroup`,`logs:CreateLogStream`, ou `logs:*` (para todas as ações do CloudWatch Logs).

CloudWatch Registra as operações da API e as permissões necessárias para ações

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
CancelExportTask	`logs:CancelExportTask` Necessária para cancelar uma tarefa de exportação pendente ou em execução.
CreateExportTask	`logs:CreateExportTask` Necessária para exportar dados de um grupo de logs para um bucket do Amazon S3.
CreateLogGroup	`logs:CreateLogGroup` Necessária para criar um novo grupo de logs.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
CreateLogStream	`logs:CreateLogStream` Necessária para criar um novo stream de logs em um grupo de logs.
DeleteDestination	`logs:DeleteDestination` Necessária para excluir um destino de log e desativar seus filtros de assinatura.
DeleteLogGroup	`logs:DeleteLogGroup` Necessária para excluir um grupo de logs e eventos de log arquivados associados.
DeleteLogStream	`logs:DeleteLogStream` Necessária para excluir um stream de logs e eventos de log arquivados associados.
DeleteMetricFilter	`logs:DeleteMetricFilter` Necessária para excluir um filtro de métrica associado a um grupo de logs.
DeleteQueryDefinition	`logs:DeleteQueryDefinition` Necessário para excluir uma definição de consulta salva no CloudWatch Logs Insights.
DeleteResourcePolicy	`logs:DeleteResourcePolicy` Necessário para excluir uma política CloudWatch de recursos do Logs.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
DeleteRetentionPolicy	`logs:DeleteRetentionPolicy` Necessária para excluir uma política de retenção do grupo de logs.
DeleteSubscriptionFilter	`logs:DeleteSubscriptionFilter` Necessária para excluir o filtro de assinatura associado a um grupo de logs.
DescribeDestinations	`logs:DescribeDestinations` Necessária para visualizar todos os destinos associado à conta.
DescribeExportTasks	`logs:DescribeExportTasks` Necessária para visualizar todas as tarefas de exportação associadas à conta.
DescribeLogGroups	`logs:DescribeLogGroups` Necessária para visualizar todos os grupos de logs associados à conta.
DescribeLogStreams	`logs:DescribeLogStreams` Necessária para visualizar todos os streams de logs associados a um grupo de logs.
DescribeMetricFilters	`logs:DescribeMetricFilters` Necessária para visualizar todas as métricas associadas a um grupo de logs.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
DescribeQueryDefinitions	`logs:DescribeQueryDefinitions` Necessário para ver a lista de definições de consulta salvas no CloudWatch Logs Insights.
DescribeQueries	`logs:DescribeQueries` Necessário para ver a lista de consultas do CloudWatch Logs Insights que estão agendadas, em execução ou foram executadas recentemente.
DescribeResourcePolicies	`logs:DescribeResourcePolicies` Necessário para visualizar uma lista de políticas de recursos do CloudWatch Logs.
DescribeSubscriptionFilters	`logs:DescribeSubscriptionFilters` Necessária para visualizar todos os filtros de assinatura associados a um grupo de logs.
FilterLogEvents	`logs:FilterLogEvents` Necessária para classificar eventos de log por padrão de filtros de grupos.
GetLogEvents	`logs:GetLogEvents` Necessária para recuperar eventos de log a partir de um stream de logs.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
GetLogGroupFields	`logs:GetLogGroupFields` Necessária para recuperar a lista de campos incluídos nos eventos de log em um grupo de log.
GetLogRecord	`logs:GetLogRecord` Necessário para recuperar os detalhes de um único evento de log.
GetLogObject	`logs:GetLogRecord` Necessário para buscar o conteúdo de grandes porções de eventos de log que foram ingeridos por meio da PutOpenTelemetryLogs API.
GetQueryResults	`logs:GetQueryResults` Necessário para recuperar os resultados das consultas do CloudWatch Logs Insights.
ListEntitiesForLogGroup (permissão CloudWatch somente para console)	`logs:ListEntitiesForLogGroup` Necessária para encontrar as entidades associadas a um grupo de logs. Necessári o para explorar os registros relacionados no CloudWatch console.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
ListLogGroupsForEntity (permissão CloudWatch somente para console)	`logs:ListLogGroupsForEntity` Necessária para encontrar os grupos de logs associados a uma entidade. Necessário para explorar os registros relacionados no CloudWatch console.
ListTagsLogGroup	`logs:ListTagsLogGroup` Necessária para listar as tags associadas a um grupo de log.
ListLogGroups	`logs:DescribeLogGroups` Necessária para visualizar todos os grupos de logs associados à conta.
ProcessWithPipeline (somente permissão)	`logs:ProcessWithPipeline` Necessário para processar e transformar eventos de log por meio de transformadores de tubulação antes do armazenamento. Essa permissão deve ser concedida na função do IAM passada como a função de origem do CloudWatch Logs em uma configuração de pipeline. O recurso é o ARN do grupo de registros.
PutDestination	`logs:PutDestination` Necessária para criar ou atualizar um fluxo de logs de destino (como um fluxo do Kinesis).

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
PutDestinationPolicy	`logs:PutDestinationPolicy` Necessária para criar ou atualizar uma política de acesso associada a um destino de log existente.
PutLogEvents	`logs:PutLogEvents` Necessária para carregar um lote de eventos de log para um stream de log.
PutMetricFilter	`logs:PutMetricFilter` Necessária para criar ou atualizar um filtro de métrica e associá-lo a um grupo de logs.
PutQueryDefinition	`logs:PutQueryDefinition` Necessário para salvar uma consulta no CloudWatch Logs Insights, incluindo consultas salvas com parâmetros.
PutResourcePolicy	`logs:PutResourcePolicy` Necessário para criar uma política de recursos de CloudWatch registros.
PutRetentionPolicy	`logs:PutRetentionPolicy` Necessária para definir o número de dias nos quais manter os eventos de log (retenção) em um grupo de logs.

CloudWatch Registra as operações da API	Permissões obrigatórias (ações de API)
PutSubscriptionFilter	`logs:PutSubscriptionFilter` Necessária para criar ou atualizar um filtro de assinatura e associá-lo a um grupo de logs.
StartQuery	`logs:StartQuery` Necessário para iniciar consultas do CloudWatch Logs Insights. Para executar uma consulta salva com parâmetros, você também precisa`logs:DescribeQueryDefinitions` .
StopQuery	`logs:StopQuery` Necessário para interromper uma consulta do CloudWatch Logs Insights que está em andamento.
TagLogGroup	`logs:TagLogGroup` Obrigatório para adicionar ou atualizar as tags do grupo de logs.
TestMetricFilter	`logs:TestMetricFilter` Necessária para testar um padrão de filtro em relação a uma amostra de mensagens de eventos de log.

Usando funções vinculadas a serviços para registros CloudWatch

O Amazon CloudWatch Logs usa funções vinculadas a serviços AWS Identity and Access Management (IAM). Uma função vinculada a serviços é um tipo exclusivo de função do IAM

vinculada diretamente aos CloudWatch registros. Service-linked as funções são predefinidas pelo CloudWatch Logs e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Uma função vinculada ao serviço torna a configuração de CloudWatch registros mais eficiente, pois você não precisa adicionar manualmente as permissões necessárias. CloudWatch Os registros definem as permissões de suas funções vinculadas ao serviço e, a menos que definido de outra forma, somente CloudWatch os registros podem assumir essas funções. As permissões definidas incluem a política de confiança e a política de permissões. Essa política de permissões não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros produtos que oferecem suporte a funções vinculadas a serviços, consulte Serviços da AWS que funcionam com o IAM. Procure os serviços que têm Sim na coluna Service-Linked Função. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Service-linked permissões de função para CloudWatch registros

CloudWatch O Logs usa a função vinculada ao serviço chamada. AWSServiceRoleForLogDelivery CloudWatch O Logs usa essa função vinculada ao serviço para gravar registros diretamente no Firehose. Para obter mais informações, consulte Ativar o registro de AWS serviços.

A função vinculada ao serviço AWSServiceRoleForLogDelivery confia nos seguintes serviços para assumir a função:

- `logs.amazonaws.com`

A política de permissões de função, AWSServiceRoleForLogDeliveryPolicy, permite que o CloudWatch Logs conclua as seguintes ações nos recursos especificados. Para ver o documento completo da política JSON, consulte o Guia AWSServiceRoleForLogDeliveryPolicy de referência de políticas AWS gerenciadas.

- Ação:`firehose:PutRecord`,`firehose:PutRecordBatch`, e `firehose:ListTagsForDeliveryStream` em todos os fluxos de entrega do Firehose que têm uma tag com uma `LogDeliveryEnabled` chave com um valor de. `true` Essa tag é anexada automaticamente a um stream de entrega do Firehose quando você cria uma assinatura para entregar os registros ao Firehose.
- Ação: `kms:GenerateDataKey` e `kms:Decrypt` em todas AWS KMS as teclas, mas somente quando a solicitação é feita por meio do Firehose (aplicada pela chave de `kms:ViaService`

condição definida como). `firehose.*.amazonaws.com` Essas permissões permitem que o CloudWatch Logs entregue registros aos fluxos de entrega do Firehose que usam criptografia do lado do servidor com chaves gerenciadas pelo cliente (). SSE-CMK A política de AWS KMS chaves do cliente também deve conceder acesso independente à função vinculada ao serviço.

Você deve configurar permissões para que uma entidade do IAM crie, edite ou exclua uma função vinculada ao serviço. Essa entidade pode ser um usuário, um grupo ou uma função. Para obter mais informações, consulte Permissões de Service-Linked função no Guia do usuário do IAM.

Criação de uma função vinculada ao serviço para registros CloudWatch

Não é necessário criar manualmente uma função vinculada ao serviço. Quando você configura os registros para serem enviados diretamente para um stream do Firehose na Console de gerenciamento da AWS, na ou na AWS API AWS CLI, o CloudWatch Logs cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, será possível usar esse mesmo processo para recriar o perfil em sua conta. Quando você configura novamente os registros para serem enviados diretamente para um stream do Firehose, o CloudWatch Logs cria a função vinculada ao serviço para você novamente.

Editando uma função vinculada ao serviço para registros CloudWatch

CloudWatch Os registros não permitem que você edite AWSServiceRoleForLogDelivery ou qualquer outra função vinculada ao serviço depois de criá-la. Não é possível alterar o nome da função, pois várias entidades podem fazer referência a ela. No entanto, você poderá editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editando uma Service-Linked função no Guia do usuário do IAM.

Excluindo uma função vinculada ao serviço para registros CloudWatch

Se você não precisar mais usar um atributo ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-lo. Dessa forma, você não tem uma entidade não utilizada que não seja monitorada ativamente ou mantida. No entanto, você deve limpar os recursos de sua função vinculada ao serviço antes de exclui-la manualmente.

> **Note**
>
> Se o serviço CloudWatch Logs estiver usando a função quando você tentar excluir os recursos, a exclusão poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Para excluir CloudWatch os recursos do Logs usados pelo AWSServiceRoleForLogDelivery perfil vinculado ao serviço

- Interrompa o envio de logs diretamente para os fluxos do Firehose.

Como excluir manualmente o perfil vinculado ao serviço usando o IAM

Use o console do IAM AWS CLI, o ou a AWS API para excluir a função AWSServiceRoleForLogDelivery vinculada ao serviço. Para obter mais informações, consulte Excluindo uma função Service-Linked

Regiões com suporte para funções CloudWatch vinculadas ao serviço Logs

CloudWatch O Logs oferece suporte ao uso de funções vinculadas ao serviço em todas as AWS regiões em que o serviço está disponível. Para obter mais informações, consulte Regiões de CloudWatch registros e endpoints.

CloudWatch Registra atualizações em AWS Funções vinculadas ao serviço

Veja detalhes sobre as atualizações da função vinculada ao AWS serviço do CloudWatch Logs desde que esse serviço começou a rastrear essas alterações. Para receber alertas automáticos sobre alterações nesta página, assine o feed RSS na página de histórico do Documento de CloudWatch Registros.

Alteração	Descrição	Data
AWSServiceRoleForLogDelivery política de função vinculada a serviços —	CloudWatch Registros adicionados `kms:GenerateDataKey` e `kms:Decrypt` permissões à	15 de maio de 2026

Alteração	Descrição	Data
atualização para uma política existente	política do IAM associadas à função AWSServiceRoleForLogDelivery vinculada ao serviço. Essas permissões têm como escopo a chave de `kms:ViaService` condição para permitir o uso somente por meio do Firehose. Essa alteração permite que CloudWatch os registros entreguem registros aos fluxos de entrega do Firehose que usam criptografia do lado do servidor com chaves gerenciadas pelo cliente (). SSE-CMK	
AWSServiceRoleForLogDelivery política de função vinculada a serviços — atualização para uma política existente	CloudWatch Os registros alteraram as permissões na política do IAM associada à função AWSServiceRoleForLogDelivery vinculada ao serviço. Foi feita a seguinte alteração: • A chave de condição `firehose:ResourceTag/LogDeliveryEnabled": "true"` foi alterada para `aws:ResourceTag/LogDeliveryEnabled": "true"` .	15 de julho de 2021

Alteração	Descrição	Data
CloudWatch Os registros começaram a rastrear as alterações	CloudWatch Os registros começaram a rastrear as alterações em suas políticas AWS gerenciadas.	10 de junho de 2021

Validação de conformidade para Amazon CloudWatch Logs

Nossa nova experiência AWS de inscrição não foi projetada para cargas de trabalho regulamentadas. Se você estiver usando nossa nova experiência de AWS inscrição, mas quiser usá-la AWS para cargas de trabalho regulamentadas, você pode se inscrever AWS (avançado) ou ativar recursos avançados para seu AWS ambiente.

Para saber se um AWS service (Serviço da AWS) está dentro do escopo de programas de conformidade específicos, consulte Serviços da AWS Escopo por Programa de Conformidade e escolha o programa de conformidade no qual você está interessado. Para obter informações gerais, consulte Programas de AWS conformidade Programas .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixando relatórios em AWS Artifact .

Sua responsabilidade de conformidade durante o uso Serviços da AWS é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentações aplicáveis. Para obter mais informações sobre sua responsabilidade de conformidade ao usar Serviços da AWS, consulte a Documentação AWS de segurança.

Resiliência no Amazon Logs CloudWatch

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. As regiões fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, que são conectadas com baixa latência, alta throughput e redes altamente redundantes. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Segurança da infraestrutura no Amazon CloudWatch Logs

Como um serviço gerenciado, o Amazon CloudWatch Logs é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar CloudWatch os registros pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Usando CloudWatch registros com endpoints VPC de interface

Se você usa a Amazon Virtual Private Cloud (Amazon VPC) para hospedar seus AWS recursos, você pode estabelecer uma conexão privada entre sua VPC e Logs. CloudWatch Você pode usar essa conexão para enviar registros para o CloudWatch Logs sem enviá-los pela Internet. CloudWatch O Logs é compatível com IPv4 VPC endpoints em todas as regiões e com IPv6 endpoints em todas as regiões.

O Amazon VPC é um AWS serviço que você pode usar para lançar AWS recursos em uma rede virtual que você define. Com a VPC, você tem controle sobre as configurações de rede, como o intervalo de endereços IP, sub-redes, tabelas de rotas e gateways de rede. Para conectar sua VPC ao CloudWatch Logs, você define uma interface VPC endpoint para Logs. CloudWatch Esse tipo de endpoint permite que você conecte a VPC aos serviços da AWS . O endpoint fornece conectividade confiável e escalável aos CloudWatch registros sem exigir um gateway de internet, instância de tradução de endereços de rede (NAT) ou conexão VPN. Para obter mais informações, consulte O que é a Amazon VPC? no Manual do usuário da Amazon VPC.

Os endpoints VPC da Interface são alimentados por AWS PrivateLink uma AWS tecnologia que permite a comunicação privada entre AWS serviços usando uma interface de rede elástica com

endereços IP privados. Para obter mais informações, consulte Novo — AWS PrivateLink para AWS serviços.

As etapas a seguir são para usuários da Amazon VPC. Para obter mais informações, consulte Conceitos básicos no Guia do usuário da Amazon VPC.

Disponibilidade

CloudWatch Atualmente, o Logs oferece suporte a VPC endpoints em todas as AWS regiões, incluindo as regiões. AWS GovCloud (US)

Criação de um VPC endpoint para registros CloudWatch

Para começar a usar o CloudWatch Logs com sua VPC, crie uma interface VPC endpoint para Logs. CloudWatch O serviço a ser escolhido é com.amazonaws. ***Region***.registros. Para se conectar a um endpoint FIPS, o serviço a ser escolhido é. `com.amazonaws.Region.logs-fips` Você não precisa alterar nenhuma configuração dos CloudWatch registros. Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário da Amazon VPC.

Alguns CloudWatch registros APIs, como StartLiveTail e GetLogObject, são hospedados em um endpoint e um endpoint VPC diferentes:. `stream-logs.Region.amazonaws.com` Para criar uma interface VPC endpoint para eles APIs, o serviço a ser escolhido é com.amazonaws. ***Region***.stream-logs. Para se conectar a um endpoint FIPS, o serviço a ser escolhido é. `com.amazonaws.Region.stream-logs-fips`

Testando a conexão entre sua VPC e Logs CloudWatch

Depois de criar o endpoint, você pode testar a conexão.

Para testar a conexão entre sua VPC e seu CloudWatch endpoint de registros

1. Conecte-se a uma instância do Amazon EC2 que reside na VPC. Para obter mais informações sobre a conexão, consulte Conecte-se à sua instância do Linux ou Conexão com sua instância Windows na documentação do Amazon EC2.
2. Na instância, use o AWS CLI para criar uma entrada de registro em um dos seus grupos de registros existentes.

   Primeiro, crie um arquivo JSON com um evento de log. O timestamp deve ser especificado como o número de milissegundos após 1º de janeiro de 1970 00:00:00 UTC.

```
[
  {
    "timestamp": 1533854071310,
    "message": "VPC Connection Test"
  }
]
```

Em seguida, use o comando `put-log-events` para criar a entrada de log:

```
aws logs put-log-events --log-group-name LogGroupName --log-stream-
name LogStreamName --log-events file://JSONFileName
```

Se a resposta ao comando incluir `nextSequenceToken`, o comando terá sido bem-sucedido e o VPC endpoint estará funcionando.

Controle do acesso ao seu endpoint CloudWatch VPC do Logs

Uma política de endpoint da VPC é uma política de recursos do IAM que você anexa a um endpoint quando cria ou modifica o endpoint. Se você não anexar uma política quando criar um endpoint, anexaremos uma política padrão que permita o acesso total ao serviço. Uma política de endpoint não substitui as políticas do IAM nem as políticas específicas do serviço. É uma política separada para controlar o acesso do endpoint ao serviço especificado.

Políticas de endpoint devem ser gravadas em formato JSON.

Para obter mais informações, consulte Controlar o acesso a serviços com VPC endpoints no Manual do usuário da Amazon VPC.

Veja a seguir um exemplo de uma política de endpoint para CloudWatch Logs. Essa política permite que os usuários se conectem ao CloudWatch Logs por meio da VPC para criar fluxos de registros e enviar CloudWatch registros para o Logs, além de impedir que eles realizem outras CloudWatch ações do Logs.

```
{
  "Statement": [
    {
      "Sid": "PutOnly",
      "Principal": "*",
      "Action": [
```

```
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Para modificar a política de VPC endpoint para Logs CloudWatch

1. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel de navegação, escolha Endpoints.
3. Se você ainda não criou o endpoint para CloudWatch Logs, escolha Create Endpoint. Em seguida, selecione com.amazonaws. ***Region***.logs e escolha Criar endpoint.
4. Selecione o com.amazonaws. ***Region***.logs endpoint e escolha a guia Política na metade inferior da tela.
5. Selecione Edit policy (Editar política) e faça as alterações na política.

Compatibilidade com chaves de contexto da VPC

CloudWatch O Logs oferece suporte às chaves de `aws:SourceVpce` contexto `aws:SourceVpc` e de contexto que podem limitar o acesso a VPCs endpoints de VPC específicos ou específicos. Essas chaves funcionam somente quando o usuário está usando VPC endpoints. Para obter mais informações, consulte Chaves disponíveis para alguns serviços no Manual do usuário do IAM.

Operações da API CloudWatch Logging Logs e do console em AWS CloudTrail

O Amazon CloudWatch Logs é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou um AWS service (Serviço da AWS). CloudTrail captura chamadas de API para CloudWatch registros como eventos. As chamadas capturadas incluem chamadas do console do CloudWatch Logs e chamadas de código para as operações da API CloudWatch Logs. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita ao CloudWatch Logs, o endereço IP do qual a solicitação foi feita, quando foi feita e detalhes adicionais.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar o seguinte:

- Se a solicitação foi feita com credenciais de usuário raiz ou credenciais de usuário.
- Se a solicitação foi feita em nome de um usuário do Centro de Identidade do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS service (Serviço da AWS).

CloudTrail está ativo Conta da AWS quando você cria a conta e você tem acesso automático ao histórico de CloudTrail eventos. O histórico de CloudTrail eventos fornece um registro visível, pesquisável, baixável e imutável dos últimos 90 dias de eventos de gerenciamento registrados em um. Região da AWS Para obter mais informações, consulte Trabalhando com o histórico de CloudTrail eventos no Guia AWS CloudTrail do usuário. Não há CloudTrail cobrança pela visualização do histórico de eventos.

Para um registro contínuo dos eventos dos Conta da AWS últimos 90 dias, crie uma trilha ou um armazenamento de dados de eventos do CloudTrailLake.

CloudTrail trilhas

Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Todas as trilhas criadas usando o Console de gerenciamento da AWS são multirregionais. Só é possível criar uma trilha de região única ou de várias regiões usando a AWS CLI. É recomendável criar uma trilha multirregional porque você captura todas as atividades Regiões da AWS em sua conta.

Ao criar uma trilha de região única, é possível visualizar somente os eventos registrados na Região da AWS da trilha. Para obter mais informações sobre trilhas, consulte Criar uma trilha para a Conta da AWS e Criar uma trilha para uma organização no Guia do usuário do AWS CloudTrail .

Você pode entregar uma cópia dos seus eventos de gerenciamento em andamento para o bucket do Amazon S3 sem nenhum custo CloudTrail criando uma trilha. No entanto, existem taxas de armazenamento do Amazon S3. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços. Para receber informações sobre a definição de preços do Amazon S3, consulte Definição de preços do Amazon S3.

CloudTrail Armazenamentos de dados de eventos em Lake

CloudTrail O Lake permite que você execute consultas baseadas em SQL em seus eventos. CloudTrail O Lake converte eventos existentes no formato JSON baseado em linhas para o formato Apache ORC. O ORC é um formato colunar de armazenamento otimizado para recuperação rápida de dados. Os eventos são agregados em armazenamentos de dados de eventos, que são coleções imutáveis de eventos baseados nos critérios selecionados com a aplicação de seletores de eventos avançados. Os seletores que aplicados a um armazenamento de dados de eventos controlam quais eventos persistem e estão disponíveis para consulta. Para obter mais informações sobre o CloudTrail Lake, consulte Trabalhando com o AWS CloudTrail Lake no Guia AWS CloudTrail do Usuário.

CloudTrail Os armazenamentos e consultas de dados de eventos em Lake incorrem em custos. Ao criar um armazenamento de dados de eventos, você escolhe a opção de preço que deseja usar para ele. A opção de preço determina o custo para a ingestão e para o armazenamento de eventos, e o período de retenção padrão e máximo para o armazenamento de dados de eventos. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços.

CloudWatch O Logs permite registrar as seguintes ações como eventos em arquivos de 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

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Informações de geração de consultas em CloudTrail

CloudTrail O registro de eventos do console do gerador de consultas também é suportado. Atualmente, o gerador de consultas é compatível com CloudWatch Logs Insights e CloudWatch Metric Insights. Nesses CloudTrail eventos, o `eventSource` é`monitoring.amazonaws.com`.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a GenerateQueryação no CloudWatch Logs Insights.


```
{
    "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"
}
```

Noções básicas sobre entradas de arquivos de log do

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

A entrada do arquivo de registro a seguir mostra que um usuário chamou a CreateExportTaskação CloudWatch Registros.

```
{
        "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"
}
```

Monitoramento com CloudWatch métricas

Você pode usar as tabelas nesta seção para analisar as métricas que o Amazon CloudWatch Logs envia para a Amazon a CloudWatch cada minuto.

CloudWatch Métricas de registros

O namespace `AWS/Logs` inclui as métricas a seguir.

Métrica	Description
`CallCount`	O número de operações de API especificadas executadas em sua conta. `CallCount` é uma métrica de uso do serviço de CloudWatch registros. Para obter mais informações, consulte CloudWatch Registra métricas de uso do serviço. Dimensões válidas: classe, recurso, serviço, tipo Estatística válida: soma Unidades: nenhuma
`DeliveryErrors`	O número de eventos de registro para CloudWatch os quais o Logs recebeu um erro ao encaminhar dados para o destino da assinatura. Se o serviço de destino retornar um erro que pode ser repetido, como uma exceção de limitação ou uma exceção de serviço que pode ser repetida (HTTP 5xx, por exemplo), o CloudWatch Logs continuará tentando entregar novamente por até 24 horas. CloudWatch Os registros não tentarão ser entregues novamente se o erro for um erro que não possa ser repetido, como ou. `AccessDeniedException` `ResourceNotFoundException` Dimensões válidas: LogGroupName, DestinationType, FilterName, PolicyLevel Estatística válida: soma

Métrica	Description
	Unidades: nenhuma
`DeliveryThrottling`	O número de eventos de registro para CloudWatch os quais o Logs foi limitado ao encaminhar dados para o destino da assinatura. Se o serviço de destino retornar um erro que pode ser repetido, como uma exceção de limitação ou uma exceção de serviço que pode ser repetida (HTTP 5xx, por exemplo), o CloudWatch Logs continuar á tentando entregar novamente por até 24 horas. CloudWatch Os registros não tentarão ser entregues novamente se o erro for um erro que não possa ser repetido, como ou. `AccessDeniedException` `ResourceNotFoundException` Dimensões válidas: LogGroupName, DestinationType, FilterName, PolicyLevel Estatística válida: soma Unidades: nenhuma
`EMFDisabledErrors`	O número de eventos de log formatados em EMF que foram ignorados devido a uma política de conta ativa do tipo METRIC_EXTRACTION_POLICY que corresponde ao grupo de registros. Para obter mais informações sobre as políticas de extração de métricas, consulte. PutAccountPolicy Dimensões válidas: `LogGroupName` Estatística válida: soma Unidades: nenhuma

Métrica	Description
`EMFParsingErrors`	O número de erros de análise encontrados durante o processamento de logs de formato métrico integrado. Esses erros ocorrem quando os logs são identificados como formato de métrica incorporada, mas não seguem o formato correto. Para obter mais informações sobre o formato de métrica incorporada, consulte Especificação: formato de métrica integrado. Dimensões válidas: `LogGroupName` Estatística válida: soma Unidades: nenhuma
`EMFValidationErrors`	O número de erros de validação encontrados durante o processamento de logs de formato de métrica integrados. Esses erros ocorrem quando as definições de métrica nos logs de formato de métrica integrados não aderem ao formato de métrica integrado e às especificações `MetricDatum` . Para obter informações sobre o formato métrico CloudWatch incorporado, consulte Especificação: formato métrico incorporado. Para obter informações sobre o tipo de dados`MetricDatum` , consulte MetricDatuma Amazon CloudWatch API Reference. **Note** Certos erros de validação podem fazer com que várias métricas em um log de EMF não sejam publicadas. Por exemplo, todas as métricas definidas com um namespace inválido serão descartadas. Dimensões válidas: `LogGroupName` Estatística válida: soma Unidades: nenhuma

Métrica	Description
`ErrorCount`	O número de operações de API executadas em sua conta que resultaram em erros. `ErrorCount` é uma métrica de uso do serviço de CloudWatch registros. Para obter mais informações, consulte CloudWatch Registra métricas de uso do serviço. Dimensões válidas: classe, recurso, serviço, tipo Estatística válida: soma Unidades: nenhuma
`ForwardedBytes`	O volume de eventos de log em bytes compactados encaminhados para o destino de inscrição. Dimensões válidas: LogGroupName, DestinationType, FilterName Estatística válida: soma Unidades: bytes
`ForwardedLogEvents`	O número de eventos de log encaminhados para o destino de inscrição. Dimensões válidas: LogGroupName, DestinationType, FilterName, PolicyLevel Estatística válida: soma Unidades: nenhuma

Métrica	Description
`IncomingBytes`	O volume de eventos de log em bytes não compactados enviados para CloudWatch o Logs. Quando usado com a dimensão `LogGroupName` , este é o volume de eventos de log em bytes descompactados carregado s no grupo de logs. Dimensões válidas: LogGroupName Estatística válida: soma Unidades: bytes
`IncomingL ogEvents`	O número de eventos de registro enviados para o CloudWatch Logs. Quando usado com a dimensão `LogGroupName` , esse é o número de eventos de log carregados no grupo de logs. Dimensões válidas: LogGroupName Estatística válida: soma Unidades: nenhuma
`LogEvents WithFindings`	O número de eventos de registro que correspondem a uma sequência de dados que você está auditando usando o recurso de proteção de dados de CloudWatch registros. Para obter mais informações, consulte Ajude a proteger dados de log confidenciais com mascaramento. Dimensões válidas: nenhuma Estatística válida: soma Unidades: nenhuma

Métrica	Description
`ThrottleCount`	O número de operações de API executadas em sua conta que foram limitadas por causa de cotas de uso. `ThrottleCount` é uma métrica de uso do serviço de CloudWatch registros. Para obter mais informações, consulte CloudWatch Registra métricas de uso do serviço. Dimensões válidas: classe, recurso, serviço, tipo Estatística válida: soma Unidades: nenhuma

Dimensões para métricas CloudWatch de registros

As dimensões que você pode usar com a maioria das métricas do CloudWatch Logs estão listadas na tabela a seguir.

Dimensão	Description
`LogGroupName`	O nome do grupo de CloudWatch registros de registros para o qual exibir métricas.
`DestinationType`	O destino da assinatura para os dados do CloudWatch Logs, que pode ser AWS Lambda Amazon Kinesis Data Streams ou Amazon Data Firehose.
`FilterName`	O nome do filtro de inscrição que está encaminhando dados do grupo de logs para o destino. O nome do filtro de assinatura é automaticamente convertido CloudWatch em ASCII e qualquer caractere não suportado é substituído por um ponto de interroga ção (?).

Dimensões métricas do filtro de assinatura

As dimensões das métricas relacionadas aos filtros de assinatura ao nível da conta estão listadas na tabela a seguir.

Dimensão	Description
`PolicyLevel`	O nível a que a política se aplica. No momento, o único valor válido para essa dimensão é `AccountPolicy`
`DestinationType`	O destino da assinatura para os dados do CloudWatch Logs, que pode ser AWS Lambda Amazon Kinesis Data Streams ou Amazon Data Firehose.
`FilterName`	O nome do filtro de inscrição que está encaminhando dados do grupo de logs para o destino. O nome do filtro de assinatura é automaticamente convertido CloudWatch em ASCII e qualquer caractere não suportado é substituído por um ponto de interroga ção (?).

Métricas e dimensões do transformador de log

CloudWatch O Logs publica as seguintes métricas do transformador de log CloudWatch no namespace. `AWS/Logs`

Métrica	Description
`TransformationErrors`	O número de erros encontrados ao transformar eventos de log com o transformador especificado. Unidade: nenhuma Estatística válida: soma
`TransformedBytes`	O volume da saída de eventos de log transformados, em bytes não compactados. Unidade: bytes Estatística válida: soma

Métrica	Description
`TransformedLogEvents`	O número de eventos de log transformados. Unidade: nenhuma Estatística válida: soma

As seguintes dimensões são usadas para as métricas do transformador.

Dimensão	Description
`LogGroupname`	Essa dimensão é usada somente para log-group-level transform adores.
`PolicyLevel`	Essa dimensão é usada somente para transformadores em nível de conta. No momento, o único valor válido para essa dimensão é `AccountPolicy`

Métricas e dimensões de centralização

Para monitoramento centralizado em várias contas e regiões, você pode usar a centralização de CloudWatch registros para consolidar dados e métricas de registros em um local central. Para obter mais informações, consulte Cross-account Centralização de registros entre regiões.

CloudWatch O Logs publica as seguintes métricas de centralização CloudWatch no namespace. `AWS/Logs` Essas métricas ajudam você a monitorar a replicação dos dados de registro das contas de origem para as contas de destino ao usar as regras de centralização de CloudWatch registros.

> **Note**
>
> CloudWatch começa a relatar métricas de centralização logo após a criação de uma regra de centralização. As métricas são publicadas com base no melhor esforço e rastreiam somente os novos eventos de registro que chegam após a criação da regra de centralização.

Métrica	Description	Publicado em
`IncomingCopiedBytes`	O volume de dados de log em bytes não compactados que foi replicado na conta de destino. Essa métrica se aplica somente aos novos eventos de log que chegam após a criação da regra de centralização. Dimensões válidas: SourceRegion, SourceAccount Estatística válida: soma Unidades: bytes	Conta de destino
`IncomingCopiedLogEvents`	O número de eventos de log que foram replicados na conta de destino. Essa métrica se aplica somente aos novos eventos de log que chegam após a criação da regra de centralização. Dimensões válidas: SourceRegion, SourceAccount Estatística válida: soma Unidades: nenhuma	Conta de destino
`OutgoingCopiedBytes`	O volume de dados de log em bytes não compactados que foi enviado das contas de origem para a conta de destino. Essa métrica se aplica somente a novos eventos de log que são copiados para o destino após a criação da regra de centralização. Dimensões válidas: DestinationRegion Estatística válida: soma	Conta de origem

Métrica	Description	Publicado em
	Unidades: bytes	
`OutgoingCopiedLogEvents`	O número de eventos de log que foram enviados das contas de origem para a conta de destino. Essa métrica se aplica somente a novos eventos de log que são copiados para o destino após a criação da regra de centralização. Dimensões válidas: DestinationRegion Estatística válida: soma Unidades: nenhuma	Conta de origem
`CentralizationError`	O número de erros encontrados durante a replicação dos dados de log. Podem ocorrer erros devido a vários motivos, como problemas de permissão da chave KMS, limites de cota de grupos de registros ou incompatibilidades de níveis de registro. Para identificar grupos ou fluxos de registros específicos que falharam na replicação e seus motivos de falha, monitore a `ErrorType` dimensão. Dimensões válidas: ErrorType Estatística válida: soma Unidades: nenhuma	Conta de destino

Métrica	Description	Publicado em
`CentralizationThrottled`	O número de vezes que o processamento da centralização foi acelerado. A limitação resulta em um processamento mais lento da centralização, mas não impede que os dados de log sejam replicados. Dimensões válidas: DestinationRegion Estatística válida: soma Unidades: nenhuma	Conta de origem

As dimensões a seguir são usadas pelas métricas de centralização.

Dimensão	Description
`SourceRegion`	A AWS região na qual os dados de log de origem se originaram.
`SourceAccount`	O ID da AWS conta de onde os dados de log de origem se originaram.
`DestinationRegion`	A AWS região para a qual os dados de registro estão sendo replicados.
`ErrorType`	O tipo de erro encontrado durante a centralização. Os possíveis valores incluem: • `LogGroupQuotaExceeded` : a conta de destino atingiu a cota do grupo de registros. • `InvalidKMS` : a chave KMS está ausente, excluída, não ativada ou é assimétrica em vez de simétrica. • `AccessDenied` : o acesso foi negado ao tentar replicar dados de log. Isso pode ocorrer devido a permissões insuficientes, como permissões incorretas da chave KMS, falta de permissões do IAM ou políticas restritivas de recursos.

Dimensão	Description
	• `LogTierMismatch` : a camada de log dos grupos de log de origem e de destino não coincide. • `InvalidLogStream` : parâmetro inválido encontrado ao criar ou gravar em um fluxo de registros, como quando o nome do fluxo de registros excede o tamanho máximo. • `InvalidLogGroup` : parâmetro inválido encontrado ao criar ou configurar um grupo de registros na conta de destino. • `DestinationEncryptionMismatch` : o grupo de registros de destino já existe com uma configuração de criptografia KMS diferente da especificada pela regra de centralização.

CloudWatch Registra métricas de uso do serviço

CloudWatch O Logs envia métricas CloudWatch que rastreiam o uso das operações da API CloudWatch Logs. Essas métricas correspondem às cotas AWS de serviço. O rastreamento dessas métricas pode ajudar a gerenciar as cotas proativamente. Para obter mais informações, consulte Integração do Service Quotas e métricas de uso.

Por exemplo, é possível rastrear a métrica `ThrottleCount` ou definir um alarme nessa métrica. Se o valor dessa métrica aumentar, você deverá considerar solicitar um aumento de cota para a operação de API que está sendo limitada. Para obter mais informações sobre CloudWatch as cotas do serviço de registros, consulteCloudWatch Registra cotas.

CloudWatch O Logs publica métricas de uso da cota de serviço a cada minuto nos namespaces `AWS/Usage` e`AWS/Logs`.

A tabela a seguir lista as métricas de uso do serviço publicadas pelo CloudWatch Logs. Essas métricas não têm uma unidade especificada. A estatística mais útil para essas métricas é `SUM`, que representa a contagem total de operações para o período de um minuto.

Cada uma dessas métricas é publicada com valores para todas as dimensões `Service`, `Class`, `Type` e `Resource`. Também são publicadas com uma única dimensão chamada `Account Metrics`. Use a dimensão `Account Metrics` para ver a soma das métricas de todas as

operações de API em sua conta. Use as outras dimensões e especifique o nome de uma operação de API para a dimensão `Resource` localizar as métricas para essa API específica.

Métricas

Métrica	Description
`CallCount`	O número de operações especificadas executadas em sua conta. `CallCount` é publicado nos namespaces `AWS/Usage` e `AWS/Logs`.
`ErrorCount`	O número de operações de API executadas em sua conta que resultaram em erros. `ErrorCount` é publicado apenas no `AWS/Logs`.
`ThrottleCount`	O número de operações de API executadas em sua conta que foram limitadas por causa de cotas de uso. `ThrottleCount` é publicado apenas no `AWS/Logs`.

Dimensões

Dimensão	Description
`Account metrics`	Use essa dimensão para obter uma soma da métrica em todos os CloudWatch registros APIs. Se quiser ver as métricas de uma API específica, use as outras dimensões listadas nesta tabela e especifique o nome da API como o valor de `Resource`.
`Service`	O nome do AWS serviço que contém o recurso. Para métricas de uso do CloudWatch Logs, o valor dessa dimensão é`Logs`.
`Class`	A classe do recurso que está sendo rastreada. CloudWatch As métricas de uso da API Logs usam essa dimensão com um valor de`None`.

Dimensão	Description
`Type`	O tipo de recurso que está sendo acompanhado. No momento, quando a dimensão `Service` é `Logs`, o único valor válido para `Type` é `API`.
`Resource`	O nome da operação da API. Os valores válidos incluem todos os nomes de operação da API listados em Actions (Ações). Por exemplo, `PutLogEvents` .

CloudWatch Registra cotas

Você pode usar a tabela nesta seção para revisar as cotas de serviço padrão, também chamadas de limites, para uma AWS conta no Amazon CloudWatch Logs. A maioria das cotas de serviço, mas não todas, estão listadas no namespace Amazon CloudWatch Logs no console Service Quotas.

> **Note**
>
> Se quiser solicitar um aumento em qualquer uma dessas cotas, consulte o procedimento nesta seção.

Nome	Padrão	Ajustá	Description
Tarefa de exportação ativa	Cada região compatível: 1	Não	O número de tarefas de exportação ativas (em execução ou pendentes) por conta
AssociateKmsKey limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas associadas de kms-key por segundo por região account/per
AssociateSourceToS3TableIntegration limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de AssociateSourceToS 3TableIntegration chamadas por segundo por account/per região
AssociateSourceToS3TableIntegration limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de AssociateSourceToS 3TableIntegration chamadas por segundo por account/per região em uma sequência

Nome	Padrão	Ajustá	Description
Tamanho do lote	Cada região com suporte: 1 megabyte	Não	O tamanho máximo do lote em MB de uma solicitação put-log-events
CancelExportTask limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de cancel-ex port-task por segundo por região account/per
CancelImportTask limite de aceleração em transações por segundo	Cada região compatível: 1	Não	O número máximo de CancelImportTask chamadas por segundo por account/per região
CancelImportTask limite de aceleraçã o em transações por segundo em uma explosão	Cada região compatível: 1	Não	O número máximo de CancelImportTask chamadas por segundo por account/per região em uma sequência
CreateExportTask limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas create-export-task por segundo por região account/per
CreateImportTask limite de aceleração em transações por segundo	Cada região compatível: 1	Não	O número máximo de CreateImportTask chamadas por segundo por account/per região
CreateImportTask limite de aceleraçã o em transações por segundo em uma explosão	Cada região compatível: 1	Não	O número máximo de CreateImportTask chamadas por segundo por account/per região em uma sequência

Nome	Padrão	Ajustá	Description
CreateLogGroup limite de aceleração em transações por segundo	Cada região compatível: 10 por segundo	Sim	O número máximo de chamadas de criação de grupos de registro por segundo por região account/per
CreateLogStream limite de aceleração em transações por segundo	Cada região compatível: 50 por segundo	Sim	O número máximo de chamadas create-log-stream por segundo por região account/per
Arquivamento de dados	Cada região compatível: 5 gigabytes	Não	O tamanho máximo em GB de arquivamento de dados gratuito
DeleteDataProtectionPolicy limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de chamadas de política de proteção de dados por segundo por região account/per
DeleteDestination limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de destino excluído por segundo por região account/per
DeleteLogGroup limite de aceleração em transações por segundo	Cada região compatível: 10 por segundo	Sim	O número máximo de chamadas de grupos de exclusão de registros por segundo por região account/per
DeleteLogStream limite de aceleração em transações por segundo	Cada região compatível: 15 por segundo	Sim	O número máximo de chamadas de delete-log-stream por segundo por região account/per

Nome	Padrão	Ajustá	Description
DeleteMetricFilter limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de delete-me tric-filter por segundo por região account/per
DeleteRetentionPolicy limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de política de exclusão e retenção por segundo por região account/per
DeleteSubscriptionFilter limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de exclusão de filtro de assinatura por segundo por região account/per
DeleteTransformer limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de delete-tr ansformador por segundo por região account/per
DescribeDestinations limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de destinos descritos por segundo por região account/per
DescribeExportTasks limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas describe-export-tasks por segundo por região account/per

Nome	Padrão	Ajustá	Description
DescribeImportTaskBatches limite de aceleração em transações por segundo	Cada região com suporte: 10	Não	O número máximo de DescribeImportTask Batches chamadas por segundo por account/per região
DescribeImportTaskBatches limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de DescribeImportTask Batches chamadas por segundo por account/per região em uma sequência
DescribeImportTasks limite de aceleração em transações por segundo	Cada região com suporte: 10	Não	O número máximo de DescribeImportTasks chamadas por segundo por account/per região
DescribeImportTasks limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de DescribeImportTasks chamadas por segundo por account/per região em uma sequência
DescribeLogGroups limite de aceleração em transações por segundo	Cada região compatível: 10 por segundo	Sim	O número máximo de chamadas de describe-log-groups por segundo por região account/per
DescribeLogStreams limite de aceleração em transações por segundo	Cada região compatível: 25 por segundo	Sim	O número máximo de chamadas describe-log-streams por segundo por região account/per

Nome	Padrão	Ajustá	Description
DescribeMetricFilters limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas describe-metric-filters por segundo por região account/per
DescribeSubscriptionFilters limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas describe-subscription-filters por segundo por região account/per
DisassociateSourceFromS3TableIntegration limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de DisassociateSourceFromS3TableIntegration chamadas por segundo por account/per região
DisassociateSourceFromS3TableIntegration limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de DisassociateSourceFromS3TableIntegration chamadas por segundo por account/per região em uma sequência
Tamanho do evento	Cada região suportada: 1.024 kilobytes	Não	O tamanho máximo do evento de logs em KB

Nome	Padrão	Ajustá	Description
FilterLogEvents limite de aceleração em transações por segundo	us-east-1: 25 por segundo ap-northeast-3: 5 por segundo ap-southeast-3: 5 por segundo ca-west-1:5 por segundo eu-central-1: 5 por segundo il-central-1:5 por segundo Cada uma das outras regiões compatíveis: 10 por segundo	Não	O número máximo de chamadas de eventos de registro de filtro por segundo por região account/per
GetDataProtectionPolicy limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de chamadas get-data-protection-policy por segundo por região account/per

Nome	Padrão	Ajustá	Description
GetLogEvents limite de aceleração em transações por segundo	us-east-2: 10 por segundo ap-northeast-3: 10 por segundo ap-southeast-3: 10 por segundo ca-west-1:10 por segundo eu-central-1: 10 por segundo eu-west-1: 10 por segundo eu-west-3: 30 por segundo il-central-1:10 por segundo Cada uma das outras regiões compatíveis: 25 por segundo	Não	O número máximo de chamadas get-log-events por segundo por região account/per
GetQueryResults limite de aceleração em transações por segundo	Cada região com suporte: 10	Não	O número máximo de chamadas get-query-results por segundo por região account/per

Nome	Padrão	Ajustá	Description
GetTransformer limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas get-trans former por segundo por região account/per
ListSourcesForS3TableIntegration limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de ListSourcesForS3Ta bleIntegration chamadas por segundo por account/ per região
ListSourcesForS3TableIntegration limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de ListSourcesForS3Ta bleIntegration chamadas por segundo por account/ per região em uma sequência
ListTagsForResource limite de aceleração em transações por segundo	Cada região compatível: 30 por segundo	Não	O número máximo de chamadas list-tags-for-resource por segundo por região account/per
ListTagsLogGroup limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas list-tags-log-group por segundo por região account/per
Limite de sessões simultâneas do Live Tail	Cada região compatível: 15	Sim	Número máximo de sessões ativas simultâne as do Live Tail por conta
Grupos de logs	Cada região compatível: 1.000.000	Sim	O número máximo de grupos de logs que uma conta pode ter

Nome	Padrão	Ajustá	Description
Grupos de logs verificados por sessão do Live Tail	Cada região com suporte: 10	Não	Número máximo de grupos de logs que podem ser verificados por sessão do Live Tail
Filtros de métricas por grupo de logs	Cada região compatível: 100	Não	O número de filtros de métricas por grupo de logs
PutBearerTokenAuthentication limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de chamadas de autentica ção put-bearer-token-p or segundo por região account/per
PutBearerTokenAuthentication limite de aceleração em transações por segundo em uma explosão	Cada região com suporte: 10	Não	O número máximo de chamadas de autentica ção put-bearer-token-por segundo por região em uma sequência account/p er
PutDataProtectionPolicy limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de chamadas put-data-protection-policy por segundo por região account/per
PutDestination limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas put-destination por segundo por região account/per

Nome	Padrão	Ajustá	Description
PutDestinationPolicy limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas put-destination-policy por segundo por região account/per
PutLogEvents limite de aceleração em transações por segundo	Cada região suportada: 5.000 por segundo	Sim	O número máximo de chamadas put-log-events por segundo por região account/per
PutMetricFilter limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas put-metric-filter por segundo por região account/per
PutRetentionPolicy limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de put-retention-policy por segundo por região account/per
PutSubscriptionFilter limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas put-subscription-filter por segundo por região account/per
PutTransformer limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de transformador put-por segundo por região account/per
Políticas de recursos	Cada região com suporte: 10	Não	O número máximo de políticas de recursos por account/per região

Nome	Padrão	Ajustá	Description
StartLiveTail limite de aceleração em transações por segundo	Cada região compatível: 5	Não	O número máximo de chamadas iniciais e diretas por segundo por região. account/per Esse limite é aplicável de maneira independente ao console e à API
StartQuery limite de aceleração em transações por segundo	Cada região com suporte: 10	Não	O número máximo de chamadas de início de consulta por segundo por região account/per
Filtros de assinatura por grupo de logs	Cada região compatível: 5	Não	O número de filtros de assinatura por grupo de logs
TagLogGroup limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de tag-log-group por segundo por região account/per
TagResource limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de recursos de tag por segundo por região account/per
TestMetricFilter limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de filtro métrico de teste por segundo por região account/per

Nome	Padrão	Ajustá	Description
TestTransformer limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de transform adores de teste por segundo por região account/per
UntagLogGroup limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas untag-log-group por segundo por região account/per
UntagResource limite de aceleração em transações por segundo	Cada região compatível: 5 por segundo	Não	O número máximo de chamadas de recursos não marcados por segundo por região account/per
detectores de anomalias de logs	Cada região com suporte: 500	Sim	Número máximo de detectores de anomalias de log ativos

Gerenciando suas cotas do serviço CloudWatch Logs

CloudWatch O Logs foi integrado ao Service Quotas, um AWS serviço que permite que você visualize e gerencie suas cotas a partir de um local central. Para obter mais informações, consulte O que é o Service Quotas? no Guia do usuário do Service Quotas.

As cotas de serviço facilitam a pesquisa do valor de suas cotas de serviço do CloudWatch Logs.

Console de gerenciamento da AWS

Para ver CloudWatch as cotas de serviço do Logs usando o console

1. Abra o console do Service Quotas em https://console.aws.amazon.com/servicequotas/.
2. No painel de navegação, escolha Serviços da AWS .
3. Na lista de AWS serviços, pesquise e selecione Amazon CloudWatch Logs.

Na lista Service quotas é possível ver o nome da service quota, o valor aplicado (se estiver disponível), a cota padrão da AWS e se o valor da cota é ajustável.

4. Para visualizar informações adicionais sobre uma Service Quota, como a descrição, escolha o nome da cota.
5. (Opcional) Para solicitar um aumento de cota, selecione a cota que deseja aumentar, selecione Requesting a quota increase (Solicitar aumento de cota), insira ou selecione as informações necessárias e selecione Request (Solicitar).

Para trabalhar mais com service quotas usando o console, consulte o Guia do usuário de Service Quotas. Para solicitar o aumento da quota, consulte Solicitar um aumento de quota no Guia do usuário do Service Quotas.

AWS CLI

Para visualizar CloudWatch as cotas do serviço Logs usando o AWS CLI

Execute o comando a seguir para ver as cotas de CloudWatch registros padrão.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code logs \
    --output table
```

Para trabalhar mais com cotas de serviço usando o AWS CLI, consulte a Referência de Comandos de AWS CLI Cotas de Serviço. Para solicitar um aumento de quotas, consulte o comando `request-service-quota-increase` na Referência de comandos do AWS CLI.

Histórico do documento

A tabela a seguir descreve mudanças importantes em cada versão do Guia do usuário do CloudWatch Logs, a partir de junho de 2018. Para receber notificações sobre atualizações dessa documentação, é possível inscrever-se em um feed RSS.

Alteração	Descrição	Data
Novo comando de consulta de estimativa para CloudWatch Logs Insights	Agora você pode usar o `estimate` comando no CloudWatch Logs Insights para retornar os bytes estimados que a consulta examinaria nos grupos de registros e no intervalo de tempo selecionados, sem executar a consulta. O `estimate` comando não gera cobranças de consulta do CloudWatch Logs Insights. Para obter mais informações, consulte Sintaxe de consulta da linguagem CloudWatch Logs Insights.	22 de julho de 2026
Política de função AWSServiceRoleForLogDelivery vinculada ao serviço atualizada	CloudWatch Registros adicionados `kms:GenerateDataKey` e `kms:Decrypt` permissões à política de função AWSServiceRoleForLogDelivery vinculada ao serviço. Essas permissões permitem a entrega de registros para fluxos de entrega do Firehose que usam criptografia do lado do servidor	15 de maio de 2026

	com chaves gerenciadas pelo cliente (). SSE-CMK Para obter mais informaçõ es, consulte Atualizações de CloudWatch registros em funções vinculadas ao AWS serviço.	
Consultas salvas com parâmetros para CloudWatch Logs Insights	Agora você pode criar modelos de consulta reutilizá veis com parâmetros nomeados no CloudWatc h Logs Insights. Defina um único modelo e transmita valores diferentes em tempo de execução, em vez de manter várias consultas salvas quase idênticas. Para obter mais informações, consulte Usando consultas salvas com parâmetros.	26 de março de 2026
Nova política gerenciada de CloudWatch registros CloudWatchLogsAPIK eyAccess	CloudWatch A Logs lançou uma nova política gerenciad a `CloudWatchLogsAPIK eyAccess` que permite a autenticação da chave da API CloudWatch Logs e a ingestão criptografada de registros. Para obter informaçõ es, consulte Atualizações de CloudWatch registros em políticas AWS gerenciadas.	17 de fevereiro de 2026

CloudWatchLogsRead OnlyAccess Política atualizada	Permissões para ações de administração de observabilidade foram adicionadas para permitir acesso somente CloudWatchLogsRead OnlyAccess para leitura a pipelines de telemetria e integrações de tabelas do S3.	2 de dezembro de 2025
CloudWatchLogsFullAccess Política atualizada	Permissões para ações de administração de observabilidade foram adicionadas para permitir acesso somente CloudWatchLogsFullAccess para leitura a pipelines de telemetria e integrações de tabelas do S3.	2 de dezembro de 2025
Novo recurso de gerenciamento de CloudWatch registros no Logs	CloudWatch consolida o gerenciamento de registros em um único serviço com recursos de governança integrados sem armazenar e manter várias cópias dos mesmos dados em diferentes ferramentas e armazenamentos de dados. Para obter mais informações, consulte Gerenciamento de registros.	2 de dezembro de 2025

Novo recurso de ingestão e normalização de dados no Logs CloudWatch	CloudWatch coleta AWS automaticamente registros vendidos em contas e AWS regiões por meio da integração com AWS organizações e conectores pré-criados para fontes de terceiros. Para obter mais informações, consulte Transformar logs durante a ingestão.	2 de dezembro de 2025
Novo recurso de análise e consulta com base em fontes de dados no CloudWatch Logs Insights	CloudWatch O Insights apresenta facetas para consultas. As facetas são úteis para analisar registros , pois permitem que você filtre e agrupe seus dados de forma interativa sem executar consultas. Uma faceta é um campo em seus registros (como ServiceName ou StatusCode) que permite filtragem, agregação e análise em grupos de registros. Para obter mais informações, consulte Usar facetas para agrupar e explorar registros.	2 de dezembro de 2025

Nova integração de tabelas do S3	A integração do S3 Tables com CloudWatch permite que você acesse dados de log ingeridos CloudWatch usando mecanismos de análise, como Amazon Athena, Amazon Redshift e ferramentas de terceiros que oferecem suporte à conexão com Iceberg-compatible lojas Apache. Para obter mais informações, consulte Acessar registros com a integração de tabelas do S3. https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/s3-tables-integration.html	2 de dezembro de 2025
CloudWatch Os registros adicionaram suporte para Cross-account centralização de registros entre regiões	CloudWatch Os registros adicionaram suporte à centralização de registros Cross-account entre regiões, que permite que CloudWatch os registros coletem dados de registro das contas dos AWS Organizations membros em uma conta central para análise. Para obter mais informações, consulte Centralização de registros Cross-account entre regiões.	17 de setembro de 2025

CloudWatch Exemplo de política atualizada de detecção de anomalias	CloudWatch Os registros atualizaram a política de exemplo do KMS para reduzir seu escopo e melhorar a segurança adicionando condições para `aws:SourceAccount` e. `aws:SourceArn` Para obter mais informações, consulte Criptografar um detector de anomalias e seus resultados com o KMS AWS	14 de julho de 2025
CloudWatch O Logs Insights adiciona suporte aos logs do Amazon VPC Route Server	CloudWatch O Logs Insights adiciona suporte aos logs do Amazon VPC Route Server. Para obter mais informaçõ es, consulte Habilitar o registro em log de serviços da AWS. A nova fonte de log do Amazon VPC Route Server `EVENT_LOGS` está documentada em PutDeliverySource	12 de junho de 2025
CloudWatch O Logs Insights adiciona suporte para AWS PCS registros	CloudWatch O Logs Insights adiciona suporte para AWS PCS registros. Para obter mais informações, consulte Habilitar o registro em log de serviços da AWS. As novas fontes de registro de AWS PCS, `PCS_SCHEDULER_LOGS` e `PCS_JOBCOMP_LOGS` estão documentadas em PutDeliverySource	12 de junho de 2025

CloudWatch O Logs Insights adiciona suporte para AWS Entity Resolution registros	CloudWatch O Logs Insights adiciona suporte para AWS Entity Resolution registros. Para obter mais informações, consulte Habilitar o registro em log de serviços da AWS. A nova fonte de registro de AWS Entity Resolution, `WORKFLOW_LOGS` está documentada em PutDeliverySource	22 de maio de 2025
CloudWatch Registra atualizações gerenciadas de políticas para oferecer suporte a resumos em linguagem natural	As permissões para `cloudwatch:GenerateQueryResultsSummary` foram adicionadas CloudWatchLogsFullAccess e CloudWatchLogsReadOnlyAccess permitem a geração de um resumo em linguagem natural dos resultados da consulta. Para ver o conteúdo das políticas, consulte CloudWatchLogsFullAccess e CloudWatchLogsReadOnlyAccess no Guia de referência de políticas AWS gerenciadas.	20 de maio de 2025

CloudWatch O Logs Insights adiciona suporte para gerar resumos em linguagem natural a partir dos resultados da consulta do CloudWatch Logs Insights	Foi adicionado suporte para resumos em linguagem natural no CloudWatch Logs Insights. Este recurso gera resumos legíveis dos resultados da consulta, disponíveis atualmente na região Leste dos EUA (Norte da Virgínia) . Para obter mais informaçõ es, consulte Gerar resumos em linguagem natural a partir dos resultados da consulta do CloudWatch Logs Insights.	20 de maio de 2025
CloudWatchOpenSear chDashboardAccesse CloudWatchOpenSear chDashboardsFullAccess são novas políticas de IAM	CloudWatch Os registros adicionaram duas novas políticas de IAM CloudWatc hOpenSearchDashboa rdsFullAccess CloudWatc hOpenSearchDashboa rdAccess e. CloudWatc hOpenSearchDashboa rdsFullAccessconcede permissão para criar e gerenciar integrações com o OpenSearch Serviço. CloudWatchOpenSear chDashboardAccessconcede acesso para visualizar painéis de registros vendidos criados nessas integrações.	1.º de dezembro de 2024

CloudWatchLogsFull Accesspolítica atualizada	CloudWatch Os registros adicionaram permissões para Amazon OpenSearch Service e IAM à CloudWatchLogsFull Access política, para permitir a integração do CloudWatch Logs com o OpenSearch Service para alguns recursos.	1.º de dezembro de 2024
CloudWatch O Logs Insights adiciona novos tipos de estrutura à sintaxe da consulta	CloudWatch O Logs Insights adiciona o `unnest` comando e duas funções JSON, que permitem operar cadeias de caracteres JSON como mapas e listas. Para obter mais informações, consulte Tipos de estrutura.	21 de novembro de 2024
CloudWatch Os registros oferecem suporte à transformação de registros durante a ingestão de registros	Você pode criar transformadores de log que podem modificar eventos de logs no momento da ingestão, ajudando a normalizar seus logs em diferentes formatos e a partir de diferentes fontes em formatos consistentes e ricos em contexto. Para obter mais informações, consulte Transformar logs durante a ingestão.	20 de novembro de 2024

CloudWatch O Logs Insights adiciona indexação de campos	CloudWatch O Logs Insights adicionou suporte para indexação de registros em campo. Quando você usa um índice de campo em uma consulta do CloudWatch Logs Insights, a consulta tenta ignorar o processamento de eventos de log que, sabidamente, não incluem o campo indexado. Para obter mais informações, consulte Criar índices de campo para melhorar o desempenho da consulta e reduzir o volume de verificação.	20 de novembro de 2024
CloudWatch O suporte do Logs Insights para geração de consultas em linguagem natural está geralmente disponível	CloudWatch O Logs Insights suporta linguagem natural para gerar e atualizar consultas. Para obter mais informações, consulte Usar linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights.	20 de junho de 2024
CloudWatchLogsReadOnlyAccesspolítica atualizada	CloudWatch Os registros adicionaram a `cloudwatch:GenerateQuery` permissão para CloudWatchLogsReadOnlyAccess que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de um prompt de linguagem natural.	26 de novembro de 2023

CloudWatchLogsFull Accesspolítica atualizada	CloudWatch Os registros adicionaram a `cloudwatch:GenerateQuery` permissão para CloudWatchLogsFullAccess que os usuários com essa política possam gerar uma string de consulta do CloudWatch Logs Insights a partir de um prompt de linguagem natural.	26 de novembro de 2023
CloudWatch Logs adiciona análise de padrão de log	CloudWatch O Logs agora verifica padrões em eventos de registro sempre que você realiza uma consulta do CloudWatch Logs Insights. Para obter mais informações, consulte Análise de padrões.	26 de novembro de 2023
CloudWatch Logs adiciona detecção de anomalias de registro	Você pode criar um detector de anomalias de log para um grupo de logs. O detector de anomalias de log varre os eventos de log ingeridos no grupo de logs e descobre anomalias nos dados dos logs. Para obter mais informações, consulte Log anomaly detection.	26 de novembro de 2023

CloudWatch Logs adiciona recurso de comparação	Agora você pode usar o CloudWatch Logs Insights para comparar as mudanças em seus eventos de registro ao longo do tempo. Para obter mais informações, consulte Compare (diff) with previous time ranges.	26 de novembro de 2023
CloudWatch Logs adiciona uma nova classe de log	CloudWatch O Logs oferece suporte a duas classes de grupos de registros para que você possa ter uma opção econômica para registros que você acessa com pouca frequência, e também tem uma opção completa para registros que exigem monitoramento em tempo real ou outros recursos. Para obter mais informações, consulte Log classes.	26 de novembro de 2023
CloudWatch O Logs Insights oferece suporte à geração de consultas em linguagem natural	CloudWatch O Logs Insights suporta linguagem natural para gerar e atualizar consultas. Para obter mais informações, consulte Usar linguagem natural para gerar e atualizar consultas do CloudWatch Logs Insights.	26 de novembro de 2023

CloudWatch O Logs adiciona suporte à sintaxe de padrão de filtro de expressão regular para Live Tail	Agora é possível personalizar ainda mais suas operações de pesquisa e correspon dência para atender às suas necessidades com expressõe s regulares flexíveis com padrões de filtros do Live Tail. Para obter mais informações, consulte Sintaxe do padrão de filtro no Guia do usuário do Amazon CloudWatch Logs.	13 de novembro de 2023
CloudWatch O Logs adiciona suporte à sintaxe do padrão de filtro de expressão regular para filtros métricos, filtros de assinatura e eventos de registro de filtro	Agora é possível personalizar ainda mais suas operações de pesquisa e correspon dência para atender às suas necessidades com expressõe s regulares flexíveis em padrões de filtros. Para obter mais informações, consulte Sintaxe do padrão de filtro no Guia do usuário do Amazon CloudWatch Logs.	5 de setembro de 2023
CloudWatch O Logs Insights adiciona um comando de padrão	Agora você pode usar o padrão em suas consultas do CloudWatch Logs Insights para agrupar automaticamente seus dados de registro em padrões. Um padrão é uma estrutura de texto compartil hada que se repete entre seus campos de log. Para obter mais informações, consulte o padrão no Guia do usuário do Amazon CloudWatch Logs.	17 de julho de 2023

CloudWatch O Logs Insights adiciona um comando de dedup	Agora você pode usar dedup em suas consultas do CloudWatch Logs Insights para remover resultados duplicados com base em valores específicos nos campos que você especific ar. Para obter mais informaçõ es, consulte dedup no Guia do usuário do Amazon CloudWatch Logs.	20 de junho de 2023
Account-level políticas de proteção de dados	Agora, é possível definir políticas de proteção de dados no nível da conta. Essas políticas podem auditar e mascarar informações confidenciais em eventos de log em todos os grupos de logs da conta. Para obter mais informações, consulte Ajude a proteger dados de log confidenciais com mascarame nto no Guia do usuário do Amazon CloudWatch Logs.	8 de junho de 2023

Adição do recurso Live Tail	CloudWatch Os registros adicionaram a capacidad e Live Tail, para que você possa escanear os registros à medida que são ingeridos para ajudar na solução de problemas. Há a opção de filtrar o fluxo exibido de eventos de log com base em termos especificados e também de realçar eventos de log que tenham termos especificados. Para obter mais informações, consulte Usar o Live Tail para visualiza r registros quase em tempo real.	6 de junho de 2023
CloudWatchLogsRead OnlyAccesspolítica atualizada	CloudWatch Os registros adicionaram permissõe s CloudWatchLogsRead OnlyAccess a. As `logs:StopLiveTail` permissões `logs:Star tLiveTail` e foram adicionadas para que os usuários com essa política possam usar o console para iniciar e interromper as sessões ao vivo do CloudWatc h Logs. Para obter mais informações, consulte Usar o Live Tail para visualizar registros quase em tempo real.	6 de junho de 2023

CloudWatch Lançamento do Logs Insights	Você pode usar o CloudWatch Logs Insights para pesquisar e analisar interativamente seus dados de registro. Para obter mais informações, consulte Analisar dados de log com o CloudWatch Logs Insights no Guia do usuário do Amazon CloudWatch Logs.	27 de novembro de 2018
Suporte para endpoints da Amazon VPC	Agora você pode estabelecer uma conexão privada entre sua VPC e seus CloudWatch registros. Para obter mais informações, consulte Usando CloudWatch registros com VPC Endpoints de interface no Guia do usuário do Amazon CloudWatch Logs.	28 de junho de 2018

A tabela a seguir descreve as mudanças importantes no Guia do usuário do Amazon CloudWatch Logs.

Alteração	Description	Data de lançamento
Endpoints da VPC de interface	Em algumas regiões, você pode usar uma interface VPC endpoint para impedir que o tráfego entre sua Amazon VPC e CloudWatch os Logs saia da rede Amazon. Para obter mais informações, consulte Usando CloudWatch registros com endpoints VPC de interface.	7 de março de 2018
Logs de consulta de DNS do Route 53	Você pode usar CloudWatch Logs para armazenar registros sobre as consultas de DNS recebidas pelo Route 53. Para mais informações, consulte O que é Amazon CloudWatch Logs? ou Registrar	7 de setembro de 2017

Alteração	Description	Data de lançamento
	consultas de DNS no Guia do desenvolvedor do Amazon Route 53.	
Etiquetar grupos de logs	Você pode usar marcas para categorizar seus grupos de logs. Para obter mais informações, consulte Marque grupos de registros no Amazon CloudWatch Logs.	13 de dezembro de 2016
Melhorias no console	Você pode navegar de gráficos de métricas para os grupos de logs associados. Para obter mais informações, consulte Passar de métricas para logs.	7 de novembro de 2016
Melhorias no uso do console	Aprimorada a experiência para facilitar a pesquisa, a filtragem e a solução de problemas. Por exemplo, agora você pode filtrar seus dados de log para um intervalo de data e hora. Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch Logs.	29 de agosto de 2016
AWS CloudTrail Suporte adicionado para Amazon CloudWatch Logs e novas métricas CloudWatch de registros	Foi adicionado AWS CloudTrail suporte para CloudWatch registros. Para obter mais informações, consulte Operações da API CloudWatch Logging Logs e do console em AWS CloudTrail.	10 de março de 2016
Suporte adicionado para exportação de CloudWatch registros para o Amazon S3	Foi adicionado suporte à exportação de dados de CloudWatch registros para o Amazon S3. Para obter mais informações, consulte Exportar dados de log para o Amazon S3.	7 de dezembro de 2015

Alteração	Description	Data de lançamento
Suporte adicionado para eventos AWS CloudTrail registrados no Amazon CloudWatch Logs	Você pode criar alarmes CloudWatch e receber notificações de uma atividade específica da API conforme capturada CloudTrail e usar a notificação para solucionar problemas.	10 de novembro de 2014
Suporte adicionado para Amazon CloudWatch Logs	Você pode usar o Amazon CloudWatch Logs para monitorar, armazenar e acessar seu sistema, aplicativo e arquivos de log personalizados a partir de instâncias do Amazon Elastic Compute Cloud (Amazon EC2) ou de outras fontes. Em seguida, você pode recuperar os dados de log associados do CloudWatch Logs usando o CloudWatch console Amazon, os comandos CloudWatch Logs no ou o AWS CLI CloudWatch Logs SDK. Para obter mais informações, consulte O que é Amazon CloudWatch Logs?.	10 de julho de 2014

AWS Glossário

Para obter a AWS terminologia mais recente, consulte o AWS glossário na Glossário da AWS Referência.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.