aws

사용자 가이드

아마존 CloudWatch 로그

아마존 CloudWatch 로그: 사용자 가이드

Table of Contents

아마존 CloudWatch 로그란 무엇입니까?

Amazon CloudWatch Logs를 사용하여 Amazon Elastic Compute Cloud (Amazon EC2) 인스턴스, Route 53 및 기타 소스에서 로그 파일을 모니터링 AWS CloudTrail, 저장 및 액세스할 수 있습니다.

CloudWatch 로그를 사용하면 사용하는 모든 시스템, 애플리케이션 및 AWS 서비스의 로그를 확장성이 뛰어난 단일 서비스로 중앙 집중화할 수 있습니다. 그런 다음 쉽게 확인하고, 특정 오류 코드나 패턴을 검색하고, 특정 필드를 기준으로 필터링하거나, 향후 분석을 위해 안전하게 보관할 수 있습니다. CloudWatch 로그를 사용하면 소스에 상관없이 모든 로그를 시간별로 정렬된 일관된 단일 이벤트 흐름으로 볼 수 있습니다.

CloudWatch 또한 로그는 강력한 쿼리 언어를 사용하여 로그를 쿼리하고, 로그의 민감한 데이터를 감사 및 마스킹하고, 필터 또는 내장된 로그 형식을 사용하여 로그에서 지표를 생성하는 기능을 지원합니다.

CloudWatch 로그는 두 개의 로그 클래스를 지원합니다. Logs Standard CloudWatch 로그 클래스의 로그 그룹은 모든 CloudWatch 로그 기능을 지원합니다. CloudWatch Logs Inrequent Access 로그 클래스의 로그 그룹은 수집 요금이 더 적게 부과되며 Standard 클래스 기능의 일부를 지원합니다. 자세한 정보는 로그 클래스을 참조하세요.

특성

- 유연성을 위한 두 개의 로그 클래스 - CloudWatch 로그는 두 개의 로그 클래스를 제공하므로 자주 액세스하지 않는 로그에 대해 비용 효율적인 옵션을 사용할 수 있습니다. 실시간 모니터링 또는 기타 기능이 필요한 로그를 위한 모든 기능을 갖춘 옵션도 있습니다. 자세한 정보는 로그 클래스을 참조하세요.
- 로그 데이터 쿼리 — CloudWatch Logs Insights를 사용하여 대화형 방식으로 로그 데이터를 검색하고 분석할 수 있습니다. 쿼리를 수행하여 운영 문제에 보다 효율적이고 효과적으로 대응할 수 있습니다. CloudWatch Logs Insights에는 몇 가지 간단하지만 강력한 명령을 포함하는 특수 목적의 쿼리 언어가 포함되어 있습니다. 시작하는 데 도움이 되는 샘플 쿼리, 명령 설명, 쿼리 자동 완성 및 로그 필드 검색이 제공됩니다. 여러 유형의 AWS 서비스 로그에 대한 샘플 쿼리가 포함되어 있습니다. 시작하려면 로그 인사이트를 통한 CloudWatch 로그 데이터 분석 섹션을 참조하세요.
- Live Tail을 사용한 탐지 및 디버깅 - Live Tail을 사용하면 모은 새 로그 이벤트의 스트리밍 목록을 확인하여 인시던트 문제를 신속하게 해결할 수 있습니다. 모은 로그를 거의 실시간으로 보고 필터링하고 강조 표시할 수 있으므로 문제를 빠르게 탐지하고 해결할 수 있습니다. 지정한 항을 기반으로 로그를 필터링하고 원하는 항목을 빠르게 찾을 수 있도록 지정된 항이 포함된 로그를 강조 표시할 수도 있습니다. 자세한 정보는 Live Tail을 사용하여 거의 실시간으로 로그 보기을 참조하세요.

- Amazon EC2 인스턴스의 로그 모니터링 — Logs를 사용하면 CloudWatch 로그 데이터를 사용하여 애플리케이션과 시스템을 모니터링할 수 있습니다. 예를 들어, Logs는 애플리케이션 CloudWatch 로그에서 발생하는 오류 수를 추적하고 오류율이 지정한 임계값을 초과할 때마다 알림을 보낼 수 있습니다. CloudWatch 로그는 모니터링에 로그 데이터를 사용하므로 코드를 변경할 필요가 없습니다. 예를 들어 애플리케이션 로그에서 특정 리터럴 용어 (예: "NullReferenceException“) 를 모니터링하거나 로그 데이터의 특정 위치에서 리터럴 용어가 발생하는 횟수 (예: Apache 액세스 로그의 “404" 상태 코드) 를 계산할 수 있습니다. 검색하려는 용어를 찾으면 CloudWatch Logs는 지정한 지표에 데이터를 보고합니다. CloudWatch 로그 데이터는 전송 시는 물론 저장 시에도 암호화됩니다. 시작하려면 CloudWatch 로그 시작하기섹션을 참조하세요.
- AWS CloudTrail 기록된 이벤트 모니터링 - 에서 캡처한 특정 API 활동에 대한 알림을 CloudWatch 생성하고 수신한 CloudTrail 다음 알림을 사용하여 문제 해결을 수행할 수 있습니다. 시작하려면 AWS CloudTrail 사용 설명서의 CloudWatch 로그에 CloudTrail 이벤트 전송을 참조하십시오.
- 민감한 데이터 감사 및 마스킹 - 로그에 민감한 데이터가 있는 경우 데이터 보호 정책을 통해 이를 보호할 수 있습니다. 이러한 정책을 통해 민감한 데이터를 감사하고 마스킹 처리할 수 있습니다. 데이터 보호를 활성화하면 선택한 데이터 식별자와 일치하는 중요한 데이터가 기본적으로 마스킹 처리됩니다. 자세한 정보는 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원을 참조하세요.
- 로그 보존 - 기본적으로 로그는 무기한으로 보존되며 만료되지 않습니다. 로그 그룹별로 보존 정책을 조정할 수 있고 무기한 보존 유지 또는 10년 및 하루 중 보존 기간을 선택합니다.
- 로그 데이터 보관 - CloudWatch 로그를 사용하여 로그 데이터를 내구성이 뛰어난 스토리지에 저장할 수 있습니다. CloudWatch Logs 에이전트를 사용하면 회전된 로그 데이터와 회전되지 않은 로그 데이터를 호스트에서 로그 서비스로 쉽게 빠르게 보낼 수 있습니다. 그런 다음 필요할 때 원시 로그 데이터에 액세스할 수 있습니다.
- Route 53 DNS 쿼리 기록 - CloudWatch 로그를 사용하여 Route 53이 수신하는 DNS 쿼리에 대한 정보를 기록할 수 있습니다. 자세한 내용은 Amazon Route 53 개발자 안내서의 DNS 쿼리 로깅을 참조하세요.

관련 AWS 서비스

CloudWatch 로그와 함께 사용되는 서비스는 다음과 같습니다.

- AWS CloudTrail AWS Management Console, AWS Command Line Interface (AWS CLI) 및 기타 서비스에서 이루어진 호출을 포함하여 계정의 CloudWatch Logs API에 대한 호출을 모니터링할 수 있는 웹 서비스입니다. CloudTrail 로깅이 켜지면 계정에서 API 호출을 CloudTrail 캡처하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 각 로그 파일에는 하나 이상의 레코드가 포함될 수 있으며, 요청을 충족하기 위해 수행해야 하는 작업의 수에 따라 그 수가 결정됩니다. 에 대한

AWS CloudTrail자세한 내용은 AWS CloudTrail무엇입니까를 참조하십시오. AWS CloudTrail 사용 설명서에서. CloudTrail 로그 파일에 CloudWatch 쓰는 데이터 유형의 예는 을 참조하십시오로깅 CloudWatch 로그 API 및 콘솔 작업 AWS CloudTrail.

- AWS Identity and Access Management (IAM) 은 사용자의 AWS 리소스 액세스를 안전하게 제어하는 데 도움이 되는 웹 서비스입니다. IAM을 사용하여 AWS 리소스를 사용할 수 있는 사람을 제어(인증)하고 이들이 사용할 수 있는 리소스 및 그 사용 방법을 제어(권한 부여)합니다. 자세한 내용은 IAM 사용 설명서의 IAM이란 무엇입니까?를 참조하세요.
- Amazon Kinesis Data Streams는 신속하고 지속적인 데이터 인테이크 및 집계를 위해 사용할 수 있는 웹 서비스입니다. 사용되는 데이터 유형으로는 IT 인프라 로그 데이터, 애플리케이션 로그, 소셜 미디어, 시장 데이터 피드, 웹 클릭스트림 데이터 등이 있습니다. 데이터 인테이크 및 처리에 대한 응답이 실시간으로 이루어지기 때문에 일반적으로 간소화된 방식으로 처리가 됩니다. 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 Amazon Kinesis Data Streams이란 무엇입니까?를 참조하세요.
- AWS Lambda는 새 정보에 신속하게 응답하는 애플리케이션을 구축하는 데 사용할 수 있는 웹 서비스입니다. Lambda 함수 형태로 애플리케이션 코드를 업로드하면 는 고가용성 컴퓨팅 인프라에서 코드를 실행하고 서버 및 운영 체제 유지 관리, 용량 프로비저닝 및 자동 조정, 코드 및 보안 패치 배포, 코드 모니터링 및 로깅 등 모든 컴퓨팅 리소스 관리를 수행합니다. Lambda가 지원하는 언어 중 하나로 코드를 공급하기만 하면 됩니다. 자세한 내용은 What is AWS Lambda? 를 참조하십시오. AWS Lambda 개발자 안내서에서.

요금

가입하면 프리 티어를 사용하여 무료로 CloudWatch 로그를 시작할 수 있습니다. AWSAWS

CloudWatch 로그를 사용하여 다른 서비스에서 저장하는 로그에는 표준 요금이 적용됩니다 (예: Amazon VPC 흐름 로그 및 Lambda 로그).

요금에 대한 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

CloudWatch 로그 비용 및 사용량을 분석하는 방법에 대한 자세한 내용과 비용 절감 방법에 대한 모범 사례는 CloudWatch 청구 및 비용을 참조하십시오. CloudWatch

아마존 CloudWatch 로그 개념

CloudWatch 로그를 이해하고 사용하는 데 있어 핵심이 되는 용어와 개념은 아래에 설명되어 있습니다.

로그 클래스

CloudWatch 로그는 두 가지 로그 그룹 클래스를 제공합니다. 표준 로그 클래스는 실시간 모니터링이 필요한 로그 또는 자주 액세스하는 로그를 위한 모든 기능을 갖춘 옵션입니다. Inrequent Access 로그 클래스는 자주 액세스하지 않는 로그를 위한 저렴한 옵션입니다. 표준 로그 클래스 기능의 일부를 지원합니다.

로그 이벤트

로그 이벤트는 모니터링 중인 애플리케이션 또는 리소스에 기록된 일부 활동에 대한 레코드입니다. CloudWatch Logs가 인식하는 로그 이벤트 레코드에는 이벤트 발생 시점의 타임스탬프와 원시 이벤트 메시지라는 두 가지 속성이 있습니다. 각 메시지는 UTF-8로 인코딩되어야 합니다.

로그 스트림

로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다. 보다 구체적으로 말하자면, 로그 스트림은 모니터링 중인 애플리케이션 인스턴스나 리소스에서 나온 이벤트의 시퀀스를 표시하는 데 주로 사용됩니다. 예를 들어 로그 스트림은 특정 호스트의 Apache 액세스 로그에 연결될 수 있습니다. 로그 스트림이 더 이상 필요하지 않으면 aws logs delete-log-stream 명령을 사용하여 삭제할 수 있습니다.

로그 그룹

로그 그룹은 동일한 보존 기간, 모니터링 및 액세스 제어 설정을 공유하는 로그 스트림 그룹을 정의합니다. 각 로그 스트림은 하나의 로그 그룹에 속해야 합니다. 예를 들어, 각 호스트의 Apache 액세스 로그에 대해 별도의 로그 스트림이 있으면 로그 스트림을 `MyWebsite.com/Apache/access_log`라는 하나의 로그 그룹으로 묶을 수 있습니다.

하나의 로그 그룹이 가질 수 있는 로그 스트림의 수는 제한이 없습니다.

지표 필터

메트릭 필터를 사용하여 수집된 이벤트에서 지표 관찰을 추출하고 이를 지표의 데이터 포인트로 변환할 수 있습니다CloudWatch . 지표 필터는 로그 그룹에 할당이 되고, 로그 그룹에 할당된 모든 필터는 로그 스트림에 적용됩니다.

보존 기간 설정

보존 설정을 사용하여 로그 이벤트가 로그에 CloudWatch 보관되는 기간을 지정할 수 있습니다. 기간이 만료된 로그 이벤트는 자동으로 삭제됩니다. 지표 필터와 마찬가지로 보존 기간 설정 역시 로그 그룹에 할당이 되며, 로그 그룹에 할당된 보존 기간은 로그 스트림에 적용됩니다.

Amazon CloudWatch Logs 과금 정보 및 비용

CloudWatch Logs 및 CloudWatch의 비용 및 사용량을 분석하는 방법에 대한 자세한 정보와 비용을 절감하는 방법에 대한 모범 사례는 CloudWatch 결제 및 비용을 참조하세요.

요금에 대한 자세한 정보는 Amazon CloudWatch 요금을 참조하세요.

AWS 가입 시 무상으로 CloudWatch Logs를 시작할 수 있는 AWS 프리 티어를 제공합니다.

표준 요금은 CloudWatch Logs를 사용하여 다른 서비스가 저장한 로그(예: Amazon VPC 흐름 로그 및 Lambda 로그)에 적용됩니다.

로그 클래스

CloudWatch 로그는 두 가지 로그 그룹 클래스를 제공합니다.

- CloudWatch Logs Standard 로그 클래스는 실시간 모니터링이 필요한 로그 또는 자주 액세스하는 로그를 위한 모든 기능을 갖춘 옵션입니다.
- CloudWatch Logs Inrequent Access 로그 클래스는 로그를 비용 효율적으로 통합하는 데 사용할 수 있는 새로운 로그 클래스입니다. 이 로그 클래스는 GB당 더 저렴한 수집 CloudWatch 가격으로 관리형 수집, 스토리지, 계정 간 로그 분석, 암호화를 비롯한 일부 로그 기능을 제공합니다. Inrequent Access 로그 클래스는 자주 액세스하지 않는 로그에 대한 임시 쿼리 및 포렌식 분석에 적합합니다. after-the-fact

> **Note**
>
> 요금의 경우 표준 액세스 로그 클래스와 비정기 액세스 로그 클래스는 수집 비용만 다릅니다. 스토리지 요금과 CloudWatch Logs Insights 요금은 각 로그 클래스에서 동일합니다.

CloudWatch 로그 요금에 대한 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

> **Important**
>
> 로그 그룹을 생성한 후에는 해당 로그 클래스를 변경할 수 없습니다.

지원되는 기능

다음 표에는 각 로그 클래스의 기능이 나열되어 있습니다.

	표준	간헐적 액세스	
완전 관리형 로그 수집 및 저장	✓	✓	
크로스 계정 기능	✓	✓	
를 사용한 암호화 AWS KMS	✓	✓	

	표준	간헐적 액세스	
CloudWatch 로그 인사이트 쿼리 명령	✓	✓ (대부분의 명령— 로그 클래스에서 지원되는 명령 참조)	
CloudWatch 로그 인사이트 검색 필드	✓		
자연어 쿼리 지원	✓		
CloudWatch 로그 이상 탐지	✓		
이전 시간 범위와 비교	✓		
구독 필터	✓		
Amazon S3로 내보내기	✓		
GetLogEvents및 FilterLogEventsAPI 작업	✓	지원하지 않음. CloudWatch 로그 인사이트를 사용하면 Inrequent Access 로그 클래스의 로그 그룹에 저장된 로그 이벤트를 볼 수 있습니다.	
지표 필터	✓		
컨테이너 인사이트 로그 수집	✓		
Lambda 인사이트 로그 수집	✓		

	표준	간헐적 액세스	
마스킹을 통한 민감한 데이터 보호	✓		
임베디드 메트릭 형식	✓		

CloudWatch 로그 시작하기

Amazon EC2 인스턴스 및 온프레미스 서버의 로그를 CloudWatch Logs로 수집하려면 통합 에이전트를 사용하십시오. CloudWatch 에이전트 하나로 로그와 고급 지표를 모두 수집할 수 있습니다. Windows Server를 실행하는 서버를 포함하여 운영 체제 전반에 걸쳐 지원을 제공합니다. 또한 이 에이전트는 우수한 성능을 제공합니다.

통합 CloudWatch 에이전트를 사용하여 CloudWatch 지표를 수집하는 경우 게스트 내 가시성을 위해 추가 시스템 지표를 수집할 수 있습니다. 또한 `StatsD` 또는 `collectd`를 사용하는 사용자 지정 지표 수집을 지원합니다.

자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트 설치를 참조하십시오.

Linux를 실행하는 서버의 CloudWatch 로그 수집만 지원하는 이전 Logs 에이전트는 더 이상 사용되지 않으며 더 이상 지원되지 않습니다. 이전 CloudWatch Logs 에이전트에서 통합 에이전트로 마이그레이션하는 방법에 대한 자세한 내용은 마법사를 사용하여 CloudWatch 에이전트 구성 파일 만들기를 참조하십시오.

내용

필수 조건

Amazon CloudWatch Logs를 사용하려면 AWS 계정이 필요합니다. AWS 계정을 사용하면 서비스 (예: Amazon EC2) 를 사용하여 웹 기반 인터페이스인 CloudWatch 콘솔에서 볼 수 있는 로그를 생성할 수 있습니다. 또한 AWS Command Line Interface ()AWS CLI를 설치하고 구성할 수 있습니다.

가입하여 AWS 계정

계정이 없는 경우 다음 단계를 완료하여 계정을 만드세요. AWS 계정

가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 여세요.

2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   에 AWS 계정가입하면 AWS 계정 루트 사용자a가 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 가입 절차가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 가서 내 계정(My Account)을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

등록한 AWS 계정후에는 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 보호하고 AWS IAM Identity Center활성화하고 생성하십시오 AWS 계정 루트 사용자.

보안을 유지하세요. AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 AWS Management Console소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자 (콘솔) 에 대한 가상 MFA 디바이스 활성화를 참조하십시오.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 ID 소스로 사용하는 방법에 대한 자습서는 사용 설명서의 기본값으로 IAM Identity Center 디렉터리사용자 액세스 구성을 참조하십시오. IAM Identity Center 디렉터리 AWS IAM Identity Center

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM IDentity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 사용 설명서의 AWS 액세스 포털에 로그인을 참조하십시오.AWS 로그인

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

Command Line Interface 설정

를 AWS CLI 사용하여 CloudWatch 로그 작업을 수행할 수 있습니다.

설치 및 구성 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS 명령줄 인터페이스를 사용한 설정을 참조하십시오. AWS CLI

통합 CloudWatch 에이전트를 사용하여 Logs를 시작하세요 CloudWatch .

통합 CloudWatch 에이전트를 사용하여 CloudWatch 로그를 시작하는 방법에 대한 자세한 내용은 Amazon User Guide의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집을 참조하십시오. CloudWatch 이 섹션에 나열된 단계를 완료하여 에이전트를 설치, 구성 및 시작합니다. 에이전트를 사용하여 CloudWatch 메트릭을 수집하지 않는 경우에는 지표와 관련된 모든 섹션을 무시해도 됩니다.

현재 이전 CloudWatch Logs 에이전트를 사용하고 있는데 새 통합 에이전트를 사용하도록 마이그레이션하려는 경우 새 에이전트 패키지에 포함된 마법사를 사용하는 것이 좋습니다. 이 마법사는 현재 CloudWatch 로그 에이전트 구성 파일을 읽고 동일한 로그를 수집하도록 CloudWatch 에이전트를 설정할 수 있습니다. 마법사에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 마법사를 사용하여 CloudWatch 에이전트 구성 파일 생성을 참조하십시오.

이전 CloudWatch 에이전트를 사용하여 CloudWatch 로그를 시작하십시오.

> **Important**
>
> CloudWatch EC2 인스턴스 및 온프레미스 서버에서 로그와 지표를 모두 수집할 수 있는 통합 CloudWatch 에이전트가 포함되어 있습니다. 이전 로그 전용 에이전트는 사용 중단되어 더 이상 지원되지 않습니다.
> 이전의 로그 전용 에이전트에서 통합 에이전트로 마이그레이션하는 방법에 대한 자세한 내용은 마법사를 사용하여 CloudWatch 에이전트 구성 파일 생성을 참조하십시오.
> 이 섹션의 나머지 부분에서는 기존 Logs 에이전트를 아직 사용 중인 고객을 위해 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

CloudWatch 로그 에이전트를 사용하면 Linux 또는 Windows Server를 실행하는 Amazon EC2 인스턴스의 로그 데이터와 로깅된 이벤트를 게시할 수 있습니다. AWS CloudTrail대신 CloudWatch 통합 에이전트를 사용하여 로그 데이터를 게시하는 것이 좋습니다. 새 에이전트에 대한 자세한 내용은 Amazon 사용 설명서의 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집을 참조하십시오. CloudWatch CloudWatch

내용

CloudWatch 로그 에이전트 사전 요구 사항

CloudWatch 로그 에이전트에는 Python 버전 2.7, 3.0 또는 3.3과 다음 Linux 버전 중 하나가 필요합니다.

- Amazon Linux 버전 2014.03.02 이상 Amazon Linux 2는 지원되지 않습니다.
- Ubuntu Server 버전 12.04, 14.04 또는 16.04
- CentOS 버전 6, 6.3, 6.4, 6.5 또는 7.0
- Red Hat Enterprise Linux(RHEL) 버전 6.5 또는 7.0
- Debian 8.0

빠른 시작: 실행 중인 EC2 Linux CloudWatch 인스턴스에 Logs 에이전트를 설치하고 구성합니다.

Important

이전 로그 에이전트는 더 이상 사용되지 않습니다. CloudWatch EC2 인스턴스와 온프레미스 서버에서 로그와 지표를 모두 수집할 수 있는 통합 에이전트가 포함되어 있습니다. 자세한 정보는 CloudWatch 로그 시작하기을 참조하세요.
이전 CloudWatch Logs 에이전트에서 통합 에이전트로 마이그레이션하는 방법에 대한 자세한 내용은 마법사를 사용하여 CloudWatch 에이전트 구성 파일 만들기를 참조하십시오.
이전 로그 에이전트는 Python 버전 2.6부터 3.5까지만 지원합니다. 또한 이전 CloudWatch Logs 에이전트는 IMDSv2 (인스턴스 메타데이터 서비스 버전 2) 를 지원하지 않습니다. 서버에서 IMDSv2를 사용하는 경우 이전 Logs 에이전트 대신 최신 통합 에이전트를 사용해야 합니다. CloudWatch
이 섹션의 나머지 부분에서는 기존 Logs 에이전트를 아직 사용 중인 고객을 위해 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

Tip

CloudWatch EC2 인스턴스와 온프레미스 서버에서 로그와 지표를 모두 수집할 수 있는 새로운 통합 에이전트가 포함되어 있습니다. 이전 CloudWatch Logs 에이전트를 아직 사용하지 않는 경우 최신 통합 에이전트를 사용하는 것이 좋습니다. CloudWatch 자세한 정보는 CloudWatch 로그 시작하기을 참조하세요.

또한 이전 에이전트는 Instance Metadata Service 버전 2(IMDSv2)를 지원하지 않습니다. 서버에서 IMDSv2를 사용하는 경우 이전 Logs 에이전트 대신 최신 통합 에이전트를 사용해야 합니다. CloudWatch
이 섹션의 나머지 부분에서는 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

실행 중인 EC2 Linux 인스턴스에서 이전 CloudWatch Logs 에이전트를 구성합니다.

기존 EC2 인스턴스에서 CloudWatch Logs 에이전트 설치 프로그램을 사용하여 Logs 에이전트를 설치하고 구성할 수 있습니다. CloudWatch 설치가 완료되면 로그가 자동으로 인스턴스에서 에이전트 설치 도중 생성한 로그 스트림으로 흐릅니다. 에이전트는 인스턴트가 시작되었고 비활성화를 할 때까지 실행이 유지되는지 확인합니다.

에이전트를 사용하는 것 외에도 AWS CLI, CloudWatch Logs SDK 또는 Logs API를 사용하여 로그 데이터를 게시할 수 있습니다. CloudWatch 명령줄이나 스크립트를 통해 데이터를 게시하는 데 AWS CLI 가장 적합합니다. CloudWatch Logs SDK는 애플리케이션에서 직접 로그 데이터를 게시하거나 자체 로그 게시 애플리케이션을 구축하는 데 가장 적합합니다.

1단계: 로그에 대한 IAM 역할 또는 사용자 구성 CloudWatch

CloudWatch Logs 에이전트는 IAM 역할 및 사용자를 지원합니다. 인스턴스가 이미 IAM 역할에 연결되어 있으면 아래의 IAM 정책이 포함되어 있는지 확인합니다. 인스턴스에 IAM 역할이 할당되어 있지 않으면 다음 단계에서 IAM 자격 증명을 사용하거나 해당 인스턴스에 IAM 역할을 할당할 수 있습니다. 자세한 내용은 IAM 역할을 인스턴스에 연결을 참조하세요.

로그에 대한 IAM 역할 또는 사용자 구성하기 CloudWatch

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 이름을 선택하여 역할을 선택합니다(이름 옆의 확인란은 선택하지 않음).
4. 정책 연결에서 정책 생성을 선택합니다.

   새 브라우저 탭 또는 창이 열립니다.
5. JSON 탭을 선택하고 다음 JSON 정책 문서를 입력합니다.

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

```
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
    ],
      "Resource": [
        "*"
    ]
  }
 ]
}
```

6. 작업이 완료되면 정책 검토를 선택합니다. 정책 검사기가 모든 구문 오류를 보고합니다.
7. 정책 검토 페이지에서 생성하는 정책에 대한 이름과 설명(선택 사항)을 입력합니다. 정책 요약을 검토하여 정책이 부여한 권한을 확인합니다. 그런 다음 정책 생성을 선택하여 작업을 저장합니다.
8. 브라우저 탭 또는 창을 닫고, 역할의 권한 추가 페이지로 돌아옵니다. 새로 고침을 선택한 후 새 정책을 선택하여 역할에 연결합니다.
9. 정책 연결을 선택하세요.

2단계: 기존 Amazon EC2 인스턴스에 CloudWatch 로그를 설치하고 구성합니다.

CloudWatch Logs 에이전트 설치 프로세스는 Amazon EC2 인스턴스가 Amazon Linux, Ubuntu, CentOS 또는 Red Hat을 실행하는지 여부에 따라 달라집니다. 인스턴스에서 Linux 버전에 적합한 단계를 사용합니다.

기존 Amazon Linux 인스턴스에 CloudWatch 로그를 설치하고 구성하려면

Amazon Linux AMI 2014.09부터 CloudWatch 로그 에이전트는 awslogs 패키지와 함께 RPM 설치로 사용할 수 있습니다. 이전 버전의 Amazon Linux는 `sudo yum update -y` 명령을 통해 인스턴스를 업데이트하여 awslogs 패키지에 액세스할 수 있습니다. CloudWatch Logs 설치 프로그램을 사용하는 대신 awslogs 패키지를 RPM으로 설치하면 Logs 에이전트를 수동으로 다시 설치할 필요 없이 인스턴스가 정기적인 패키지 업데이트 및 패치를 받을 수 있습니다. AWS CloudWatch

> **Warning**
>
> 이전에 Python 스크립트를 사용하여 에이전트를 설치한 경우 RPM 설치 방법을 사용하여 CloudWatch Logs 에이전트를 업데이트하지 마십시오. 이렇게 하면 구성 문제가 발생하여 CloudWatch CloudWatch Logs 에이전트에서 로그를 전송하지 못할 수 있습니다.

1. Amazon Linux 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하십시오.

   연결 문제에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 연결 문제 해결을 참조하십시오.

2. Amazon Linux 인스턴스를 업데이트하여 패키지 리포지토리에서 최신 변경 사항을 찾아냅니다.

   ```
   sudo yum update -y
   ```

3. `awslogs` 패키지를 설치합니다. 이는 Amazon Linux 인스턴스에서 awslogs를 설치하는 데 있어 권장되는 방법입니다.

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

4. `/etc/awslogs/awslogs.conf` 파일을 편집하여 추적할 로그를 구성합니다. 이 파일을 편집하는 방법에 대한 자세한 내용은 CloudWatch 로그 에이전트 참조 섹션을 참조하세요.

5. 기본적으로 `/etc/awslogs/awscli.conf`는 us-east-1 리전을 가리킵니다. 다른 리전으로 로그를 푸시하려면 `awscli.conf` 파일을 편집하고 해당 리전을 지정합니다.

6. `awslogs` 서비스를 시작합니다.

   ```
   sudo service awslogs start
   ```

   Amazon Linux 2를 실행 중인 경우 다음 명령과 함께 `awslogs` 서비스를 시작합니다.

   ```
   sudo systemctl start awslogsd
   ```

7. (선택 사항) 서비스 시작 시 기록된 오류에 대해서는 `/var/log/awslogs.log` 파일을 확인하세요.

8. (선택 사항) 다음 명령을 실행하여 시스템 부팅 때마다 `awslogs` 서비스를 시작합니다.

```
sudo chkconfig awslogs on
```

Amazon Linux 2를 실행 중인 경우 다음 명령으로 각 시스템 부팅에서 서비스를 시작합니다.

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

9. 에이전트가 잠시 실행되면 CloudWatch 콘솔에서 새로 생성된 로그 그룹과 로그 스트림을 확인할 수 있습니다.

   자세한 정보는 Logs로 전송된 로그 데이터 보기 CloudWatch 을 참조하세요.

기존 우분투 서버, CentOS 또는 Red Hat 인스턴스에 CloudWatch 로그를 설치하고 구성하려면

Ubuntu Server, CentOS 또는 Red Hat을 실행하는 AMI를 사용하는 경우 다음 절차를 사용하여 인스턴스에 CloudWatch 로그 에이전트를 수동으로 설치합니다.

1. EC2 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하십시오.

   연결 문제에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 연결 문제 해결을 참조하십시오.
2. 두 옵션 중 하나를 사용하여 CloudWatch 로그 에이전트 설치 프로그램을 실행합니다. 인터넷에서 직접 실행하거나 파일을 다운로드하여 독립적으로 실행할 수 있습니다.

   > **Note**
   >
   > CentOS 6.x, Red Hat 6.x, 또는 Ubuntu 12.04를 사용 중인 경우 독립 실행형 설치 관리자의 다운로드 및 실행 단계를 사용합니다. 이러한 시스템에서는 인터넷에서 직접 CloudWatch 로그 에이전트를 설치할 수 없습니다.

   > **Note**
   >
   > 아래 명령을 실행하기 전에 Ubuntu에서 `apt-get update`를 실행합니다.

   인터넷에서 직접 실행하려면 다음 명령을 사용하고 프롬프트의 메시지를 따릅니다.

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

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

위의 명령이 작동하지 않으면 다음 작업을 시도합니다.

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

단독으로 다운로드 및 실행하려면 다음 명령을 사용하고 프롬프트의 메시지를 따릅니다.

```
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
```

us-east-1, us-west-1, us-west-2, ap-south-1, ap-northeast-2, ap-southeast-1, ap-southeast-2, ap-southeast-2, ap-southeast-2, ap-northeast-2, ap-northeast-2, ap-northeast-2, ap-northeast-2를 지정하여 CloudWatch 로그 에이전트를 설치할 수 있습니다. ap-northea-1, eu-central-1, eu-west-1 또는 sa-east-1 지역

> **Note**
>
> 최신 버전과 `awslogs-agent-setup`의 버전 이력에 대한 자세한 내용은 CHANGELOG.txt를 참조하세요.

로그 에이전트 설치 CloudWatch 프로그램은 설치 중에 특정 정보가 필요합니다. 시작에 앞서 모니터링할 로그 파일과 타임스탬프 형식을 알아야 합니다. 또한 다음 정보를 이미 확보하고 있어야 합니다.

Item	설명
AWS 액세스 키 ID	IAM 역할을 사용하고 있는 경우에는 Enter 키를 누릅니다. 그렇지 않으면 AWS 액세스 키 ID를 입력합니다.
AWS 비밀 액세스 키	IAM 역할을 사용하고 있는 경우에는 Enter 키를 누릅니다. 그렇지 않으면 AWS 보안 액세스 키를 입력하세요.
기본 리전 이름	Enter를 누릅니다. 기본값은 us-east-2입니다. 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 또는 sa-east-1 리전으로 설정할 수 있습니다.
기본 출력 형식	공백으로 남겨 두고 Enter를 누릅니다.
업로드할 로그 파일의 경로	전송할 로그 데이터가 포함된 파일의 위치입니다. 설치 프로그램이 경로를 제안합니다.
대상 로그 그룹 이름	로그 그룹의 이름. 설치 프로그램이 로그 그룹 이름을 제안합니다.
대상 로그 스트림 이름	기본적으로 이 이름은 호스트 이름입니다. 설치 프로그램이 호스트 이름을 제안합니다.
타임스탬프 형식	지정된 로그 파일 내의 타임스탬프 형식을 지정합니다. 사용자 지정을 선택해서 자체 형식을 지정합니다.
초기 위치	데이터가 업로드된 방법입니다. 이를 start_of_file로 설정하면 데이터 파일에 모든 것이 업로드됩니다. end_of_file로 설정하면 새로 추가된 데이터만 업로드됩니다.

이러한 단계가 완료되면 설치 프로그램이 또 다른 로그 파일 구성에 대해 묻습니다. 각 로그 파일에서 원하는 횟수 만큼 프로세스를 실행할 수 있습니다. 더 이상 모니터링할 로그 파일이 없고 설치 프로그램에 또 다른 로그를 설정하라는 프롬프트 메시지가 나타나면 N을 선택합니다. 에이전

트 구성 파일을 설정하는 방법에 대한 자세한 내용은 CloudWatch 로그 에이전트 참조 섹션을 참조하세요.

> **Note**
>
> 단일 로그 스트림으로 데이터를 전송하기 위해 여러 개의 로그 소스를 구성하는 것이 불가능합니다.

3. 에이전트가 잠시 실행된 후 CloudWatch 콘솔에 새로 생성된 로그 그룹과 로그 스트림이 표시되어야 합니다.

   자세한 정보는 Logs로 전송된 로그 데이터 보기 CloudWatch 을 참조하세요.

빠른 시작: 시작 시 EC2 Linux 인스턴스에 CloudWatch Logs 에이전트를 설치하고 구성합니다.

> **Tip**
>
> 이 섹션에서 설명한 이전 CloudWatch Logs 에이전트는 지원 중단될 예정입니다. 대신 로그와 지표를 모두 수집할 수 있는 새 통합 CloudWatch 에이전트를 사용하는 것이 좋습니다. 또한 이전 CloudWatch Logs 에이전트에는 Python 3.3 또는 이전 버전이 필요하며 이러한 버전은 기본적으로 새 EC2 인스턴스에 설치되지 않습니다. 통합 CloudWatch 에이전트에 대한 자세한 내용은 에이전트 설치를 참조하십시오. CloudWatch
> 이 섹션의 나머지 부분에서는 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

시작 시 EC2 Linux 인스턴스에 이전 CloudWatch Logs 에이전트 설치

시작 시 파라미터 정보를 인스턴스에 전달할 수 있는 Amazon EC2의 기능인 Amazon EC2 사용자 데이터를 사용하여 해당 인스턴스에 Logs 에이전트를 설치하고 구성할 CloudWatch 수 있습니다. CloudWatch Logs 에이전트 설치 및 구성 정보를 Amazon EC2에 전달하려면 Amazon S3 버킷과 같은 네트워크 위치에 구성 파일을 제공하면 됩니다.

단일 로그 스트림으로 데이터를 전송하기 위해 여러 개의 로그 소스를 구성하는 것이 불가능합니다.

전제 조건

모든 로그 그룹 및 로그 스트림을 설명하는 에이전트 구성 파일을 생성합니다. 모니터링할 로그 파일을 비롯해 로그 파일을 업로드할 로그 그룹 및 로그 스트림을 설명하는 텍스트 파일입니다. 에이전트는 이

구성 파일을 사용하여 여기에 설명된 모든 로그 파일들에 대한 모니터링 및 업로드를 시작합니다. 에이전트 구성 파일을 설정하는 방법에 대한 자세한 내용은 CloudWatch 로그 에이전트 참조 섹션을 참조하세요.

다음은 Amazon Linux 2를 위한 에이전트 구성 파일 예제입니다.

```
[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
```

다음은 Ubuntu을 위한 에이전트 구성 파일 예제입니다.

```
[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
```

IAM 역할을 구성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택한 후 Create Policy를 선택합니다.
3. 정책 생성 페이지의 Create Your Own Policy(자체 정책 생성)에서 선택을 선택합니다. 사용자 지정 정책 생성에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2용 IAM 정책을 참조하십시오.
4. 정책 검토 페이지의 정책 이름에 정책 이름을 입력합니다.
5. 정책 문서에 다음 정책을 붙여넣습니다.

```
{
    "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:::myawsbucket/*"
            ]
        }
    ]
}
```

6. 정책 생성을 선택하세요.
7. 탐색 창에서 역할, 새 역할 생성을 선택합니다.
8. Set Role Name(역할 이름 설정) 페이지에서 역할 이름을 입력한 다음 다음 단계를 선택합니다.
9. 역할 유형 선택 페이지에서 Amazon EC2 옆에 있는 선택을 선택합니다.
10. 정책 연결 페이지의 테이블 헤더에서 정책 유형과 Customer Managed(고객 관리형)를 선택합니다.
11. 생성한 IAM 정책을 선택하고 다음 단계를 선택합니다.
12. Create Role(역할 생성)을 선택합니다.

    사용자 및 정책에 대한 자세한 내용은 IAM 사용 설명서의 IAM 사용자 및 그룹과 IAM 정책 관리를 참조하세요.

새 인스턴스를 시작하고 로그를 활성화하려면 CloudWatch

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 인스턴스 시작을 선택합니다.

자세한 내용은 Amazon EC2에서 인스턴스 시작 사용 설명서를 참조하십시오.

3. 1단계: Amazon Machine Image(AMI) 선택 페이지에서 시작할 Linux 인스턴스 유형을 선택하고 2단계: 인스턴스 유형 선택 페이지에서 다음: 인스턴스 세부 정보 구성을 선택합니다.

   cloud-init가 Amazon Machine Image(AMI)에 포함되어 있는지 확인합니다. Amazon Linux AMI와 우분투 및 RHEL용 AMI에는 이미 cloud-init가 포함되어 있지만 CentOS 및 기타 AMI에는 포함되어 있지 않을 수 있습니다. AWS Marketplace

4. 3단계: 인스턴스 정보 구성 페이지에서 IAM 역할에 대해 생성한 IAM 역할을 선택합니다.

5. 고급 세부 정보의 사용자 데이터에 다음 스크립트를 붙여넣습니다. 그런 다음, -c 옵션의 값을 에이전트 구성 파일의 위치로 변경하여 해당 스크립트를 업데이트합니다.

   ```
   #!/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://DOC-EXAMPLE-BUCKET1/my-config-file
   ```

6. 인스턴스를 더 변경하고 시작 설정을 검토한 다음 시작을 선택합니다.

7. 에이전트가 잠시 실행되면 CloudWatch 콘솔에서 새로 생성된 로그 그룹과 로그 스트림을 확인할 수 있습니다.

   자세한 정보는 Logs로 전송된 로그 데이터 보기 CloudWatch 을 참조하세요.

빠른 시작: Windows Server 2016을 실행하는 Amazon EC2 인스턴스가 로그 에이전트를 사용하여 로그로 로그를 전송할 수 있도록 CloudWatch 합니다 CloudWatch .

> **Tip**
>
> CloudWatch EC2 인스턴스 및 온프레미스 서버에서 로그와 지표를 모두 수집할 수 있는 새로운 통합 에이전트가 포함되어 있습니다. 최신 통합 에이전트를 사용하는 것이 좋습니다. CloudWatch 자세한 정보는 CloudWatch 로그 시작하기을 참조하세요.
> 이 섹션의 나머지 부분에서는 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

Windows Server 2016을 실행하는 Amazon EC2 인스턴스가 이전 CloudWatch 로그 에이전트를 사용하여 로그에 CloudWatch 로그를 전송할 수 있도록 합니다.

Windows Server 2016을 실행하는 인스턴스가 로그로 로그를 전송하도록 하는 데는 여러 가지 방법을 사용할 수 있습니다. CloudWatch 이 섹션의 단계에서는 Systems Manager Run Command를 사용합니다. 가능한 다른 방법에 대한 자세한 내용은 CloudWatchAmazon에 로그, 이벤트 및 성능 카운터 전송을 참조하십시오.

단계

- 샘플 구성 파일 다운로드
- JSON 파일을 다음과 같이 구성하십시오. CloudWatch
- Systems Manager 대한 IAM 역할 생성
- Systems Manager 사전 조건 확인
- 인터넷 액세스 확인
- Systems Manager 실행 명령을 사용하여 CloudWatch 로그 활성화

샘플 구성 파일 다운로드

컴퓨터에 다음 샘플 파일을 다운로드합니다. `AWS.EC2.Windows.CloudWatch.json`

JSON 파일을 다음과 같이 구성하십시오. CloudWatch

구성 파일에 선택 사항을 CloudWatch 지정하여 전송할 로그를 결정합니다. 이 파일을 만들고 선택 항목을 지정하는 프로세스를 완료하는 데 30분 이상 걸릴 수 있습니다. 이 작업을 한 번 완료한 후 모든 인스턴스에 구성 파일을 재사용할 수 있습니다.

단계

- 1단계: CloudWatch 로그 활성화
- 2단계: 설정 구성 CloudWatch
- 3단계: 전송할 데이터 구성
- 4단계: 흐름 제어(Flow Control) 구성
- 5단계: JSON 내용 저장

1단계: CloudWatch 로그 활성화

JSON 파일의 상단에서 `IsEnabled`를 "false"에서 "true"로 변경합니다.

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

2단계: 설정 구성 CloudWatch

자격 증명, 리전, 로그 그룹 이름, 로그 스트림 네임스페이스를 지정합니다. 이렇게 하면 인스턴스가 로그 데이터를 CloudWatch Logs로 전송할 수 있습니다. 동일한 로그 데이터를 다른 위치로 보내려면 고유한 ID (예: "CloudWatchLogs2" 및 CloudWatchLogs 3") 가 있는 섹션을 추가하고 각 ID에 대해 다른 지역을 추가할 수 있습니다.

로그 데이터를 Logs로 전송하도록 설정을 구성하려면 CloudWatch

1. JSON 파일에서 `CloudWatchLogs` 섹션을 찾습니다.

   ```
   {
       "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. `AccessKey` 및 `SecretKey` 필드는 비워둡니다. IAM 역할을 사용하여 자격 증명을 구성합니다.
3. `Region`에 로그 데이터를 보내려는 리전을 입력합니다(예: `us-east-2`).
4. `LogGroup`에 로그 그룹의 이름을 입력합니다. 이 이름은 CloudWatch 콘솔의 로그 그룹 화면에 표시됩니다.
5. `LogStream`에 대상 로그 스트림을 입력합니다. 이 이름은 CloudWatch 콘솔의 로그 그룹 > 스트림 화면에 표시됩니다.

   `{instance_id}`를 사용하는 경우 기본적으로 로그 스트림 이름은 이 인스턴스의 인스턴스 ID입니다.

   아직 존재하지 않는 로그 스트림 이름을 지정하면 CloudWatch Logs가 자동으로 이름을 생성합니다. 리터럴 문자열, 미리 정의된 변수 `{instance_id}`, `{hostname}` 및 `{ip_address}`, 또는 이들의 조합을 사용하여 로그 스트림 이름을 정의할 수 있습니다.

3단계: 전송할 데이터 구성

이벤트 로그 데이터, Windows용 이벤트 추적 (ETW) 데이터 및 기타 로그 데이터를 Logs로 CloudWatch 보낼 수 있습니다.

Windows 응용 프로그램 이벤트 로그 데이터를 로그로 보내려면 CloudWatch

1. JSON 파일에서 `ApplicationEventLog` 섹션을 찾습니다.

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

2. `Levels`에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

보안 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 `SecurityEventLog` 섹션을 찾습니다.

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

```
    }
},
```

2. 모든 메시지를 업로드하려면 `Levels`에 **7**을 입력합니다.

시스템 이벤트 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 `SystemEventLog` 섹션을 찾습니다.

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

2. `Levels`에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

다른 유형의 이벤트 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 새 섹션을 추가합니다. 각 섹션에는 고유한 `Id`가 있어야 합니다.

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

```
    }
},
```

2. Id에 업로드할 로그의 이름을 입력합니다(예: **WindowsBackup**).
3. LogName에 업로드할 로그의 이름을 입력합니다. 로그 이름은 다음과 같이 확인할 수 있습니다.
   a. 이벤트 뷰어를 엽니다.
   b. 탐색 창에서 Applications and Services Logs(응용 프로그램 및 서비스 로그)를 선택합니다.
   c. 로그로 이동한 다음 작업, 속성을 선택합니다.
4. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.
   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

Windows용 이벤트 추적 데이터를 로그로 보내려면 CloudWatch

ETW(Windows용 이벤트 추적)는 애플리케이션이 로그를 기록할 수 있는 효율적이고 세부적인 로깅 메커니즘을 제공합니다. 로깅 세션을 시작하고 중지할 수 있는 세션 관리자가 각 ETW를 제어합니다. 각 세션에는 한 공급자와 하나 또는 그 이상의 소비자가 있습니다.

1. JSON 파일에서 ETW 섹션을 찾습니다.

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

2. LogName에 업로드할 로그의 이름을 입력합니다.

3. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

사용자 지정 로그 (모든 텍스트 기반 로그 파일) 를 로그로 보내려면 CloudWatch

1. JSON 파일에서 CustomLogs 섹션을 찾습니다.

```
{
    "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. LogDirectoryPath에서 인스턴스에 로그가 저장된 경로를 입력합니다.
3. TimestampFormat에 사용할 타임스탬프 형식을 입력합니다. 지원되는 값에 대한 자세한 내용은 MSDN의 사용자 지정 날짜 및 시간 형식 문자열 주제를 참조하세요.

> **Important**
>
> 원본 로그 파일에는 각 로그 줄의 시작 부분에 타임스탬프가 있어야 하고 타임스탬프 뒤에는 공백이 있어야 합니다.

4. `Encoding`에 사용할 파일 인코딩을 입력합니다(예: UTF-8). 지원되는 값 목록은 MSDN에서 Encoding Class 항목을 참조하세요.

   > **Note**
   >
   > 표시 이름이 아니라 인코딩 이름을 사용하세요.

5. (선택 사항) `Filter`에 로그 이름의 접두사를 입력합니다. 모든 파일을 모니터링하려면 이 파라미터를 공백으로 둡니다. 지원되는 값에 대한 자세한 내용은 MSDN FileSystemWatcherFilter 속성 항목을 참조하십시오.

6. (선택 사항) `CultureName`에 타임스탬프가 기록되는 로캘을 입력합니다. `CultureName`이 공백이면 기본적으로 Windows 인스턴스에서 현재 사용 중인 것과 같은 로캘로 설정됩니다. 자세한 내용은 MSDN에서 제품 동작 주제의 표에 있는 `Language tag` 열을 참조하세요.

   > **Note**
   >
   > `div`, `div-MV`, `hu` 및 `hu-HU` 값은 지원되지 않습니다.

7. (선택 사항) `TimeZoneKind`에 `Local` 또는 `UTC`를 입력합니다. 로그의 타임스탬프에 표준 시간대 정보가 포함되어 있지 않을 때 이렇게 설정하여 표준 시간대 정보를 제공할 수 있습니다. 이 매개변수를 비워 두고 타임스탬프에 시간대 정보가 포함되지 않은 경우 CloudWatch 로그는 현지 시간대로 기본 설정됩니다. 타임스탬프에 표준 시간대 정보가 이미 포함된 경우 이 파라미터는 무시됩니다.

8. (선택 사항) `LineCount`에 로그 파일을 식별할 헤더의 줄 수를 입력합니다. 예를 들어 IIS 로그 파일에 있는 헤더들은 사실상 동일합니다. **5**를 입력하면 로그 파일 헤더에서 처음 나오는 세 줄을 읽어 식별하는 식입니다. IIS 로그 파일에서 처음 나오는 세 줄은 날짜와 타임스탬프이지만, 로그 파일 간에 타임스탬프가 반드시 다르지는 않습니다. 이러한 이유로, 로그 파일에 고유한 지문을 남기기 위해 실제 로그 데이터를 한 줄 이상 포함하는 것이 좋습니다.

IIS 로그 데이터를 Logs로 보내려면 CloudWatch

1. JSON 파일에서 `IISLog` 섹션을 찾습니다.

```
{
    "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. `LogDirectoryPath`에 개별 사이트에 대해 IIS 로그를 저장할 폴더를 입력합니다(예: `C:\inetpub\logs\LogFiles\W3SVC`*n*).

> **Note**
>
> W3C 로그 형식만 지원됩니다. IIS, NCSA 및 사용자 지정 형식은 지원되지 않습니다.

3. `TimestampFormat`에 사용할 타임스탬프 형식을 입력합니다. 지원되는 값에 대한 자세한 내용은 MSDN의 사용자 지정 날짜 및 시간 형식 문자열 주제를 참조하세요.
4. `Encoding`에 사용할 파일 인코딩을 입력합니다(예: UTF-8). 지원되는 값에 대한 자세한 내용은 MSDN의 인코딩 클래스 주제를 참조하세요.

> **Note**
>
> 표시 이름이 아니라 인코딩 이름을 사용하세요.

5. (선택 사항) `Filter`에 로그 이름의 접두사를 입력합니다. 모든 파일을 모니터링하려면 이 파라미터를 공백으로 둡니다. 지원되는 값에 대한 자세한 내용은 MSDN FileSystemWatcherFilter 속성 항목을 참조하십시오.
6. (선택 사항) `CultureName`에 타임스탬프가 기록되는 로캘을 입력합니다. `CultureName`이 공백이면 기본적으로 Windows 인스턴스에서 현재 사용 중인 것과 같은 로캘로 설정됩니다. 지원되는 값에 대한 자세한 내용은 MSDN에서 제품 동작 주제의 표에 있는 `Language tag` 열을 참조하세요.

> **Note**
>
> `div`, `div-MV`, `hu` 및 `hu-HU` 값은 지원되지 않습니다.

7. (선택 사항) TimeZoneKind에 Local 또는 UTC를 입력합니다. 로그의 타임스탬프에 표준 시간대 정보가 포함되어 있지 않을 때 이렇게 설정하여 표준 시간대 정보를 제공할 수 있습니다. 이 매개 변수를 비워 두고 타임스탬프에 시간대 정보가 포함되지 않은 경우 CloudWatch 로그는 현지 시간대로 기본 설정됩니다. 타임스탬프에 표준 시간대 정보가 이미 포함된 경우 이 파라미터는 무시됩니다.
8. (선택 사항) LineCount에 로그 파일을 식별할 헤더의 줄 수를 입력합니다. 예를 들어 IIS 로그 파일에 있는 헤더들은 사실상 동일합니다. **5**를 입력하면 로그 파일 헤더에서 처음 나오는 다섯 줄을 읽어 식별하는 식입니다. IIS 로그 파일에서 처음 나오는 세 줄은 날짜와 타임스탬프이지만, 로그 파일 간에 타임스탬프가 반드시 다르지는 않습니다. 이러한 이유로, 로그 파일에 고유한 지문을 남기기 위해 실제 로그 데이터를 한 줄 이상 포함하는 것이 좋습니다.

4단계: 흐름 제어(Flow Control) 구성

각 데이터 형식의 Flows 섹션에 해당 대상이 있어야 합니다. 예를 들어 사용자 지정 로그, ETW 로그 및 시스템 CloudWatch 로그를 Logs로 (CustomLogs,ETW,SystemEventLog),CloudWatchLogs 보내려면 섹션에 추가하십시오. Flows

> **Warning**
>
> 유효하지 않은 단계를 추가하면 흐름이 차단됩니다. 예를 들어, 디스크 지표 단계를 추가하지만 인스턴스에 디스크가 없는 경우 흐름의 모든 단계가 차단됩니다.

같은 로그 파일을 두 개 이상의 대상으로 보낼 수 있습니다. 예를 들어, 애플리케이션 로그를 CloudWatchLogs 섹션에 정의된 두 개의 대상으로 보내려면 Flows 섹션에 ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)를 추가합니다.

흐름 제어를 구성하는 방법

1. AWS.EC2.Windows.CloudWatch.json 파일에서 Flows 섹션을 찾습니다.

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

```
}
```

2. Flows에서 업로드할 각 데이터 형식(예: ApplicationEventLog) 및 대상(예: CloudWatchLogs)을 추가합니다.

5단계: JSON 내용 저장

이제 JSON 파일 편집이 끝났습니다. 이것을 저장하여 다른 창의 텍스트 편집기에 파일 내용을 붙여넣습니다. 이 절차의 이후 단계에서 이 파일 내용이 필요할 것입니다.

Systems Manager 대한 IAM 역할 생성

Systems Manager Run Command를 사용할 때 인스턴스 자격 증명에 대한 IAM 역할이 필요합니다. 이 역할은 Systems Manager가 인스턴스에 대한 작업을 수행하도록 허용합니다. 자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 대한 보안 역할 구성을 참조하세요. 기존 인스턴스에 IAM 역할을 연결하는 방법에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 IAM 역할 연결을 참조하십시오.

Systems Manager 사전 조건 확인

Systems Manager Run Command를 사용하여 CloudWatch 로그와의 통합을 구성하기 전에 인스턴스가 최소 요구 사항을 충족하는지 확인하십시오. 자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 사전 조건 단원을 참조하세요.

인터넷 액세스 확인

Amazon EC2 Windows Server 인스턴스 및 관리형 인스턴스에 로그 및 이벤트 데이터를 전송하려면 아웃바운드 인터넷 액세스 권한이 있어야 합니다. CloudWatch 인터넷 액세스를 구성하는 방법에 대한 자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이 단원을 참조하세요.

Systems Manager 실행 명령을 사용하여 CloudWatch 로그 활성화

Run Command를 사용하면 요청 시 인스턴스의 구성을 관리할 수 있습니다. Systems Manager 문서, 파라미터를 지정하고 하나 이상의 인스턴스에 명령을 실행합니다. 인스턴스의 SSM 에이전트는 명령을 처리하고 지정된 대로 인스턴스를 구성합니다.

명령 실행을 사용하여 CloudWatch 로그와의 통합을 구성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. SSM 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

3. 탐색 창에서 Run Command를 선택합니다.
4. Run a command를 선택합니다.
5. 명령 문서의 경우 AWS-를 선택합니다ConfigureCloudWatch.
6. Target 인스턴스의 경우 CloudWatch Logs와 통합할 인스턴스를 선택합니다. 이 목록에서 인스턴스가 보이지 않는 경우, Run Command에 대해 구성되지 않은 인스턴스일 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 Systems Manager 사전 요구 사항을 참조하십시오.
7. 상태에서 활성을 선택합니다.
8. 속성에 이전 작업에서 생성한 JSON 내용을 복사하여 붙여넣습니다.
9. 나머지 옵션 필드를 완성하고 Run을 선택합니다.

다음 절차를 이용해 Amazon EC2 콘솔에서 명령 실행 결과를 확인합니다.

콘솔에서 명령 출력을 보는 방법

1. 명령을 선택합니다.
2. 출력 탭을 선택합니다.
3. 출력 보기를 선택합니다. 명령 출력 페이지에 명령 실행 결과가 표시됩니다.

빠른 시작: 윈도우 서버 2012 및 윈도우 서버 2008을 실행하는 Amazon EC2 인스턴스가 로그를 로그로 전송하도록 활성화합니다. CloudWatch

> **Tip**
>
> CloudWatch EC2 인스턴스 및 온프레미스 서버에서 로그와 지표를 모두 수집할 수 있는 새로운 통합 에이전트가 포함되어 있습니다. 최신 통합 에이전트를 사용하는 것이 좋습니다. CloudWatch 자세한 정보는 CloudWatch 로그 시작하기을 참조하세요.
> 이 섹션의 나머지 부분에서는 이전 CloudWatch Logs 에이전트의 사용에 대해 설명합니다.

윈도우 서버 2012 및 윈도우 서버 2008을 실행하는 Amazon EC2 인스턴스가 로그를 로그로 전송하도록 활성화합니다. CloudWatch

다음 단계를 사용하여 Windows Server 2012 및 Windows Server 2008을 실행하는 인스턴스가 로그로 로그를 보낼 수 있도록 할 수 CloudWatch 있습니다.

샘플 구성 파일 다운로드

컴퓨터에 다음 샘플 JSON 파일을 다운로드합니다. `AWS.EC2.Windows.CloudWatch.json` 다음 단계에서 이 파일을 편집합니다.

다음에 대해 JSON 파일을 구성하십시오. CloudWatch

JSON 구성 파일에서 선택 사항을 CloudWatch 지정하여 전송할 로그를 결정합니다. 이 파일을 만들고 선택 항목을 지정하는 프로세스를 완료하는 데 30분 이상 걸릴 수 있습니다. 이 작업을 한 번 완료한 후 모든 인스턴스에 구성 파일을 재사용할 수 있습니다.

단계

- 1단계: 로그 활성화 CloudWatch
- 2단계: 설정 구성 CloudWatch
- 3단계: 전송할 데이터 구성
- 4단계: 흐름 제어(Flow Control) 구성

1단계: 로그 활성화 CloudWatch

JSON 파일의 상단에서 `IsEnabled`를 "false"에서 "true"로 변경합니다.

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

2단계: 설정 구성 CloudWatch

자격 증명, 리전, 로그 그룹 이름, 로그 스트림 네임스페이스를 지정합니다. 이렇게 하면 인스턴스가 로그 데이터를 CloudWatch Logs로 전송할 수 있습니다. 동일한 로그 데이터를 다른 위치로 보내려면 고유한 ID (예: "CloudWatchLogs2" 및 CloudWatchLogs 3") 가 있는 섹션을 추가하고 각 ID에 대해 다른 지역을 추가할 수 있습니다.

로그 데이터를 Logs로 전송하도록 설정을 구성하려면 CloudWatch

1. JSON 파일에서 `CloudWatchLogs` 섹션을 찾습니다.

```
{
    "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. AccessKey 및 SecretKey 필드는 비워둡니다. IAM 역할을 사용하여 자격 증명을 구성합니다.
3. Region에 로그 데이터를 보내려는 리전을 입력합니다(예: us-east-2).
4. LogGroup에 로그 그룹의 이름을 입력합니다. 이 이름은 CloudWatch 콘솔의 로그 그룹 화면에 표시됩니다.
5. LogStream에 대상 로그 스트림을 입력합니다. 이 이름은 CloudWatch 콘솔의 로그 그룹 > 스트림 화면에 표시됩니다.

   {instance_id}를 사용하는 경우 기본적으로 로그 스트림 이름은 이 인스턴스의 인스턴스 ID입니다.

   아직 존재하지 않는 로그 스트림 이름을 지정하면 CloudWatch Logs가 자동으로 이름을 생성합니다. 리터럴 문자열, 미리 정의된 변수 {instance_id}, {hostname} 및 {ip_address}, 또는 이들의 조합을 사용하여 로그 스트림 이름을 정의할 수 있습니다.

3단계: 전송할 데이터 구성

이벤트 로그 데이터, Windows용 이벤트 추적 (ETW) 데이터 및 기타 로그 데이터를 Logs로 CloudWatch 보낼 수 있습니다.

Windows 응용 프로그램 이벤트 로그 데이터를 로그로 보내려면 CloudWatch

1. JSON 파일에서 ApplicationEventLog 섹션을 찾습니다.

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

```
},
```

2. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

보안 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 SecurityEventLog 섹션을 찾습니다.

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

2. 모든 메시지를 업로드하려면 Levels에 **7**을 입력합니다.

시스템 이벤트 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 SystemEventLog 섹션을 찾습니다.

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

```
},
```

2. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

다른 유형의 이벤트 로그 데이터를 CloudWatch Logs로 보내려면

1. JSON 파일에서 새 섹션을 추가합니다. 각 섹션에는 고유한 Id가 있어야 합니다.

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

2. Id에 업로드할 로그의 이름을 입력합니다(예: **WindowsBackup**).
3. LogName에 업로드할 로그의 이름을 입력합니다. 로그 이름은 다음과 같이 확인할 수 있습니다.

   a. 이벤트 뷰어를 엽니다.

   b. 탐색 창에서 Applications and Services Logs(응용 프로그램 및 서비스 로그)를 선택합니다.

   c. 로그로 이동한 다음 작업, 속성을 선택합니다.
4. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

Windows용 이벤트 추적 데이터를 로그로 보내려면 CloudWatch

ETW(Windows용 이벤트 추적)는 애플리케이션이 로그를 기록할 수 있는 효율적이고 세부적인 로깅 메커니즘을 제공합니다. 로깅 세션을 시작하고 중지할 수 있는 세션 관리자가 각 ETW를 제어합니다. 각 세션에는 한 공급자와 하나 또는 그 이상의 소비자가 있습니다.

1. JSON 파일에서 ETW 섹션을 찾습니다.

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

2. LogName에 업로드할 로그의 이름을 입력합니다.
3. Levels에서 업로드할 메시지의 유형을 지정합니다. 다음 값 중 하나를 지정할 수 있습니다.

   - **1** - 오류 메시지만 업로드됩니다.
   - **2** - 경고 메시지만 업로드됩니다.
   - **4** - 정보 메시지만 업로드됩니다.

   값을 적절히 조합하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 값 **3**을 지정하면 오류 메시지(**1**)와 경고 메시지(**2**)가 업로드됩니다. 값 **7**을 지정하면 오류 메시지(**1**), 경고 메시지(**2**) 및 정보 메시지(**4**)가 업로드됩니다.

사용자 지정 로그 (모든 텍스트 기반 로그 파일) 를 로그로 보내려면 CloudWatch

1. JSON 파일에서 CustomLogs 섹션을 찾습니다.

```
{
    "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. `LogDirectoryPath`에서 인스턴스에 로그가 저장된 경로를 입력합니다.
3. `TimestampFormat`에 사용할 타임스탬프 형식을 입력합니다. 지원되는 값에 대한 자세한 내용은 MSDN의 사용자 지정 날짜 및 시간 형식 문자열 주제를 참조하세요.

> **Important**
>
> 원본 로그 파일에는 각 로그 줄의 시작 부분에 타임스탬프가 있어야 하고 타임스탬프 뒤에는 공백이 있어야 합니다.

4. `Encoding`에 사용할 파일 인코딩을 입력합니다(예: UTF-8). 지원되는 값에 대한 자세한 내용은 MSDN의 인코딩 클래스 주제를 참조하세요.

> **Note**
>
> 표시 이름이 아니라 인코딩 이름을 사용하세요.

5. (선택 사항) `Filter`에 로그 이름의 접두사를 입력합니다. 모든 파일을 모니터링하려면 이 파라미터를 공백으로 둡니다. 지원되는 값에 대한 자세한 내용은 MSDN FileSystemWatcherFilter 속성 항목을 참조하십시오.
6. (선택 사항) `CultureName`에 타임스탬프가 기록되는 로캘을 입력합니다. `CultureName`이 공백이면 기본적으로 Windows 인스턴스에서 현재 사용 중인 것과 같은 로캘로 설정됩니다. 지원되는 값에 대한 자세한 내용은 MSDN에서 제품 동작 주제의 표에 있는 `Language tag` 열을 참조하세요.

> **Note**
>
> `div`, `div-MV`, `hu` 및 `hu-HU` 값은 지원되지 않습니다.

7. (선택 사항) `TimeZoneKind`에 `Local` 또는 `UTC`를 입력합니다. 로그의 타임스탬프에 표준 시간대 정보가 포함되어 있지 않을 때 이렇게 설정하여 표준 시간대 정보를 제공할 수 있습니다. 이 매개 변수를 비워 두고 타임스탬프에 시간대 정보가 포함되지 않은 경우 CloudWatch 로그는 현지 시간대로 기본 설정됩니다. 타임스탬프에 표준 시간대 정보가 이미 포함된 경우 이 파라미터는 무시됩니다.
8. (선택 사항) `LineCount`에 로그 파일을 식별할 헤더의 줄 수를 입력합니다. 예를 들어 IIS 로그 파일에 있는 헤더들은 사실상 동일합니다. **5**를 입력하면 로그 파일 헤더에서 처음 나오는 세 줄을 읽어 식별하는 식입니다. IIS 로그 파일에서 처음 나오는 세 줄은 날짜와 타임스탬프이지만, 로그 파일 간에 타임스탬프가 반드시 다르지는 않습니다. 이러한 이유로, 로그 파일에 고유한 지문을 남기기 위해 실제 로그 데이터를 한 줄 이상 포함하는 것이 좋습니다.

IIS 로그 데이터를 Logs로 보내려면 CloudWatch

1. JSON 파일에서 `IISLog` 섹션을 찾습니다.

```
{
    "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. `LogDirectoryPath`에 개별 사이트에 대해 IIS 로그를 저장할 폴더를 입력합니다(예: `C:\inetpub\logs\LogFiles\W3SVC`*n*).

> **Note**
>
> W3C 로그 형식만 지원됩니다. IIS, NCSA 및 사용자 지정 형식은 지원되지 않습니다.

3. `TimestampFormat`에 사용할 타임스탬프 형식을 입력합니다. 지원되는 값에 대한 자세한 내용은 MSDN의 사용자 지정 날짜 및 시간 형식 문자열 주제를 참조하세요.
4. `Encoding`에 사용할 파일 인코딩을 입력합니다(예: UTF-8). 지원되는 값에 대한 자세한 내용은 MSDN의 인코딩 클래스 주제를 참조하세요.

> **Note**
>
> 표시 이름이 아니라 인코딩 이름을 사용하세요.

5. (선택 사항) `Filter`에 로그 이름의 접두사를 입력합니다. 모든 파일을 모니터링하려면 이 파라미터를 공백으로 둡니다. 지원되는 값에 대한 자세한 내용은 MSDN FileSystemWatcherFilter 속성 항목을 참조하십시오.
6. (선택 사항) `CultureName`에 타임스탬프가 기록되는 로캘을 입력합니다. `CultureName`이 공백이면 기본적으로 Windows 인스턴스에서 현재 사용 중인 것과 같은 로캘로 설정됩니다. 지원되는 값에 대한 자세한 내용은 MSDN에서 제품 동작 주제의 표에 있는 `Language tag` 열을 참조하세요.

> **Note**
>
> `div`, `div-MV`, `hu` 및 `hu-HU` 값은 지원되지 않습니다.

7. (선택 사항) `TimeZoneKind`에 `Local` 또는 `UTC`를 입력합니다. 로그의 타임스탬프에 표준 시간대 정보가 포함되어 있지 않을 때 이렇게 설정하여 표준 시간대 정보를 제공할 수 있습니다. 이 매개변수를 비워 두고 타임스탬프에 시간대 정보가 포함되지 않은 경우 CloudWatch 로그는 현지 시간대로 기본 설정됩니다. 타임스탬프에 표준 시간대 정보가 이미 포함된 경우 이 파라미터는 무시됩니다.
8. (선택 사항) `LineCount`에 로그 파일을 식별할 헤더의 줄 수를 입력합니다. 예를 들어 IIS 로그 파일에 있는 헤더들은 사실상 동일합니다. **5**를 입력하면 로그 파일 헤더에서 처음 나오는 다섯 줄을 읽어 식별하는 식입니다. IIS 로그 파일에서 처음 나오는 세 줄은 날짜와 타임스탬프이지만, 로그 파일 간에 타임스탬프가 반드시 다르지는 않습니다. 이러한 이유로, 로그 파일에 고유한 지문을 남기기 위해 실제 로그 데이터를 한 줄 이상 포함하는 것이 좋습니다.

4단계: 흐름 제어(Flow Control) 구성

각 데이터 형식의 Flows 섹션에 해당 대상이 있어야 합니다. 예를 들어 사용자 지정 로그, ETW 로그 및 시스템 CloudWatch 로그를 Logs로 (CustomLogs,ETW,SystemEventLog),CloudWatchLogs 보내려면 섹션에 추가하십시오. Flows

> **Warning**
>
> 유효하지 않은 단계를 추가하면 흐름이 차단됩니다. 예를 들어, 디스크 지표 단계를 추가하지만 인스턴스에 디스크가 없는 경우 흐름의 모든 단계가 차단됩니다.

같은 로그 파일을 두 개 이상의 대상으로 보낼 수 있습니다. 예를 들어, 애플리케이션 로그를 CloudWatchLogs 섹션에 정의된 두 개의 대상으로 보내려면 Flows 섹션에 ApplicationEventLog,(CloudWatchLogs,CloudWatchLogs2)를 추가합니다.

흐름 제어를 구성하는 방법

1. AWS.EC2.Windows.CloudWatch.json 파일에서 Flows 섹션을 찾습니다.

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

2. Flows에서 업로드할 각 데이터 형식(예: ApplicationEventLog) 및 대상(예: CloudWatchLogs)을 추가합니다.

이제 JSON 파일 편집이 끝났습니다. 이후 단계에서 사용하게 됩니다.

에이전트 시작

윈도우 서버 2012 또는 윈도우 서버 2008을 실행하는 Amazon EC2 인스턴스가 로그에 로그를 전송할 수 있도록 하려면 CloudWatch EC2Config 서비스 (. EC2Config.exe) 인스턴스에 EC2Config 4.0 이상이 있어야 다음 절차를 사용할 수 있습니다. 이전 버전의 EC2Config 사용에 대한 자세한 내

용은 Amazon EC2 사용 설명서의 EC2Config 3.x 또는 이전 버전을 사용하여 구성을 참조하십시오. CloudWatch

EC2Config 4.x를 사용하여 구성하려면 CloudWatch

1. 이 절차에서 앞서 편집한 `AWS.EC2.Windows.CloudWatch.json` 파일의 인코딩을 확인합니다. BOM 없는 UTF-8 인코딩만 지원됩니다. Windows Server 2008 - 2012 R2 인스턴스에서 다음 폴더에 파일을 저장합니다. `C:\Program Files\Amazon\SSM\Plugins\awsCloudWatch\`.
2. Windows 서비스 제어판을 사용하거나 다음 명령을 사용하여 SSM 에이전트 (`AmazonSSMAgent.exe`) 를 시작하거나 다시 시작합니다. PowerShell

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

SSM 에이전트가 다시 시작되면 구성 파일을 감지하고 통합을 위한 인스턴스를 구성합니다. CloudWatch 로컬 구성 파일에서 파라미터와 설정을 변경할 경우 SSM 에이전트를 다시 시작하여 변경 사항을 선택해야 합니다. 인스턴스에서 CloudWatch 통합을 비활성화하려면 구성 파일로 `IsEnabled` `false` 변경하여 변경 내용을 저장합니다.

빠른 시작: Chef를 사용하여 CloudWatch AWS OpsWorks Logs 에이전트를 설치합니다.

타사 시스템 및 클라우드 인프라 자동화 도구인 Chef를 사용하여 CloudWatch AWS OpsWorks Logs 에이전트를 설치하고 로그 스트림을 생성할 수 있습니다. Chef는 컴퓨터에 소프트웨어를 설치 및 구성하기 위해 작성하는 "레시피"와 구성 및 정책 배포 작업을 수행하기 위한 레시피 모음인 "쿡북"을 사용합니다. 자세한 내용은 Chef를 참조하세요.

아래의 Chef 레시피 예제는 각 EC2 인스턴스에서 하나의 로그 파일을 모니터링하는 방법을 보여줍니다. 레시피는 로그 그룹으로 스택 이름을 사용하고 로그 스트림 이름으로 인스턴스의 호스트 이름을 사용합니다. 여러 개의 로그 파일을 모니터링하려면 여러 로그 그룹 및 로그 스트림을 생성하도록 레시피를 확장해야 합니다.

1단계: 사용자 지정 레시피 생성

리포지토리를 만들어 레시피를 저장하세요. AWS OpsWorks Git 및 Subversion을 지원하거나 Amazon S3에 아카이브를 저장할 수 있습니다. 쿡북 리포지토리의 구조는 AWS OpsWorks 사용 설명서의 쿡북 리포지토리에서 설명합니다. 아래 예제는 쿡북의 이름이 `logs`라고 가정합니다. install.rb 레시피는 로그 에이전트를 설치합니다. CloudWatch 쿡북 예제 (-Cookbooks.zip) 도 다운로드할 수 있습니다. CloudWatchLogs

다음 코드가 포함된 metadata.rb라는 파일을 생성합니다.

```
#metadata.rb

name             'logs'
version          '0.0.1'
```

CloudWatch 로그 구성 파일 생성:

```
#config.rb

template "/tmp/cwlogs.cfg" do
  cookbook "logs"
  source "cwlogs.cfg.erb"
  owner "root"
  group "root"
  mode 0644
end
```

CloudWatch Logs 에이전트 다운로드 및 설치:

```
# install.rb

directory "/opt/aws/cloudwatch" do
  recursive true
end

remote_file "/opt/aws/cloudwatch/awslogs-agent-setup.py" do
  source "https://s3.amazonaws.com/aws-cloudwatch/downloads/latest/awslogs-agent-
setup.py"
  mode "0755"
end

  execute "Install CloudWatch Logs agent" do
  command "/opt/aws/cloudwatch/awslogs-agent-setup.py -n -r region -c /tmp/cwlogs.cfg"
  not_if { system "pgrep -f aws-logs-agent-setup" }
end
```

> **Note**
>
> 위의 예제에서 다음 리전 중 하나로 *region*을 대체합니다. 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 또는 sa-east-1.

에이전트 설치가 실패할 경우 python-dev 패키지가 설치되어 있는지 확인하세요. 설치되어 있지 않은 경우 다음 명령을 사용한 후 에이전트 설치를 다시 시도하세요.

```
sudo apt-get -y install python-dev
```

이 레시피는 기록할 파일 같이 다양한 속성을 지정하기 위해 수정이 가능한 cwlogs.cfg.erb 템플릿 파일을 사용합니다. 이들 속성에 대한 자세한 내용은 CloudWatch 로그 에이전트 참조 섹션을 참조하세요.

```
[general]
# Path to the AWSLogs agent's state file. Agent uses this file to maintain
# client side state across its executions.
state_file = /var/awslogs/state/agent-state


## Each log file is defined in its own section. The section name doesn't
## matter as long as its unique within this file.
#
#[kern.log]
#
## Path of log file for the agent to monitor and upload.
#
#file = /var/log/kern.log
#
## Name of the destination log group.
#
#log_group_name = kern.log
#
## Name of the destination log stream.
#
#log_stream_name = {instance_id}
#
## Format specifier for timestamp parsing.
#
#datetime_format = %b %d %H:%M:%S
```

```
#
#

[<%= node[:opsworks][:stack][:name] %>]
datetime_format = [%Y-%m-%d %H:%M:%S]
log_group_name = <%= node[:opsworks][:stack][:name].gsub(' ','_') %>
file = <%= node[:cwlogs][:logfile] %>
log_stream_name = <%= node[:opsworks][:instance][:hostname] %>
```

템플릿은 스택 구성 및 배포 JSON에서 해당되는 속성을 참조하여 스택 이름과 호스트 이름을 얻습니다. 기록할 파일을 지정하는 속성은 cwlogs 쿡북의 default.rb 속성 파일(logs/attributes/default.rb)에 정의되어 있습니다.

```
default[:cwlogs][:logfile] = '/var/log/aws/opsworks/opsworks-agent.statistics.log'
```

2단계: AWS OpsWorks 스택 생성

1. https://console.aws.amazon.com/opsworks/ 에서 AWS OpsWorks 콘솔을 엽니다.
2. OpsWorks 대시보드에서 Add stack (스택 추가) 을 선택하여 AWS OpsWorks 스택을 생성합니다.
3. Add stack(스택 추가) 화면에서 Chef 11 stack(Chef 11 스택)을 선택합니다.
4. 스택 이름에 이름을 입력합니다.
5. Use custom Chef Cookbooks(사용자 지정 Chef 쿡북 사용)에서 예를 선택합니다.
6. Repository type(리포지토리 유형)에서 사용할 리포지토리 유형을 선택합니다. 위의 예제를 사용하는 경우에는 Http Archive(Http 아카이브)를 선택합니다.
7. 리포지토리 URL에 이전 단계에서 생성한 쿡북이 저장된 리포지토리를 입력합니다. 위의 예제를 사용하는 경우에는 **`https://s3.amazonaws.com/aws-cloudwatch/downloads/CloudWatchLogs-Cookbooks.zip`**를 입력합니다.
8. Add stack(스택 추가)을 선택해서 스택을 생성합니다.

3단계: IAM 역할 확대

인스턴스에 CloudWatch 로그를 사용하려면 AWS OpsWorks 인스턴스에서 사용하는 IAM 역할을 확장해야 합니다.

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택한 후 Create Policy를 선택합니다.

3. 정책 생성 페이지의 Create Your Own Policy(자체 정책 생성)에서 선택을 선택합니다. 사용자 지정 정책 생성에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2용 IAM 정책을 참조하십시오.
4. 정책 검토 페이지의 정책 이름에 정책 이름을 입력합니다.
5. 정책 문서에 다음 정책을 붙여넣습니다.

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

6. 정책 생성을 선택하세요.
7. 탐색 창에서 [Roles] 를 선택한 다음 콘텐츠 창에서 [Role Name] 으로 스택에서 사용하는 인스턴스 역할의 이름을 선택합니다. AWS OpsWorks 스택 설정에서 스택에서 사용되는 역할을 찾을 수 있습니다(기본값 `aws-opsworks-ec2-role`).

> **Note**
>
> 확인란이 아니라 사용자 이름을 선택합니다.

8. 권한 탭의 Managed Policies(관리형 정책)에서 정책 연결을 선택합니다.
9. 정책 연결 창의 테이블 헤더(필터 및 검색 옆에 있음)에서 정책 유형과 Customer Managed Policies(고객 관리형 정책)를 선택합니다.
10. 고객 관리형 정책에서 위에서 생성한 IAM 정책을 선택하고 정책 연결을 선택합니다.

사용자 및 정책에 대한 자세한 내용은 IAM 사용 설명서의 IAM 사용자 및 그룹과 IAM 정책 관리를 참조하세요.

4단계: 계층 추가

1. https://console.aws.amazon.com/opsworks/ 에서 AWS OpsWorks 콘솔을 엽니다.
2. 탐색 창에서 계층을 선택합니다.
3. 콘텐츠 창에서 계층을 선택하고 Add layer(계층 추가)를 선택합니다.
4. OpsWorks탭에서 레이어 유형으로 사용자 지정을 선택합니다.
5. 이름 및 Short name(짧은 이름) 필드에서 계층의 긴 이름과 짧은 이름을 입력한 다음 Add layer(계층 추가)를 선택합니다.
6. 레시피 탭의 사용자 지정 Chef 레시피 아래에는 라이프사이클 이벤트에 해당하는 설정, 구성, 배포, 배포 취소, 종료 등 여러 제목이 있습니다. AWS OpsWorks AWS OpsWorks 인스턴스 수명 주기의 주요 시점에서 이러한 이벤트를 트리거하여 관련 레시피를 실행합니다.

   > **Note**
   >
   > 위의 제목들이 보이지 않으면 Custom Chef Recipes(사용자 지정 Chef 레시피)로 가서 편집을 선택합니다.

7. 설정 옆에 logs::config, logs::install을 입력하고 +를 선택하여 목록에 추가한 다음 저장을 선택합니다.

   AWS OpsWorks 인스턴스가 부팅된 직후 이 계층의 새 인스턴스 각각에서 이 레시피를 실행합니다.

5단계: 인스턴스 추가

이 계층은 인스턴스를 구성하는 방법을 제어만 합니다. 해당 계층에 몇몇 인스턴스를 추가하고 이를 시작해야 합니다.

1. https://console.aws.amazon.com/opsworks/ 에서 AWS OpsWorks 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택한 다음 계층으로 가서 + 인스턴스를 선택합니다.
3. 기본 설정을 수락하고 Add Instance(인스턴스 추가)를 선택해서 해당 계층에 인스턴스를 추가합니다.

4. 해당 행의 작업 열에서 시작을 클릭해서 인스턴스를 시작합니다.

   AWS OpsWorks 새 EC2 인스턴스를 시작하고 로그를 구성합니다 CloudWatch . 준비가 되면 인스턴스가 온라인 상태로 바뀝니다.

6단계: 로그 보기

에이전트가 잠시 실행되면 CloudWatch 콘솔에서 새로 생성된 로그 그룹과 로그 스트림을 확인할 수 있습니다.

자세한 정보는 Logs로 전송된 로그 데이터 보기 CloudWatch 을 참조하세요.

CloudWatch 로그 에이전트 상태를 보고하세요.

다음 절차를 사용하여 EC2 인스턴스의 CloudWatch Logs 에이전트 상태를 보고하십시오.

에이전트 상태를 보고하려면

1. EC2 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하십시오.

   연결 문제에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 연결 문제 해결을 참조하십시오.
2. 명령 프롬프트에서 다음 명령을 입력합니다:

   ```
   sudo service awslogs status
   ```

   Amazon Linux 2를 실행 중인 경우 다음 명령을 입력합니다.

   ```
   sudo service awslogsd status
   ```
3. /var/log/awslogs.log 파일에서 CloudWatch 로그 에이전트와 관련된 오류, 경고 또는 문제를 확인하십시오.

CloudWatch 로그 에이전트를 시작합니다.

EC2 인스턴스의 CloudWatch Logs 에이전트가 설치 후 자동으로 시작되지 않거나 에이전트를 중지한 경우 다음 절차를 사용하여 에이전트를 시작할 수 있습니다.

에이전트를 시작하려면

1. EC2 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하십시오.

   연결 문제에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 연결 문제 해결을 참조하십시오.
2. 명령 프롬프트에서 다음 명령을 입력합니다:

   ```
   sudo service awslogs start
   ```

   Amazon Linux 2를 실행 중인 경우 다음 명령을 입력합니다.

   ```
   sudo service awslogsd start
   ```

CloudWatch 로그 에이전트를 중지하십시오.

다음 절차를 사용하여 EC2 인스턴스에서 CloudWatch Logs 에이전트를 중지하십시오.

에이전트를 중지하려면

1. EC2 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하십시오.

   연결 문제에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 연결 문제 해결을 참조하십시오.
2. 명령 프롬프트에서 다음 명령을 입력합니다:

   ```
   sudo service awslogs stop
   ```

   Amazon Linux 2를 실행 중인 경우 다음 명령을 입력합니다.

   ```
   sudo service awslogsd stop
   ```

빠른 시작: 로그를 시작하는 AWS CloudFormation CloudWatch 데 사용합니다.

AWS CloudFormation AWS 리소스를 JSON 형식으로 설명하고 프로비저닝할 수 있습니다. 이 방법의 장점은 리소스 컬렉션을 단일 단위로 관리할 수 있고 AWS AWS 리소스를 여러 지역에 쉽게 복제할 수 있다는 점입니다.

AWS 를 사용하여 프로비저닝할 AWS CloudFormation때는 사용할 AWS 리소스를 설명하는 템플릿을 생성합니다. 다음은 로그 그룹과 404 출현 횟수를 계산해서 로그 그룹에 전송하는 지표 필터를 생성하는 템플릿 코드 조각의 예제입니다.

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

다음은 기본적인 예제입니다. 를 사용하여 훨씬 더 풍부한 CloudWatch 로그 배포를 설정할 수 있습니다. AWS CloudFormation템플릿 예제에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 Amazon CloudWatch Logs 템플릿 스니펫을 참조하십시오. 시작하기에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation시작하기를 참조하세요.

SDK와 함께 CloudWatch 로그 사용 AWS

AWS 소프트웨어 개발 키트 (SDK) 는 널리 사용되는 여러 프로그래밍 언어에 사용할 수 있습니다. 각 SDK는 개발자가 선호하는 언어로 애플리케이션을 쉽게 구축할 수 있도록 하는 API, 코드 예시 및 설명서를 제공합니다.

SDK 설명서	코드 예시
AWS SDK for C++	AWS SDK for C++ 코드 예제
AWS CLI	AWS CLI 코드 예제
AWS SDK for Go	AWS SDK for Go 코드 예제
AWS SDK for Java	AWS SDK for Java 코드 예제
AWS SDK for JavaScript	AWS SDK for JavaScript 코드 예제
AWS SDK for Kotlin	AWS SDK for Kotlin 코드 예제
AWS SDK for .NET	AWS SDK for .NET 코드 예제
AWS SDK for PHP	AWS SDK for PHP 코드 예제
AWS Tools for PowerShell	PowerShell 코드 예제를 위한 도구
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) 코드 예제
AWS SDK for Ruby	AWS SDK for Ruby 코드 예제
AWS SDK for Rust	AWS SDK for Rust 코드 예제
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP 코드 예제
AWS SDK for Swift	AWS SDK for Swift 코드 예제

CloudWatch 로그와 관련된 예는 을 참조하십시오AWS SDK를 사용한 CloudWatch 로그의 코드 예제.

가용성 예제

필요한 예제를 찾을 수 없습니까? 이 페이지 하단의 피드백 제공 링크를 사용하여 코드 예시를 요청하세요.

로그 인사이트를 통한 CloudWatch 로그 데이터 분석

CloudWatch Logs Insights를 사용하면 Amazon CloudWatch Logs의 로그 데이터를 대화형 방식으로 검색하고 분석할 수 있습니다. 운영상의 문제에 보다 효율적이고 효과적으로 대처할 수 있도록 쿼리를 수행할 수 있습니다. 문제가 발생하는 경우 CloudWatch Logs Insights를 사용하여 잠재적 원인을 식별하고 배포된 수정 사항을 검증할 수 있습니다.

CloudWatch Logs Insights에는 몇 가지 간단하지만 강력한 명령을 포함하는 용도에 맞게 구축된 쿼리 언어가 포함되어 있습니다. CloudWatch Logs Insights는 시작하는 데 도움이 되는 샘플 쿼리, 명령 설명, 쿼리 자동 완성 및 로그 필드 검색을 제공합니다. 여러 가지 유형의 AWS 서비스 로그에 대한 샘플 쿼리가 포함되어 있습니다.

CloudWatch Logs Insights는 Amazon Route 53, AWS Lambda AWS CloudTrail, Amazon VPC와 같은 AWS 서비스의 로그와 로그 이벤트를 JSON으로 내보내는 모든 애플리케이션 또는 사용자 지정 로그의 필드를 자동으로 검색합니다.

CloudWatch 로그 인사이트를 사용하여 2018년 11월 5일 이후에 CloudWatch Logs로 전송된 로그 데이터를 검색할 수 있습니다.

> **Important**
>
> CloudWatch Logs Insights는 로그 그룹 생성 시간보다 오래된 타임스탬프가 있는 로그 이벤트에 액세스할 수 없습니다.

자연어를 사용하여 CloudWatch Logs Insights 쿼리를 만들 수도 있습니다. 그러려면 찾고 있는 데이터에 대해 질문하거나 설명합니다. 이 AI 지원 기능은 프롬프트를 기반으로 쿼리를 생성하고 쿼리 작동 방식에 line-by-line 대한 설명을 제공합니다. 자세한 내용은 자연어를 사용하여 CloudWatch Logs Insights 쿼리 생성 및 업데이트를 참조하십시오.

CloudWatch 교차 계정 Observability에서 모니터링 계정으로 설정된 계정에 로그인한 경우 이 모니터링 계정에 연결된 소스 계정의 CloudWatch 로그 그룹에서 Logs Insights 쿼리를 실행할 수 있습니다. 다른 계정에 있는 여러 로그 그룹을 쿼리하는 쿼리를 실행할 수 있습니다. 자세한 내용은 계정 CloudWatch 간 옵저버빌리티를 참조하세요.

하나의 요청으로 최대 50개의 로그 그룹을 쿼리할 수 있습니다. 쿼리가 완료되지 않은 경우 60분 후에 쿼리가 시간 초과됩니다. 쿼리 결과는 7일 동안 사용할 수 있습니다.

생성한 쿼리를 저장할 수 있습니다. 그러면 복잡한 쿼리를 실행해야 할 때마다 다시 만들지 않고도 실행할 수 있습니다.

CloudWatch Logs Insights 쿼리에는 쿼리된 데이터 양에 따라 요금이 부과됩니다. 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

> **Important**
>
> 네트워크 보안 팀에서 웹 소켓 사용을 허용하지 않는 경우, 현재 CloudWatch 콘솔의 CloudWatch Logs Insights 부분에 액세스할 수 없습니다. API를 사용하여 CloudWatch 로그 인사이트 쿼리 기능을 사용할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs API 레퍼런스를 참조하십시오 StartQuery.

내용

로그 클래스에서 지원되는 명령

모든 CloudWatch Logs Insights 쿼리 명령은 표준 로그 클래스의 로그 그룹에서 지원됩니다. Inrequent Access 로그 클래스의 로그 그룹은`pattern`,`diff`, 를 제외한 모든 쿼리 명령을 지원합니다. `unmask`

시작하기: 쿼리 자습서

다음 섹션에는 Logs Insights를 시작하는 CloudWatch 데 도움이 되는 샘플 쿼리 자습서가 포함되어 있습니다.

주제

자습서: 샘플 쿼리 실행 및 수정

다음 튜토리얼은 CloudWatch Logs Insights를 시작하는 데 도움이 됩니다. 샘플 쿼리를 실행한 다음 수정해 다시 실행하는 방법을 살펴봅니다.

쿼리를 실행하려면 로그에 CloudWatch 로그가 이미 저장되어 있어야 합니다. 이미 CloudWatch 로그를 사용하고 있고 로그 그룹과 로그 스트림을 설정했다면 시작할 준비가 된 것입니다. Amazon Route 53 또는 Amazon VPC와 AWS CloudTrail같은 서비스를 사용하고 로그로 이동하도록 CloudWatch 해당 서비스에서 로그를 설정한 경우에도 이미 로그가 있을 수 있습니다. 로그로 CloudWatch 로그를 보내는 방법에 대한 자세한 내용은 을 참조하십시오 CloudWatch 로그 시작하기.

CloudWatch Logs Insights의 쿼리는 로그 이벤트의 필드 집합이나 로그 이벤트에 대해 수행된 수학적 집계 또는 기타 작업의 결과를 반환합니다. 이 자습서에서는 로그 이벤트 목록을 반환하는 쿼리를 보여줍니다.

샘플 쿼리 실행

CloudWatch Logs Insights 샘플 쿼리를 실행하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.

   로그 인사이트 페이지의 쿼리 편집기에는 최신 로그 이벤트 20개를 반환하는 기본 쿼리가 포함되어 있습니다.
3. 로그 그룹 선택(Select log group(s)) 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.

CloudWatch 교차 계정 옵저버빌리티의 모니터링 계정인 경우 모니터링 계정뿐만 아니라 소스 계정에서도 로그 그룹을 선택할 수 있습니다. 하나의 쿼리로 다른 계정의 로그를 한 번에 쿼리할 수 있습니다.

로그 그룹 이름, 계정 ID 또는 계정 레이블을 기준으로 로그 그룹을 필터링할 수 있습니다.

표준 로그 클래스에서 로그 그룹을 선택하면 CloudWatch Logs Insights가 그룹의 데이터 필드를 자동으로 탐지합니다. 검색된 필드를 보려면 페이지 오른쪽 상단의 필드(Fields) 메뉴를 선택합니다.

> **Note**
>
> 검색된 필드는 표준 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

4. (선택 사항) 시간 간격 선택기를 사용하여 쿼리할 기간을 선택합니다.

   5~30분 간격, 1, 3, 12시간 간격 또는 사용자 지정 시간 범위 중에서 선택할 수 있습니다.

5. 실행(Run)을 선택하고 결과를 봅니다.

   이 자습서에서는 가장 최근에 추가된 로그 이벤트 20개가 결과에 포함되어 있습니다.

   CloudWatch 로그에는 시간 경과에 따른 로그 그룹 내 로그 이벤트의 막대 그래프가 표시됩니다. 이 막대 그래프는 테이블의 이벤트뿐만 아니라 쿼리 및 시간 범위와 일치하는 로그 그룹 내 이벤트의 분포를 보여줍니다.

6. 반환된 로그 이벤트의 모든 필드를 보려면 번호가 매겨진 이벤트의 왼쪽에 있는 삼각형 드롭다운 아이콘을 선택합니다.

샘플 쿼리 수정

이 자습서에서는 최신 로그 이벤트 50개를 표시하도록 샘플 쿼리를 수정합니다.

이전 자습서를 아직 실행하지 않은 경우 지금 실행하는 것이 좋습니다. 이 자습서에는 이전 자습서를 마친 지점에서 시작합니다.

> **Note**
>
> CloudWatch Logs Insights와 함께 제공되는 일부 샘플 쿼리는 대신 `head` 또는 `tail` 명령을 사용합니다`limit`. 이러한 명령은 더 이상 사용되지 않으며 `limit`로 대체되었습니다. 작성하는 모든 쿼리에 `head` 또는 `tail` 대신 `limit`를 사용합니다.

CloudWatch Logs Insights 샘플 쿼리를 수정하려면

1. 쿼리 편집기에서 20을 50으로 변경한 다음 실행을 선택합니다.

   새 쿼리의 결과가 표시됩니다. 기본 시간 범위 내에서 로그 파일에 데이터가 충분하다고 가정하고 이제 로그 이벤트 50개가 나열됩니다.
2. (선택 사항) 생성한 쿼리를 저장할 수 있습니다. 이 쿼리를 저장하려면 저장을 선택합니다. 자세한 정보는 Logs Insights 쿼리를 저장하고 다시 실행합니다 CloudWatch . 섹션을 참조하세요.

샘플 쿼리에 필터 명령 추가

이 자습서에서는 쿼리 편집기에서 쿼리를 보다 과감하게 변경하는 방법에 대해 살펴봅니다. 이 자습서에서는 검색된 로그 이벤트의 필드를 기반으로 이전 쿼리의 결과를 필터링합니다.

이전 자습서를 아직 실행하지 않은 경우 지금 실행하는 것이 좋습니다. 이 자습서에는 이전 자습서를 마친 지점에서 시작합니다.

이전 쿼리에 필터 명령을 추가하려면

1. 필터링할 필드를 결정합니다. 선택한 로그 그룹에 포함된 CloudWatch 로그 이벤트에서 지난 15분 동안 Logs가 탐지한 가장 일반적인 필드와 각 필드가 나타나는 해당 로그 이벤트의 비율을 보려면 페이지 오른쪽에 있는 필드를 선택합니다.

   특정 로그 이벤트에 포함된 필드를 확인하려면 해당 행 왼쪽에 있는 아이콘을 선택합니다.

   로그에 포함된 이벤트에 따라 로그 이벤트에 awsRegion 필드가 나타날 수 있습니다. 이 자습서의 나머지 부분에서는 필터 필드로 awsRegion을 사용하는데, 이 필드를 사용할 수 없는 경우에는 다른 필드를 사용할 수 있습니다.
2. 쿼리 편집기 상자에서 50 뒤에 커서를 놓고 Enter를 누릅니다.
3. 새 줄에 먼저 |(파이프 문자)와 공백을 입력합니다. CloudWatch Logs Insights 쿼리의 명령은 파이프 문자로 구분해야 합니다.

4. **`filter awsRegion="us-east-1"`**을 입력합니다.
5. Run(실행)을 선택합니다.

   다시 쿼리를 실행하면 이제 새 필터와 일치하는 최신 결과 50개가 표시됩니다.

   다른 필드를 필터링했는데 오류 결과가 표시되면 해당 필드 이름을 이스케이프해야 할 수 있습니다. 필드 이름에 영숫자가 아닌 문자가 포함되어 있으면 필드 이름의 앞/뒤에 백틱 문자(`)를 입력해야 합니다(예: **`` `error-code`="102" ``**).

   영숫자가 아닌 문자를 포함하는 필드 이름에 백틱 문자를 사용해야 하지만 값은 그렇지 않습니다. 값은 항상 따옴표(") 안에 포함됩니다.

CloudWatch Logs Insights에는 몇 가지 명령과 정규 표현식, 수학 및 통계 연산에 대한 지원을 비롯한 강력한 쿼리 기능이 포함되어 있습니다. 자세한 정보는 CloudWatch 로그 인사이트 쿼리 구문을 참조하세요.

자습서: 집계 함수를 사용하여 쿼리 실행

집계 함수를 `stats` 명령과 사용하고 다른 함수의 인수로 사용할 수 있습니다. 이 자습서에서는 지정된 필드를 포함하는 로그 이벤트의 수를 계산하는 쿼리 명령을 실행합니다. 쿼리 명령은 지정된 필드의 하나 이상의 값으로 그룹화된 총 개수를 반환합니다. 집계 함수에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 지원되는 작업 및 함수를 참조하십시오.

집계 함수를 사용하여 쿼리 실행

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 로그 그룹 선택(Select log group(s)) 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.

   CloudWatch 교차 계정 옵저버빌리티의 모니터링 계정인 경우 모니터링 계정뿐만 아니라 소스 계정에서도 로그 그룹을 선택할 수 있습니다. 하나의 쿼리로 다른 계정의 로그를 한 번에 쿼리할 수 있습니다.

   로그 그룹 이름, 계정 ID 또는 계정 레이블을 기준으로 로그 그룹을 필터링할 수 있습니다.

   로그 그룹을 선택하면 CloudWatch Logs Insights는 해당 로그 그룹이 표준 클래스 로그 그룹인 경우 해당 로그 그룹의 데이터 필드를 자동으로 탐지합니다. 검색된 필드를 보려면 페이지 오른쪽 상단의 필드(Fields) 메뉴를 선택합니다.
4. 쿼리 편집기에서 기본 쿼리를 삭제하고 다음 명령을 입력합니다.

```
stats count(*) by fieldName
```

5. *fieldName*을 필드(Fields) 메뉴의 검색된 필드로 바꿉니다.

   필드 메뉴는 페이지 오른쪽 상단에 있으며 CloudWatch Logs Insights가 로그 그룹에서 탐지한 모든 검색된 필드를 표시합니다.
6. 실행(Run)을 선택하고 쿼리 결과를 봅니다.

   쿼리 결과에는 쿼리 명령과 일치하는 로그 그룹의 레코드 수와 지정된 필드의 하나 이상의 값으로 그룹화된 총 개수가 표시됩니다.

자습서: 로그 필드로 그룹화된 시각화를 생성하는 쿼리 실행

`stats` 함수를 사용하여 반환된 결과를 로그 항목에 있는 하나 이상의 필드 값으로 그룹화하는 쿼리를 실행하면 결과를 막대 차트, 파이 차트, 선 그래프 또는 누적 영역 그래프로 볼 수 있습니다. 이렇게 하면 로그에서 추세를 보다 효율적으로 시각화할 수 있습니다.

시각화를 위한 쿼리를 실행하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 로그 그룹 선택(Select log group(s)) 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.

   CloudWatch 교차 계정 옵저버빌리티의 모니터링 계정인 경우 모니터링 계정뿐만 아니라 소스 계정에서도 로그 그룹을 선택할 수 있습니다. 하나의 쿼리로 다른 계정의 로그를 한 번에 쿼리할 수 있습니다.

   로그 그룹 이름, 계정 ID 또는 계정 레이블을 기준으로 로그 그룹을 필터링할 수 있습니다.
4. 쿼리 편집기에서 현재 내용을 삭제하고 다음 `stats` 함수를 입력한 후 쿼리 실행을 선택합니다.

   ```
   stats count(*) by @logStream
       | limit 100
   ```

   결과는 로그 그룹에서 각 로그 스트림에 대한 로그 이벤트 수를 보여줍니다. 결과는 100개 행으로 제한됩니다.
5. 시각화(Visualization) 탭을 선택합니다.
6. 선 옆에 있는 화살표를 선택한 다음 막대를 선택합니다.

막대 차트가 나타나고 로그 그룹의 각 로그 스트림에 대한 막대가 표시됩니다.

자습서: 시계열 시각화를 생성하는 쿼리 실행

`bin()` 함수를 사용해 기간별로 반환되는 결과를 그룹화하는 쿼리를 실행하는 경우 결과를 선 그래프, 누적 영역 그래프, 파이 차트 또는 막대 차트로 볼 수 있습니다. 이렇게 하면 로그 이벤트의 경시적 추세를 보다 효율적으로 시각화할 수 있습니다.

시각화를 위한 쿼리를 실행하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 로그 그룹 선택(Select log group(s)) 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.

   CloudWatch 교차 계정 옵저버빌리티의 모니터링 계정인 경우 모니터링 계정뿐만 아니라 소스 계정에서도 로그 그룹을 선택할 수 있습니다. 하나의 쿼리로 다른 계정의 로그를 한 번에 쿼리할 수 있습니다.

   로그 그룹 이름, 계정 ID 또는 계정 레이블을 기준으로 로그 그룹을 필터링할 수 있습니다.
4. 쿼리 편집기에서 현재 내용을 삭제하고 다음 `stats` 함수를 입력한 후 쿼리 실행을 선택합니다.

   ```
   stats count(*) by bin(30s)
   ```

   결과는 각 30초 기간 동안 로그가 수신한 로그 그룹 내 CloudWatch 로그 이벤트 수를 보여줍니다.
5. 시각화(Visualization) 탭을 선택합니다.

   결과가 선 그래프로 표시됩니다. 막대 차트, 파이 차트 또는 누적 영역 차트로 전환하려면 그래프 오른쪽 위 Line(선형) 옆에 있는 화살표를 선택합니다.

지원되는 로그 및 검색되는 필드

CloudWatch 로그 인사이트는 다양한 로그 유형을 지원합니다. CloudWatch Logs Insights는 표준 클래스 로그 그룹 Amazon Logs로 전송되는 모든 CloudWatch 로그에 대해 자동으로 다섯 개의 시스템 필드를 생성합니다.

- `@message`에는 구문 분석되지 않은 원시 로그 이벤트가 포함되어 있습니다. 이는 의 `message` 필드와 동일합니다 InputLogevent.
- `@timestamp`에는 로그 이벤트 `timestamp` 필드의 이벤트 타임스탬프가 포함되어 있습니다. 이 `timestamp` 필드는 의 필드와 동일합니다 InputLogevent.
- `@ingestionTime` CloudWatch Logs가 로그 이벤트를 수신한 시간을 포함합니다.
- `@logStream`에는 로그 이벤트가 추가된 로그 스트림의 이름이 포함되어 있습니다. 로그 스트림은 로그를 생성한 프로세스와 동일한 프로세스를 통해 로그를 그룹화합니다.
- `@log`는 형식의 로그 그룹 식별자입니다.*`account-id`*`:`*`log-group-name`* 이 필드는 여러 로그 그룹을 쿼리할 때 특정 이벤트가 속한 로그 그룹을 식별하는 데 유용할 수 있습니다.

> **Note**
>
> 필드 검색은 Standard 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

CloudWatch Logs Insights는 생성하는 필드의 시작 부분에 @ 기호를 삽입합니다.

또한 많은 로그 유형의 경우 CloudWatch Logs는 로그에 포함된 로그 필드를 자동으로 검색합니다. 다음 표에는 이러한 자동 검색 필드가 나와 있습니다.

CloudWatch Logs Insights가 자동으로 검색하지 않는 필드가 있는 다른 유형의 로그의 경우 `parse` 명령을 사용하여 해당 쿼리에 사용할 추출된 필드를 추출하고 만들 수 있습니다. 자세한 정보는 CloudWatch 로그 인사이트 쿼리 구문을 참조하세요.

검색된 로그 필드의 이름이 `@` 문자로 시작하는 경우 CloudWatch Logs Insights는 처음에 `@` 추가된 문자로 필드를 표시합니다. 예를 들어, 로그 필드 이름이 `@example.com`이면 이 필드 이름은 `@@example.com`으로 표시됩니다.

로그 유형	검색된 로그 필드
Amazon VPC 흐름 로그	`@timestamp` , `@logStream` , `@message`, `accountId` , `endTime`, `interfaceId` , `logStatus` , `startTime` , `version`, `action`, `bytes`, `dstAddr`, `dstPort`, `packets`, `protocol`, `srcAddr`, `srcPort`

로그 유형	검색된 로그 필드
Route 53 로그	`@timestamp` , `@logStream` , `@message`, `edgeLocation` , `ednsClientSubnet` , `hostZoneId` , `protocol`, `queryName` , `queryTimestamp` , `queryType` , `resolverIp` , `responseCode` , `version`
Lambda 로그	`@timestamp` , `@logStream` , `@message`, `@requestId` , `@duration`, `@billedDuration` , `@type`, `@maxMemoryUsed` , `@memorySize` Lambda 로그 행에 X-Ray 추적 ID가 포함된 경우 `@xrayTraceId` 및 `@xraySegmentId` 필드도 포함됩니다. CloudWatch Logs Insights는 Lambda 로그의 로그 필드를 자동으로 검색하지만, 각 로그 이벤트에 포함된 첫 번째 JSON 프래그먼트만 검색합니다. Lambda 로그 이벤트에 JSON 조각이 여러 개 포함된 경우 **`parse`** 명령을 사용하여 로그 필드를 구문 분석하고 추출할 수 있습니다. 자세한 정보는 JSON 로그의 필드을 참조하세요.
CloudTrail 로그 JSON 형식의 로그	자세한 정보는 JSON 로그의 필드 섹션을 참조하세요.
기타 로그 유형	`@timestamp` , `@ingestionTime` , `@logStream` , `@message`, `@log`.

JSON 로그의 필드

CloudWatch 로그 인사이트에서는 점 표기법을 사용하여 JSON 필드를 나타냅니다. 이 섹션에는 점 표기법을 사용하여 JSON 필드에 액세스하는 방법을 보여주는 JSON 이벤트 예제와 코드 조각이 포함되어 있습니다.

예제: JSON 이벤트

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

예제 JSON 이벤트에는 이름이 `userIdentity`로 지정된 객체가 포함되어 있고, `userIdentity`에는 이름이 `type`으로 지정된 필드가 포함되어 있습니다. 점 표기법을 사용하여 `type` 값을 표시하려면 `userIdentity.type`을 사용합니다.

예제 JSON 이벤트에는 중첩 필드 이름 및 값 목록으로 병합되는 배열이 포함되어 있습니다. `requestParameters.instancesSet`의 첫 번째 항목에 대해 `instanceId` 값을 표시하려면 `requestParameters.instancesSet.items.0.instanceId`를 사용합니다. 필드 `instanceID` 앞에 위치한 숫자 `0`은 필드 `items` 값의 위치를 나타냅니다. 다음 예제에는 JSON 로그 이벤트의 중첩된 JSON 필드에 액세스하는 방법을 보여주는 코드 조각이 포함되어 있습니다.

예제: 쿼리

```
fields @timestamp, @message
| filter requestParameters.instancesSet.items.0.instanceId="i-abcde123"
| sort @timestamp desc
```

중첩된 JSON 필드 `instanceId` 값에 액세스하기 위해 `filter` 명령과 함께 점 표기법을 사용하는 쿼리를 보여주는 코드 조각입니다. `instanceId` 값이 `"i-abcde123"`과 같은 메시지를 필터링하고 지정된 값을 포함하는 모든 로그 이벤트를 반환하는 쿼리입니다.

> **Note**
>
> CloudWatch 로그 인사이트는 JSON 로그에서 최대 200개의 로그 이벤트 필드를 추출할 수 있습니다. 추출되지 않은 추가 필드의 경우 `parse` 명령을 사용하여 메시지 필드의 구문 분석되지 않은 원시 로그 이벤트에서 이러한 필드를 추출할 수 있습니다. `parse`명령에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 쿼리 구문을 참조하십시오.

CloudWatch 로그 인사이트 쿼리 구문

CloudWatch Logs Insights를 사용하면 쿼리 언어를 사용하여 로그 그룹을 쿼리할 수 있습니다. 쿼리 구문은 일반 함수, 산술 및 비교 연산, 정규 표현식을 포함하되 이에 국한되지 않는 다양한 함수 및 연산을 지원합니다.

여러 명령이 포함된 쿼리를 생성하려면 파이프 문자(|)로 명령을 구분합니다.

설명이 포함된 쿼리를 생성하려면 설명을 해시 문자(#)로 설정합니다.

> **Note**
>
> CloudWatch Logs Insights는 다양한 로그 유형의 필드를 자동으로 검색하고 @ 문자로 시작하는 필드를 생성합니다. 이러한 필드에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 지원되는 로그 및 발견된 필드를 참조하십시오.

다음 표에는 각 명령에 대한 간략한 설명이 나와 있습니다. 다음 표에는 예제와 함께 각 명령에 대한 보다 포괄적인 설명이 나와 있습니다.

> **Note**
>
> 모든 CloudWatch Logs Insights 쿼리 명령은 표준 로그 클래스의 로그 그룹에서 지원됩니다. Inrequent Access 로그 클래스의 로그 그룹은`pattern`,`diff`, 를 제외한 모든 쿼리 명령을 지원합니다. `unmask`

display	쿼리 결과에 하나 이상의 특정 필드를 표시합니다.
fields	쿼리 결과에 특정 필드를 표시하고, 필드 값을 수정하고 쿼리에 사용할 새 필드를 생성하는 데 사용할 수 있는 함수와 연산을 지원합니다.
filter	하나 이상의 조건과 일치하는 로그 이벤트만 반환하도록 쿼리를 필터링합니다.
pattern	로그 데이터를 패턴으로 자동 클러스터링합니다. 패턴은 로그 필드 간에 반복되는 공유 텍스트 구조입니다. CloudWatch Logs Insights는 로그 이벤트에서 발견된 패턴을 분석할 수 있는 방법을 제공합니다. 자세한 정보는 패턴 분석을 참조하세요.
diff	요청한 기간에 발견된 로그 이벤트를 길이가 같은 이전 기간의 로그 이벤트와 비교하여 추세를 파악하고 특정 로그 이벤트가 새로운 것인지 확인할 수 있습니다.
parse	로그 필드에서 데이터를 추출하여 쿼리에서 처리할 수 있는 추출된 필드를 생성합니다. **parse**는 와일드카드를 사용하는 글로브 모드와 정규 표현식을 모두 지원합니다.
sort	반환된 로그 이벤트를 오름차순(asc) 또는 내림차순(desc)으로 표시합니다.
stats	로그 필드의 값을 사용하여 집계 통계를 계산합니다.
limit	쿼리에서 반환할 최대 로그 이벤트 수를 지정합니다. '상위 20개' 또는 '최근 20개' 결과를 반환하는 **sort**에 유용합니다.

dedup	지정한 필드의 특정 값을 기준으로 중복 결과를 제거합니다.
unmask	데이터 보호 정책으로 인해 일부 내용이 마스킹된 로그 이벤트의 모든 내용을 표시합니다. 로그 그룹의 데이터 보호에 대한 자세한 내용은 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원 섹션을 참조하세요.
기타 연산 및 함수	CloudWatch 또한 Logs Insights는 다양한 비교, 산술, 날짜/시간, 숫자, 문자열, IP 주소, 일반 함수 및 연산을 지원합니다.

다음 섹션에서는 Logs Insights 쿼리 명령에 대한 자세한 내용을 제공합니다. CloudWatch

주제

표시

`display` 명령을 사용하여 쿼리 결과에 하나 또는 여러 개의 특정 필드를 표시할 수 있습니다.

`display` 명령은 사용자가 지정한 필드만 표시합니다. 쿼리에 여러 `display` 명령이 포함된 경우, 쿼리 결과는 마지막 `display` 명령에서 지정한 필드만 표시합니다.

예제: 하나의 필드 표시

아래 코드 조각은 parse 명령을 사용하여 @message에서 데이터를 추출하여 추출된 필드 loggingType 및 loggingMessage를 생성하는 쿼리의 예를 보여줍니다. 쿼리는 loggingType에 대한 값이 ERROR인 모든 로그 이벤트를 반환합니다. display는 쿼리 결과에서 loggingMessage에 대한 값만 표시합니다.

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

> **Tip**
>
> 쿼리에서 display를 한 번만 사용하세요. 쿼리에서 display를 두 번 이상 사용할 경우, 쿼리 결과에는 마지막으로 사용된 display 명령에서 지정한 필드만 표시됩니다.

필드

fields를 사용하여 쿼리 결과에 특정 필드를 표시할 수 있습니다.

쿼리에 fields 명령이 여러 개 포함되어 있는데 display 명령은 포함되지 않은 경우 결과에 fields 명령에 지정된 모든 필드가 표시됩니다.

예제: 특정 필드 표시

다음 예제에서는 20개의 로그 이벤트를 반환하고 내림차순으로 표시하는 쿼리를 보여줍니다. 쿼리 결과에 @timestamp 및 @message에 대한 값이 표시됩니다.

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

필드 값을 수정하고 쿼리에 사용할 수 있는 새 필드를 생성하기 위해 fields에서 지원하는 다양한 함수와 연산을 사용하려는 경우 display 대신 fields를 사용하세요.

as 키워드와 함께 fields 명령을 사용하면 로그 이벤트에 있는 필드 및 함수를 사용하는 추출된 필드를 생성할 수 있습니다. 예를 들어, fields ispresent as isRes에서는 isRes라는 추출된 필드를 생성하며, 쿼리의 나머지 부분에서 추출된 필드를 사용할 수 있습니다.

필터

filter 명령을 사용하여 하나 이상의 조건과 일치하는 로그 이벤트를 가져올 수 있습니다.

예제: 하나의 조건을 사용한 로그 이벤트 필터링

아래 코드 조각은 range에 대한 값이 3000보다 큰 모든 로그 이벤트를 반환하는 쿼리의 예를 보여줍니다. 쿼리는 결과를 20개의 로그 이벤트로 제한하고 로그 이벤트를 내림차순의 @timestamp로 정렬합니다.

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

예제: 둘 이상의 조건을 사용한 로그 이벤트 필터링

키워드 and 및 or을 사용하여 둘 이상의 조건을 결합할 수 있습니다.

아래 코드 조각은 range에 대한 값이 3000보다 크고 accountId에 대한 값이 123456789012와 동일한 로그 이벤트를 반환하는 쿼리의 예를 보여줍니다. 쿼리는 결과를 20개의 로그 이벤트로 제한하고 로그 이벤트를 내림차순의 @timestamp로 정렬합니다.

```
fields @timestamp, @message
| filter (range>3000 and accountId=123456789012)
| sort @timestamp desc
| limit 20
```

필터 명령의 일치 항목 및 정규식

필터 명령은 정규 표현식 사용을 지원합니다. 비교 연산자(=, !=, <, <=, >, >=) 및 부울 연산자(and, or, not)를 사용할 수 있습니다.

in 키워드를 사용하여 설정된 멤버십을 테스트하고 배열의 요소를 확인할 수 있습니다. 배열에서 요소를 확인하려면 in 뒤에 해당 배열을 넣습니다. in과 함께 부울 연산자 not을 사용할 수 있습니다. in을 사용하여 필드가 문자열과 일치하는 로그 이벤트를 반환하는 쿼리를 생성할 수 있습니다. 필드는 완전한 문자열이어야 합니다. 예를 들어, 다음 코드 조각은 in을 사용하여 logGroup 필드가 완전한 문자열 example_group인 로그 이벤트를 반환하는 쿼리를 보여줍니다.

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

```
| filter logGroup in ["example_group"]
```

키워드 구문 like 및 not like를 사용하여 하위 문자열을 일치시킬 수 있습니다. 정규 표현식 연산자 =~를 사용하여 하위 문자열을 일치시킬 수 있습니다. like 및 not like로 하위 문자열을 일치시키려면, 일치해야 하는 하위 문자열을 큰따옴표 또는 작은따옴표로 둘러쌉니다. like 및 not like와 함께 정규식 패턴을 사용할 수 있습니다. 하위 문자열을 정규식 연산자와 일치시키려면 원하는 하위 문자열을 슬래시로 둘러쌉니다. 다음 예제에는 filter 명령을 사용하여 하위 문자열을 일치시키는 방법을 보여주는 코드 조각이 포함되어 있습니다.

예: 하위 문자열 일치

다음 예제에서는 f1에 단어 Exception이 포함된 모든 이벤트를 반환합니다. 세 예제 모두 대/소문자를 구별합니다.

첫 번째 예제에서는 like를 사용하여 하위 문자열과 일치시킵니다.

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

두 번째 예제에서는 like 및 정규 표현식 패턴을 사용하여 하위 문자열을 일치시킵니다.

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

세 번째 예제에서는 정규 표현식을 사용하여 하위 문자열을 일치시킵니다.

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

예: 와일드카드를 사용하여 하위 문자열 일치

정규 표현식에서 마침표 기호(.)를 와일드카드로 사용하여 하위 문자열을 일치시킬 수 있습니다. 다음 예에서 쿼리는 f1에 대한 값이 ServiceLog 문자열로 시작하는 일치 항목을 반환합니다.

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

마침표 기호(.*) 뒤에 별표 기호를 배치하여 가능한 한 많은 일치 항목을 반환하는 탐욕적 수량자(Greedy Quantifier)를 생성할 수 있습니다. 예를 들어, 다음 쿼리는 f1 값이 ServiceLog 문자열로 시작할 뿐만 아니라 ServiceLog 문자열도 포함하는 일치 항목을 반환합니다.

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

가능한 일치 항목은 다음과 같습니다.

- `ServiceLogSampleApiLogGroup`
- `SampleApiLogGroupServiceLog`

예제: 일치 항목에서 하위 문자열 제외

다음 예제에서는 `f1`에 단어 Exception이 포함되지 않은 모든 로그 이벤트를 반환하는 쿼리를 보여줍니다. 이 예제에서는 대/소문자를 구분합니다.

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

예제: 대/소문자를 구분하지 않는 패턴으로 하위 문자열 일치

대/소문자를 구분하지 않는 하위 문자열을 `like` 및 정규 표현식을 사용하여 일치시킬 수 있습니다. 일치시키려는 하위 문자열 앞에 다음 파라미터(?i)를 입력합니다. 다음 예제에서는 `f1`에 단어 Exception 또는 exception이 포함된 모든 로그 이벤트를 반환하는 쿼리를 보여줍니다.

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

패턴

`pattern`을 사용하여 로그 데이터를 패턴으로 자동 클러스터링합니다.

패턴은 로그 필드 간에 반복되는 공유 텍스트 구조입니다. `pattern`이를 사용하여 새로운 추세를 파악하고, 알려진 오류를 모니터링하고, 자주 발생하거나 비용이 많이 드는 로그 라인을 식별할 수 있습니다. CloudWatch 또한 Logs Insights는 로그 이벤트의 패턴을 찾고 추가로 분석하는 데 사용할 수 있는 콘솔 환경을 제공합니다. 자세한 정보는 패턴 분석을 참조하세요.

이 `pattern` 명령은 일반적인 패턴을 자동으로 식별하므로 이 명령을 시작 지점으로 사용하여 로그를 검색하고 분석할 수 있습니다. `filter`, `parse` 또는 `sort` 명령과 `pattern`을 결합하여 더 세밀하게 조정된 쿼리에서 패턴을 식별할 수도 있습니다.

패턴 명령 입력

pattern 명령에는 @message 필드, parse 명령을 사용하여 생성한 추출된 필드 또는 하나 이상의 문자열 함수를 사용하여 조작된 문자열과 같은 입력이 필요합니다.

패턴 명령 출력

pattern 명령으로 다음 출력이 생성됩니다.

- @pattern: 로그 이벤트 필드 간에 반복되는 공유 텍스트 구조입니다. 요청 ID 또는 타임스탬프와 같이 패턴 내에서 서로 다른 필드는 <*>로 표시됩니다. 예를 들어 [INFO] Request time: <*> ms는 로그 메시지 [INFO] Request time: 327 ms의 잠재적 출력입니다.
- @ratio: 선택한 기간 및 식별된 패턴과 일치하는 지정된 로그 그룹의 로그 이벤트 비율입니다. 예를 들어 선택한 로그 그룹 및 기간에 있는 로그 이벤트의 절반이 패턴과 일치할 경우 @ratio는 0.50을 반환합니다.
- @sampleCount: 선택한 기간 및 식별된 패턴과 일치하는 지정된 로그 그룹의 로그 이벤트의 수입니다.
- @severityLabel: 로그에 포함된 정보의 유형을 나타내는 로그 심각도 또는 수준입니다. 예: Error, Warning, Info, Debug 등.

예제

다음 명령은 선택한 시간 범위 동안 지정된 로그 그룹에서 구조가 비슷한 로그를 식별하여 패턴 및 개수별로 그룹화합니다.

```
pattern @message
```

pattern 명령은 filter 명령과 함께 사용할 수 있습니다.

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

pattern 명령은 parse 및 sort 명령과 함께 사용할 수 있습니다.

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

```
| sort @sampleCount asc
```

DIFF

요청된 기간에 발견된 로그 이벤트를 길이가 같은 이전 기간의 로그 이벤트와 비교합니다. 이렇게 하면 추세를 살펴보고 특정 로그 이벤트가 새로운 것인지 확인할 수 있습니다.

diff명령에 수정자를 추가하여 비교하려는 기간을 지정하십시오.

- diff현재 선택한 시간 범위의 로그 이벤트를 바로 이전 시간 범위의 로그 이벤트와 비교합니다.
- diff previousDay현재 선택한 시간 범위의 로그 이벤트를 전날 같은 시간의 로그 이벤트와 비교합니다.
- diff previousWeek현재 선택한 시간 범위의 로그 이벤트를 지난 주 같은 시간의 로그 이벤트와 비교합니다.
- diff previousMonth현재 선택한 시간 범위의 로그 이벤트를 전월 같은 시간의 로그 이벤트와 비교합니다.

자세한 정보는 이전 시간 범위와 비교 (diff)을 참조하세요.

parse

parse를 사용하여 로그 필드에서 데이터를 추출하고 쿼리에서 처리할 수 있는 추출된 필드를 생성합니다. **parse**는 와일드카드를 사용하는 글로브 모드와 정규 표현식을 모두 지원합니다. 정규 표현식 구문에 대한 자세한 내용은 을 참조하십시오. 지원되는 정규식 구문

정규 표현식을 사용하여 중첩된 JSON 필드를 구문 분석할 수 있습니다.

예: 중첩된 JSON 필드 구문 분석

아래 코드 조각은 수집 중에 평면화된 JSON 로그 이벤트를 구문 분석하는 방법을 보여줍니다.

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

아래 코드 조각은 추출된 필드 fld 및 array를 생성하기 위해 fieldsA 및 fieldsB에 대한 값을 추출하는 정규식이 포함된 쿼리를 보여줍니다.

```
parse @message "'fieldsA': '*', 'fieldsB': ['*']" as fld, array
```

이름이 지정된 캡처 그룹

정규 표현식과 **parse**을(를) 함께 사용하는 경우 이름이 지정된 캡처 그룹을 사용하여 패턴을 필드에 캡처할 수 있습니다. 구문은 `parse @message (?<Name>pattern).`입니다.

다음 예에서는 VPC 흐름 로그의 캡처 그룹을 사용하여 NetworkInterface(이)라는 이름의 필드에 ENI를 추출합니다.

```
parse @message /(?<NetworkInterface>eni-.*?) / display @timestamp, NetworkInterface
```

> **Note**
>
> JSON 로그 이벤트는 수집 중에 평면화됩니다. 현재 글로브 표현식을 사용하여 중첩된 JSON 필드를 파싱하는 것은 지원되지 않습니다. 200개 이하의 로그 이벤트 필드를 포함하는 JSON 로그 이벤트만 구문 분석할 수 있습니다. 중첩된 JSON 필드를 구문 분석할 때 JSON 로그 이벤트의 형식과 일치하도록 쿼리의 정규식 형식을 지정해야 합니다.

분석 명령의 예제

glob 표현식을 사용하여 로그 필드 **@message**에서 필드 **@user**, **@method** 및 **@latency**를 추출하고 **@method** 및 **@user**의 고유한 개별 조합에 대한 평균 지연 시간을 반환합니다.

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

정규식을 사용하여 로그 필드 **@message**에서 필드 **@user2**, **@method2** 및 **@latency2**를 추출하고 **@method2** 및 **@user2**의 고유한 개별 조합에 대한 평균 지연 시간을 반환합니다.

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

loggingTime, **loggingType** 및 **loggingMessage** 필드를 추출하고 **ERROR** 또는 **INFO** 문자열이 포함된 이벤트를 기록하도록 필터링한 다음 **ERROR** 문자열이 포함된 이벤트에 대해 **loggingMessage** 및 **loggingType** 필드만 표시합니다.

```
FIELDS @message
```

```
    | PARSE @message "* [*] *" as loggingTime, loggingType, loggingMessage
    | FILTER loggingType IN ["ERROR", "INFO"]
    | DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

정렬

sort를 사용하여 지정된 필드에 로그 이벤트를 오름차순(asc) 또는 내림차순(desc)으로 표시합니다. 이를 limit 명령과 함께 사용하여 '상위 N개' 또는 '하위 N개' 쿼리를 생성할 수 있습니다.

정렬 알고리즘은 자연 정렬의 업데이트된 버전입니다. 오름차순으로 정렬하는 경우 다음 논리가 사용됩니다.

- 숫자가 아닌 모든 값이 모든 숫자 값보다 앞에 옵니다. 숫자 값은 숫자만 포함하는 값이며 숫자와 다른 문자의 혼합은 포함하지 않습니다.
- 숫자가 아닌 값의 경우 알고리즘은 비교를 위해 연속된 숫자 문자와 연속된 영문자를 별도의 청크로 그룹화합니다. 숫자가 아닌 부분은 유니코드 값을 기준으로 정렬하고, 숫자 부분은 먼저 길이를 기준으로 정렬한 다음 숫자 값을 기준으로 정렬합니다.

유니코드 순서에 대한 자세한 내용은 유니코드 문자 목록을 참조하십시오.

예를 들어, 다음은 오름차순으로 정렬한 결과입니다.

```
!: >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> 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
```

내림차순으로 정렬하면 정렬 결과는 그 반대입니다.

예를 들어 Amazon VPC 흐름 로그에 대한 다음 쿼리는 호스트 전반의 상위 15개 패킷 전송을 찾습니다.

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

stats

stats를 사용하여 막대형 차트, 꺾은선형 차트, 누적 영역 차트와 같은 로그 데이터의 시각화를 생성합니다. 이렇게 하면 로그 데이터에서 패턴을 더 효율적으로 식별할 수 있습니다. CloudWatch Logs Insights는 해당 stats 함수와 하나 이상의 집계 함수를 사용하는 쿼리에 대한 시각화를 생성합니다.

예를 들어, Route 53 로그 그룹의 다음 쿼리는 시간당 Route 53 레코드의 분포를 쿼리 유형별로 보여주는 시각화를 반환합니다.

```
stats count(*) by queryType, bin(1h)
```

이러한 쿼리는 모두 막대 차트를 생성할 수 있습니다. 쿼리가 bin() 함수를 사용하여 필드별로 데이터를 그룹화하는 경우 선형 차트 및 누적 영역 차트도 볼 수 있습니다.

bin 함수에서는 다음과 같은 시간 단위 및 약어가 지원됩니다. 둘 이상의 문자를 포함하는 모든 단위 및 약어의 경우 s를 추가하여 복수화할 수 있습니다. 따라서 hr 및 hrs 모두 시간을 지정하는 데 사용됩니다.

- 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

주제

시계열 데이터 시각화

시계열 시각화는 다음과 같은 특성을 가진 쿼리에 사용할 수 있습니다.

- 쿼리에 집계 함수가 하나 이상 포함되어 있습니다. 자세한 정보는 Aggregation Functions in the Stats Command 섹션을 참조하세요.
- 쿼리가 `bin()` 함수를 사용하여 필드 하나를 기준으로 데이터를 그룹화합니다.

이러한 쿼리는 선형 차트, 누적 영역 차트, 막대 차트 및 파이 차트를 생성할 수 있습니다.

예제

전체 자습서는 the section called “자습서: 시계열 시각화를 생성하는 쿼리 실행” 섹션을 참조하세요.

다음은 시계열 시각화에 사용할 수 있는 쿼리 예제입니다.

다음 쿼리는 5분마다 생성된 데이터 포인트와 함께 `myfield1` 필드의 평균 값에 대한 시각화를 생성합니다. 각 데이터 포인트는 이전 5분 동안 생성된 로그의 `myfield1` 값 평균을 집계한 것입니다.

```
stats avg(myfield1) by bin(5m)
```

다음 쿼리는 5분마다 생성된 데이터 포인트와 함께 여러 필드를 기준으로 세 값의 시각화를 생성합니다. 쿼리가 집계 함수를 포함하고 있고 그룹화 필드로 `bin()`을 사용하기 때문에 시각화가 생성됩니다.

```
stats avg(myfield1), min(myfield2), max(myfield3) by bin(5m)
```

선형 차트 및 누적 영역 차트 제한 사항

로그 항목 정보를 집계하지만 `bin()` 함수를 사용하지 않는 쿼리는 막대 차트를 생성할 수 있습니다. 그러나 이러한 쿼리는 선형 차트 또는 누적 영역 차트를 생성할 수 없습니다. 이러한 쿼리 유형에 대한 자세한 내용은 the section called “필드별로 그룹화된 로그 데이터 시각화” 섹션을 참조하세요.

필드별로 그룹화된 로그 데이터 시각화

stats 함수와 하나 이상의 집계 함수를 사용하는 쿼리에 대해 막대 차트를 생성할 수 있습니다. 자세한 정보는 Aggregation Functions in the Stats Command 섹션을 참조하세요.

시각화를 보려면 쿼리를 실행합니다. 그런 다음 Visualization(시각화) 탭을 선택하고 Line(선형) 옆의 화살표를 선택한 다음 Bar(막대)를 선택합니다. 막대 차트에서는 시각화가 최대 100개 막대로 제한됩니다.

예제

전체 자습서는 the section called “자습서: 로그 필드로 그룹화된 시각화를 생성하는 쿼리 실행” 섹션을 참조하세요. 다음 단락에는 필드별 시각화에 대한 더 많은 예제 쿼리가 포함되어 있습니다.

다음 VPC 흐름 로그 쿼리는 각 대상 주소에 대해 세션당 전송된 평균 바이트 수를 찾습니다.

```
stats avg(bytes) by dstAddr
```

각 결과 값에 대해 둘 이상의 막대가 포함된 차트를 생성할 수도 있습니다. 예를 들어 다음 VPC 흐름 로그 쿼리는 각 대상 주소에 대해 세션당 전송된 평균 및 최대 바이트 수를 찾습니다.

```
stats avg(bytes), max(bytes) by dstAddr
```

다음 쿼리는 쿼리 유형에 대해 Amazon Route 53 쿼리 로그 수를 찾습니다.

```
stats count(*) by queryType
```

단일 쿼리에서 여러 통계 명령 사용

단일 쿼리에 최대 두 개의 stats 명령을 사용할 수 있습니다. 이렇게 하면 첫 번째 집계의 출력에 대해 추가 집계를 수행할 수 있습니다.

예: 두 개의 **stats** 명령을 사용한 쿼리

예를 들어, 다음 쿼리는 먼저 5분 빈(bin)의 총 트래픽 볼륨을 찾아낸 다음 해당 5분 빈 중 최고, 최저, 평균 트래픽 볼륨을 계산합니다.

```
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
```

예: 여러 통계 명령어를 다른 함수(**filter**, **fields**, **bin**)와 결합

단일 쿼리에서 두 개의 stats 명령을 다른 명령(filter 및 fields)과 결합할 수 있습니다. 예를 들어 다음 쿼리는 세션에서 고유한 IP 주소 수를 찾고 클라이언트 플랫폼별로 세션 수를 찾아내고, 해당 IP 주소를 필터링한 다음, 마지막으로 클라이언트 플랫폼별 평균 세션 요청을 찾아냅니다.

```
STATS count_distinct(client_ip) AS session_ips,
      count(*) AS requests BY session_id, client_platform
| FILTER session_ips > 1
| STATS count(*) AS multiple_ip_sessions,
        sum(requests) / count(*) AS avg_session_requests BY client_platform
```

여러 stats 명령을 사용하여 쿼리에서 bin 및 dateceil 함수를 사용할 수 있습니다. 예를 들어 다음 쿼리는 먼저 메시지를 5분 블록으로 결합한 다음 5분 블록을 10분 블록으로 집계하고 각 10분 블록 내의 최고, 최저 및 평균 트래픽 볼륨을 계산합니다.

```
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)
```

참고 및 제한 사항

쿼리당 최대 두 개의 stats 명령을 사용할 수 있습니다. 이 할당량은 변경할 수 없습니다.

sort 또는 limit 명령을 사용하는 경우 이 명령은 두 번째 stats 명령 뒤에 나타나야 합니다. 두 번째 stats 명령 앞에 있는 경우 쿼리는 유효하지 않습니다.

쿼리에 두 개의 stats 명령이 있는 경우 첫 번째 stats 집계가 완료될 때까지 쿼리의 일부 결과가 표시되지 않습니다.

단일 쿼리의 두 번째 stats 명령에서는 첫 번째 stats 명령에 정의된 필드만 참조할 수 있습니다. 예를 들어 첫 번째 stats 집계 후에는 @message 필드를 사용할 수 없으므로 다음 쿼리는 유효하지 않습니다.

```
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
```

첫 번째 stats 명령 이후에 참조하는 모든 필드는 첫 번째 stats 명령에서 정의해야 합니다.

```
STATS sum(x) as sum_x by y, z
| STATS max(sum_x) as max_x by z
# You can only reference `max(sum_x)`, max_x or z at this point
```

Important

bin 함수는 항상 @timestamp 필드를 묵시적으로 사용합니다. 즉, 첫 번째 stats 명령을 사용하여 timestamp 필드를 전파하지 않으면 두 번째 stats 명령에서 bin 함수를 사용할 수 없습니다. 예를 들어 다음 쿼리는 유효하지 않습니다.

```
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
```

대신 첫 번째 stats 명령에서 @timestamp 필드를 정의하면 다음 예제와 같이 두 번째 stats 명령에서 dateceil과 함께 필드를 사용할 수 있습니다.

```
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)
```

stats와 함께 사용할 함수

CloudWatch Logs Insights는 통계 집계 함수와 통계 비집계 함수를 모두 지원합니다.

statsaggregation 함수를 stats 명령에서 사용하고 다른 함수의 인수로 사용합니다.

함수	결과 유형	설명
avg(fieldName: NumericLogField)	number	지정된 필드의 값 평균입니다.

함수	결과 유형	설명

`count()`
 	
`count(fieldName: LogField)`
	number	로그 이벤트를 계산합니다. `count()`(또는`count(*)`)는 쿼리에서 반환하는 이벤트 수를 모두 세고 `count(fieldName)` 은 지정한 필드 이름이 포함된 레코드 수를 모두 계산합니다.

`count_distinct(fieldName: LogField)`	number	필드에 대해 고유한 값의 개수를 반환합니다. 필드의 카디널리티가 매우 높은 경우(고유한 값이 많이 포함되어 있음) `count_distinct` 가 반환하는 값은 근사치입니다.
`max(fieldName: LogField)`	LogFieldValue	쿼리된 로그에서 이 로그 필드에 대한 최댓값입니다.
`min(fieldName: LogField)`	LogFieldValue	쿼리된 로그에서 이 로그 필드에 대한 최솟값입니다.
`pct(fieldName: LogFieldValue, percent: number)`	LogFieldValue	백분위수는 데이터 세트에서 값의 상대적 위치를 나타냅니다. 예를 들어, `pct(@duration, 95)`는 `@duration` 의 값 중 95퍼센트가 이 값보다 낮고 5퍼센트는 이 값보다 큰 `@duration` 값을 반환합니다.
`stddev(fieldName: NumericLogField)`	number	지정된 필드의 값에 대한 표준 편차입니다.
`sum(fieldName: NumericLogField)`	number	지정된 필드의 값 합계입니다.

통계 비집계 함수

비집계 함수를 `stats` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

함수	결과 유형	설명
earliest(fieldName: LogField)	LogField	쿼리된 로그에서 가장 이른 타임스탬프가 있는 로그 이벤트에서 fieldName 의 값을 반환합니다.
latest(fieldName: LogField)	LogField	쿼리된 로그에서 최신 타임스탬프가 있는 로그 이벤트에서 fieldName 의 값을 반환합니다.
sortsFirst(fieldName: LogField)	LogField	쿼리된 로그에서 가장 빨리 정렬된 fieldName 의 값을 반환합니다.
sortsLast(fieldName: LogField)	LogField	쿼리된 로그에서 가장 늦게 정렬된 fieldName 의 값을 반환합니다.

제한

limit을 사용하여 쿼리에서 반환할 로그 이벤트 수를 지정할 수 있습니다.

예를 들어 다음 예제에서는 가장 최근의 로그 이벤트 25개만 반환합니다.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

dedup

dedup을 사용하여 지정한 필드의 특정 값을 기준으로 중복 결과를 제거합니다. 하나 이상의 필드와 함께 dedup를 사용할 수 있습니다. dedup을 사용하여 하나의 필드를 지정하면 해당 필드의 각 고유 값에 대해 하나의 로그 이벤트만 반환됩니다. 여러 필드를 지정하면 해당 필드의 고유한 값 조합마다 하나의 로그 이벤트가 반환됩니다.

중복 항목은 정렬 순서에 따라 삭제되며 정렬 순서의 첫 번째 결과만 유지됩니다. dedup 명령을 실행하기 전에 결과를 정렬하는 것이 좋습니다. dedup을 실행하기 전에 결과가 정렬되지 않으면 @timestamp를 사용하는 기본 내림차순 정렬 순서가 사용됩니다.

Null 값은 평가를 위해 중복으로 간주되지 않습니다. 지정된 필드에 대해 null 값이 있는 로그 이벤트는 유지됩니다. Null 값이 있는 필드를 제거하려면 isPresent(field) 함수를 사용하여 **filter**를 사용하세요.

dedup 명령 이후 쿼리에서 사용할 수 있는 유일한 쿼리 명령은 limit입니다.

예제: **server**라는 필드의 각 고유 값에 대한 가장 최근 로그 이벤트만 보기

다음 예제에서는 server의 각 고유 값에 대해 가장 최근 이벤트에 대한 timestamp, server, severity 및 message 필드를 표시합니다.

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server
```

CloudWatch Logs Insights 쿼리의 더 많은 샘플은 을 참조하십시오. 일반 쿼리

unmask

unmask를 사용하여 데이터 보호 정책으로 인해 일부 내용이 마스킹된 로그 이벤트의 모든 내용을 표시할 수 있습니다. 이 명령을 사용하려면 logs:Unmask 권한이 있어야 합니다.

로그 그룹의 데이터 보호에 대한 자세한 내용은 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원 섹션을 참조하세요.

부울, 비교, 숫자, 날짜/시간 및 기타 함수

CloudWatch Logs Insights는 다음 섹션에 설명된 대로 쿼리에서 다른 많은 작업과 함수를 지원합니다.

주제

- 산술 연산자
- 부울 연산
- 비교 연산자
- 숫자 연산자
- 날짜/시간 함수
- 일반 함수
- IP 주소 문자열 함수
- 문자열 함수

산술 연산자

산술 연산자에서는 숫자 데이터 형식을 인수로 수락하고 숫자 결과를 반환합니다. 산술 연산자를 filter 및 fields 명령에서 사용하고 다른 함수의 인수로 사용합니다.

Operation	설명
`a + b`	Addition
`a - b`	뺄셈
`a * b`	곱셈
`a / b`	나눗셈
`a ^ b`	거듭제곱(`2 ^ 3`에서 8 반환)
`a % b`	나머지 또는 모듈러스(`10 % 3`에서 1 반환)

부울 연산

부울 연산자 **`and`**, **`or`** 및 **`not`**을 사용합니다.

> **Note**
>
> TRUE 또는 FALSE의 값을 반환하는 함수에서만 부울 연산자를 사용합니다.

비교 연산자

비교 연산자에서는 모든 데이터 형식을 인수로 수락하고 부울 결과를 반환합니다. 비교 연산은 `filter` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

연산자	설명
`=`	같음
`!=`	같지 않음
`<`	보다 작음
`>`	보다 큼
`<=`	작거나 같음

연산자	설명
>=	크거나 같음

숫자 연산자

숫자 연산은 숫자 데이터 형식을 인수로 수락하고 숫자 결과를 반환합니다. 숫자 연산은 `filter` 및 `fields` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

Operation	결과 유형	설명
`abs(a: number)`	number	절대값
`ceil(a: number)`	number	천장값으로 반올림(a의 값보다 큰 수 중 가장 작은 정수)
`floor(a: number)`	number	바닥값으로 반올림(a 값보다 작은 수 중 가장 큰 정수)
`greatest(a: number, ...numbers: number[])`	number	가장 큰 값 반환
`least(a: number, ...numbers: number[])`	number	가장 작은 값 반환
`log(a: number)`	number	자연 로그
`sqrt(a: number)`	number	제곱근

날짜/시간 함수

날짜/시간 함수

날짜/시간 함수를 `fields` 및 `filter` 명령에서 사용하고 다른 함수의 인수로 사용합니다. 이러한 함수를 사용하여 집계 함수가 포함된 쿼리에 대한 시간 버킷을 생성합니다. 숫자와 다음 중 하나로 구성된 기간을 사용하세요.

- ms밀리초 동안
- s몇 초 동안
- m몇 분 동안
- h몇 시간 동안

예를 들어, 10m은 10분을, 1h는 1시간을 나타냅니다.

> **Note**
>
> datetime 함수에 가장 적합한 시간 단위를 사용하십시오. CloudWatch 로그는 선택한 시간 단위에 따라 요청을 제한합니다. 예를 들어, 사용하는 모든 요청의 최대값은 60으로 제한됩니다s. bin(300s)따라서 지정하면 CloudWatch Logs는 실제로 이 값을 60초로 구현합니다. 60은 분당 초 수이므로 CloudWatch Logs는 60초보다 큰 숫자를 사용하지 않습니다. s 5분 버킷을 만들려면 bin(5m) 대신 를 사용하십시오.
> 의 ms 상한선은 1000이고, s 과 의 m 상한선은 60이며, 상한선은 h 24입니다.

다음 표에는 쿼리 명령에 사용할 수 있는 다양한 날짜 시간 함수의 목록이 포함되어 있습니다. 이 표에는 각 함수의 결과 유형이 나열되며 각 함수에 대한 설명이 포함되어 있습니다.

> **Tip**
>
> 쿼리 명령을 생성할 때 시간 간격 선택기를 사용하여 쿼리할 기간을 선택할 수 있습니다. 예를 들어, 5~30분 간격, 1, 3, 12시간 간격 또는 사용자 지정 시간 범위 중에서 설정할 수 있습니다. 특정 날짜 사이의 기간을 설정할 수도 있습니다.

함수	결과 유형	설명
bin(period: Period)	Timestamp	@timestamp 값을 지정한 기간으로 반올림한 다음 자릅니다. 예를 들어, bin(5m)은(는) @timestamp 값을 가장 가까운 5분 단위로 반올림합니다. 이를 사용하여 쿼리에서 여러 로그 항목을 함께 그룹화할 수 있습니다. 다음 예제에서는 시간당 발생한 예외 수를 반환합니다.

함수	결과 유형	설명
		`filter @message like /Exception/	stats count(*) as exceptionCount by bin(1h)	sort exceptionCount desc` bin 함수에서는 다음과 같은 시간 단위 및 약어가 지원됩니다. 둘 이상의 문자를 포함하는 모든 단위 및 약어의 경우 s를 추가하여 복수화할 수 있습니다. 따라서 hr 및 hrs 모두 시간을 지정하는 데 사용됩니다. • 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	타임스탬프를 지정한 기간으로 자릅니다. 예를 들어, datefloor(@timestamp, 1h) 는 @timestamp 의 모든 값을 해당 시간 아래로 자릅니다.
dateceil(timestamp : Timestamp, period: Period)	Timestamp	타임스탬프를 지정한 기간으로 반올림한 다음 자릅니다. 예를 들어, dateceil(@timestamp, 1h) 는 @timestamp 의 모든 값을 해당 시간 위로 자릅니다.
fromMillis(fieldName: number)	Timestamp	입력 필드를 Unix Epoch 밀리초로 해석하여 타임스탬프로 변환합니다.

함수	결과 유형	설명
`toMillis(fieldName: Timestamp)`	number	지정된 필드에 있는 타임스탬프를 Unix Epoch 밀리초를 나타내는 숫자로 변환합니다. 예를 들어 `toMillis(@timestamp)` 는 타임스탬프를 `2022-01-14T13:18:031.000-08:00` 에서 `1642195111000` 으로 변환합니다.

> **Note**
>
> 현재 CloudWatch Logs Insights는 사람이 읽을 수 있는 타임스탬프가 포함된 로그 필터링을 지원하지 않습니다.

일반 함수

일반 함수

일반 함수를 `fields` 및 `filter` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

함수	결과 유형	설명
`ispresent(fieldName: LogField)`	불	이 필드가 존재하는 경우 `true` 반환
`coalesce(fieldName: LogField, ...fieldNames: LogField[])`	LogField	목록에서 null이 아닌 첫 번째 값 반환

IP 주소 문자열 함수

IP 주소 문자열 함수

IP 주소 문자열 함수를 `filter` 및 `fields` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

함수	결과 유형	설명
`isValidIp(fieldName: string)`	boolean	필드가 유효한 IPv4 또는 IPv6 주소인 경우 `true`를 반환합니다.
`isValidIpV4(fieldName: string)`	boolean	필드가 유효한 IPv4 주소인 경우 `true`를 반환합니다.
`isValidIpV6(fieldName: string)`	boolean	필드가 유효한 IPv6 주소인 경우 `true`를 반환합니다.
`isIpInSubnet(fieldName: string, subnet: string)`	boolean	필드가 지정된 v4 또는 v6 서브넷 내 유효한 IPv4 또는 IPv6 주소인 경우 `true`를 반환합니다. 서브넷을 지정할 때는 `192.0.2.0/24` 또는 `2001:db8::/32` 와 같은 CIDR 표기법을 사용합니다. 여기서는 `192.0.2.0` 또는`2001:db8::` 이 CIDR 블록의 시작입니다.
`isIpv4InSubnet(fieldName: string, subnet: string)`	boolean	필드가 지정된 v4 서브넷 내 유효한 IPv4 주소인 경우 `true`를 반환합니다. 서브넷을 지정할 때는 `192.0.2.0/24` 와 같은 CIDR 표기법을 사용합니다. 여기서는 `192.0.2.0` 이 CIDR 블록의 시작입니다.
`isIpv6InSubnet(fieldName: string, subnet: string)`	boolean	필드가 지정된 v6 서브넷 내 유효한 IPv6 주소인 경우 `true`를 반환합니다. 서브넷을 지정할 때는 `2001:db8::/32` 와 같은 CIDR 표기법을 사용합니다. 여기서는 `2001:db8::` 이 CIDR 블록의 시작입니다.

문자열 함수

문자열 함수

문자열 함수를 `fields` 및 `filter` 명령에서 사용하고 다른 함수의 인수로 사용합니다.

함수	결과 유형	설명
`isempty(fieldName: string)`	숫자	필드가 누락되어 있거나 빈 문자열일 경우 1를 반환합니다.
`isblank(fieldName: string)`	숫자	필드가 누락되어 있거나 빈 문자열이거나 빈 공백만 포함된 경우 1를 반환합니다.
`concat(str: string, ...strings: string[])`	문자열	문자열을 연결합니다.

`ltrim(str: string)`
 	
`ltrim(str: string, trimChars: string)`
	문자열	함수에 두 번째 인수가 없는 경우에는 문자열의 왼쪽에서 공백을 제거합니다. 함수에 두 번째 문자열 인수가 없는 경우에는 공백을 제거하지 않습니다. 대신 `str`의 왼쪽에서 `trimChars` 의 문자를 제거합니다. 예를 들어, `ltrim("xyZxyfooxyZ","xyZ")` 는 `"fooxyZ"`를 반환합니다.

`rtrim(str: string)`
 	
`rtrim(str: string, trimChars: string)`
	문자열	함수에 두 번째 인수가 없는 경우에는 문자열의 오른쪽에서 공백을 제거합니다. 함수에 두 번째 문자열 인수가 없는 경우에는 공백을 제거하지 않습니다. 대신 `str`의 오른쪽에서 `trimChars` 의 문자를 제거합니다. 예를 들어, `rtrim("xyZfooxyxyZ","xyZ")` 는 `"xyZfoo"`를 반환합니다.

`trim(str: string)`
 	
`trim(str: string, trimChars: string)`
	문자열	함수에 두 번째 인수가 없는 경우에는 문자열의 양쪽 끝에서 공백을 제거합니다. 함수에 두 번째 문자열 인수가 없는

함수	결과 유형	설명
		경우에는 공백을 제거하지 않습니다. 대신 str의 양쪽에서 trimChars 의 문자를 제거합니다. 예를 들어, trim("xyZxyfooxyxyZ","xyZ") 는 "foo"를 반환합니다.
strlen(str: string)	number	문자열 길이를 Unicode 코드 포인트로 반환합니다.
toupper(str: string)	문자열	문자열을 대문자로 변환합니다.
tolower(str: string)	문자열	문자열을 소문자로 변환합니다.
substr(str: string, startIndex: number) substr(str: string, startIndex: number, length: number)	문자열	숫자 인수가 지정한 인덱스의 하위 문자열을 문자열 끝에 반환합니다. 함수에 두 번째 숫자 인수가 있는 경우 해당 인수에는 검색되는 하위 문자열의 길이가 포함됩니다. 예를 들어, substr("xyZfooxyZ",3, 3) 는 "foo"를 반환합니다.

함수	결과 유형	설명
replace(fieldName: string, searchValue: string, replaceValue: string)	문자열	fieldName: string 에서 searchValue 의 모든 인스턴스를 replaceValue 로 바꿉니다. 예를 들어, replace(logGroup,"smoke_test","Smoke") 함수는 logGroup 필드에 문자열 값 smoke_test 를 포함한 로그 이벤트를 검색하고 해당 값을 Smoke 문자열로 바꿉니다.
strcontains(str: string, searchValue: string)	number	str에 searchValue 와 00이 포함되어 있으면 1을 반환합니다.

특수 문자가 포함된 필드

필드에 @ 기호나 마침표 () 이외의 영숫자가 아닌 문자가 포함된 경우 필드를 백틱 문자 (.) 로 둘러싸야 합니다. ` 예를 들어, foo-bar 로그 필드는 영숫자가 아닌 문자 하이픈(-)을 포함하고 있기 때문에 백틱(`foo-bar`)으로 묶어야 합니다.

쿼리에 별칭 및 설명 사용

별칭이 있는 쿼리를 생성합니다. 별칭은 로그 필드의 이름을 바꾸거나 값을 필드로 추출할 때 사용합니다. as라는 키워드를 사용하여 로그 필드 또는 결과에 별칭을 제공합니다. 쿼리에서 별칭을 2개 이상 사용할 수 있습니다. 다음과 같은 명령에서 별칭을 사용할 수 있습니다.

- fields
- parse
- sort
- stats

다음 예제에서는 별칭이 있는 쿼리를 생성하는 방법을 보여줍니다.

예

쿼리의 명령 `fields`에 별칭이 포함되어 있습니다.

```
fields @timestamp, @message, accountId as ID
| sort @timestamp desc
| limit 20
```

쿼리를 통해 `@timestamp`, `@message` 및 `accountId` 필드에 대한 값이 반환됩니다. 결과는 내림차순으로 정렬되며 20개로 제한됩니다. `accountId`에 대한 값이 `ID`라는 별칭 아래에 나열됩니다.

예

쿼리의 명령 `sort` 및 `stats`에 별칭이 포함되어 있습니다.

```
stats count(*) by duration as time
| sort time desc
```

쿼리에서는 로그 그룹에서 발생하는 `duration` 필드의 횟수를 계산하고 결과를 내림차순으로 정렬합니다. `duration`에 대한 값이 `time`라는 별칭 아래에 나열됩니다.

설명 사용

CloudWatch Logs Insights는 쿼리의 댓글을 지원합니다. 해시 문자(#)를 사용하여 설명을 시작합니다. 설명을 사용하여 쿼리 또는 문서 쿼리의 줄을 무시할 수 있습니다.

예제: 쿼리

다음 쿼리가 실행되면 두 번째 줄이 무시됩니다.

```
fields @timestamp, @message, accountId
# | filter accountId not like "7983124201998"
| sort @timestamp desc
| limit 20
```

패턴 분석

CloudWatch Logs Insights는 머신 러닝 알고리즘을 사용하여 로그를 쿼리할 때 패턴을 찾습니다. 패턴은 로그 필드 간에 반복되는 공유 텍스트 구조입니다. 쿼리 결과를 볼 때 패턴 탭을 선택하여 결과 샘플

을 기반으로 CloudWatch 로그가 찾은 패턴을 확인할 수 있습니다. 또는 쿼리에 pattern 명령을 추가하여 일치하는 전체 로그 이벤트 세트의 패턴을 분석할 수 있습니다.

패턴은 많은 수의 로그 이벤트가 종종 몇 가지 패턴으로 압축될 수 있기 때문에 대규모 로그 세트를 분석하는 데 유용합니다.

세 가지 로그 이벤트의 다음 예제를 고려해 보십시오.

```
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
```

이전 샘플에서는 세 개의 로그 이벤트가 모두 한 가지 패턴을 따릅니다.

```
<*> <*> [INFO] Calling DynamoDB to store for resource id <*>
```

패턴 내의 필드를 토큰이라고 합니다. 요청 ID 또는 타임스탬프와 같이 패턴 내에서 달라지는 필드는 동적 토큰입니다. 각 동적 토큰은 CloudWatch 로그에 <*> 표시되는 시점으로 표시됩니다.

동적 토큰의 일반적인 예로는 오류 코드, 타임스탬프, 요청 ID 등이 있습니다. 토큰 값은 동적 토큰의 특정 값을 나타냅니다. 예를 들어 동적 토큰이 HTTP 오류 코드를 나타내는 경우 토큰 값은 다음과 같을 수 501 있습니다.

패턴 감지는 CloudWatch 로그 이상 탐지기 및 비교 기능에서도 사용됩니다. 자세한 내용은 로그 이상 탐지 및 이전 시간 범위와 비교 (diff) 섹션을 참조하세요.

패턴 분석 시작하기

패턴 감지는 모든 CloudWatch Logs Insights 쿼리에서 자동으로 수행됩니다. pattern명령이 포함되지 않은 쿼리는 결과에 로그 이벤트와 패턴을 모두 가져옵니다.

쿼리에 pattern 명령을 포함하면 일치하는 전체 로그 이벤트 세트에 대해 패턴 분석이 수행됩니다. 이렇게 하면 패턴 결과가 더 정확해지지만 pattern 명령을 사용할 때 원시 로그 이벤트가 반환되지 않습니다. 쿼리에 포함되지 pattern 않은 경우 패턴 결과는 처음 1000개의 반환된 로그 이벤트 또는 쿼리에 사용한 제한값을 기반으로 합니다. pattern쿼리에 포함하는 경우 패턴 탭에 표시되는 결과는 쿼리와 일치하는 모든 로그 이벤트에서 파생됩니다.

CloudWatch Logs Insights에서 패턴 분석을 시작하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.

2. 탐색 창에서 로그, 로그 인사이트를 선택합니다.

   로그 인사이트 페이지의 쿼리 편집기에는 최신 로그 이벤트 20개를 반환하는 기본 쿼리가 포함되어 있습니다.

3. 쿼리 상자에서 해당 `| limit 20` 줄을 제거하여 쿼리가 다음과 같이 표시되도록 합니다.

   ```
   fields @timestamp, @message, @logStream, @log
   | sort @timestamp desc
   ```

4. 로그 그룹 선택 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.
5. (선택 사항) 시간 간격 선택기를 사용하여 쿼리할 기간을 선택합니다.

   5분에서 30분 간격, 1시간, 3시간, 12시간 간격 또는 사용자 지정 시간 프레임 중에서 선택할 수 있습니다.

6. 쿼리 실행을 선택하여 쿼리를 시작합니다.

   쿼리 실행이 끝나면 로그 탭에 쿼리에서 반환된 로그 이벤트 테이블이 표시됩니다. 테이블 위에는 쿼리와 일치하는 레코드 수에 대한 메시지가 있는데, 이는 71,101개의 일치하는 레코드 중 1000개 표시와 비슷합니다.

7. 패턴 탭을 선택합니다.
8. 이제 쿼리에서 찾은 패턴이 테이블에 표시됩니다. 쿼리에 **pattern** 명령이 포함되지 않았으므로 이 탭에는 로그 탭의 표에 표시된 1000개의 로그 이벤트 중에서 발견된 패턴만 표시됩니다.

   각 패턴에 대해 다음과 같은 정보가 표시됩니다.

   - 패턴, 각 동적 토큰이 로 표시됩니다<*>.
   - 이벤트 수는 쿼리된 로그 이벤트에 패턴이 나타난 횟수입니다. 이벤트 횟수 열 제목을 선택하여 패턴을 빈도별로 정렬합니다.
   - 이벤트 비율은 쿼리된 로그 이벤트 중 이 패턴을 포함하는 이벤트 이벤트의 백분율입니다.
   - 심각도 유형은 다음 중 하나입니다.
     - 패턴에 오류라는 단어가 포함된 경우 오류가 발생합니다.
     - 패턴에 Warn이라는 단어는 포함되지만 오류는 포함되지 않은 경우 경고합니다.
     - 패턴에 경고 또는 오류가 포함되지 않은 경우의 정보

   심각도 정보 열 제목을 선택하여 심각도별로 패턴을 정렬합니다.

9. 이제 쿼리를 변경하십시오. 전체 쿼리가 다음과 같이 되도록 쿼리의 `| sort @timestamp desc` 줄을 로 바꾸십시오`| pattern @message`.

```
fields @timestamp, @message, @logStream, @log
| pattern @message
```

10. 쿼리 실행을 선택합니다.

    쿼리가 끝나면 로그 탭에 결과가 표시되지 않습니다. 그러나 쿼리된 로그 이벤트의 총 수에 따라 패턴 탭에는 더 많은 수의 패턴이 나열될 수 있습니다.

11. pattern쿼리에 포함했는지 여부에 관계없이 쿼리가 반환하는 패턴을 더 자세히 검사할 수 있습니다. 이렇게 하려면 Inspect 열에서 패턴 중 하나의 아이콘을 선택하세요.

    패턴 검사 창이 나타나고 다음이 표시됩니다.

    - 패턴. 패턴 내에서 토큰을 선택하여 해당 토큰의 값을 분석하십시오.
    - 쿼리된 시간 범위 동안의 패턴 발생 횟수를 보여주는 히스토그램입니다. 이를 통해 패턴 발생의 급격한 증가와 같은 흥미로운 추세를 파악할 수 있습니다.
    - 로그 샘플 탭에는 선택한 패턴과 일치하는 몇 가지 로그 이벤트가 표시됩니다.
    - 토큰 값 탭에는 선택한 동적 토큰의 값이 표시됩니다 (선택한 경우).

      > **Note**
      >
      > 각 토큰에 대해 최대 10개의 토큰 값이 캡처됩니다. 토큰 수가 정확하지 않을 수 있습니다. CloudWatch 로그는 절대값이 아닌 확률적 카운터를 사용하여 토큰 수를 생성합니다.

    - 관련 패턴 탭에는 검사 중인 패턴과 거의 같은 시기에 자주 발생한 다른 패턴이 표시됩니다. 예를 들어, 일반적으로 ERROR 메시지 패턴에 추가 세부 정보가 표시된 다른 로그 이벤트가 수반되는 경우 해당 패턴이 여기에 표시됩니다. INFO

패턴 명령에 대한 세부 정보

이 섹션에는 pattern 명령 및 사용에 대한 자세한 내용이 포함되어 있습니다.

- 이전 자습서에서는 sort 명령 뒤에 명령이 포함된 쿼리는 유효하지 pattern 않기 때문에 명령을 추가할 때 이 pattern 명령을 제거했습니다. sort a pattern 앞에 가 있어도 sort 유효합니다.

  pattern구문에 대한 자세한 내용은 을 참조하십시오패턴.
- 쿼리에 사용할 때는 pattern pattern 명령에서 선택한 필드 중 @message 하나여야 합니다.

- `filter`명령 앞에 명령을 포함하여 필터링된 로그 이벤트 집합만 패턴 분석을 위한 입력으로 사용되도록 할 수 있습니다. `pattern`
- `parse`명령에서 파생된 필드와 같은 특정 필드의 패턴 결과를 보려면 를 사용하십시오`pattern @fieldname`.
- `stats`명령을 사용한 쿼리와 같이 로그가 아닌 출력이 있는 쿼리는 패턴 결과를 반환하지 않습니다.

이전 시간 범위와 비교 (diff)

CloudWatch Logs Insights를 사용하여 시간 경과에 따른 로그 이벤트의 변화를 비교할 수 있습니다. 최근 기간 동안 수집된 로그 이벤트를 바로 이전 기간의 로그와 비교할 수 있습니다. 또는 비슷한 과거 기간과 비교할 수도 있습니다. 이렇게 하면 로그의 오류가 최근에 발생했는지 아니면 이미 발생했는지 확인하고 다른 추세를 찾는 데 도움이 될 수 있습니다.

비교 쿼리는 결과에서 패턴만 반환하고 원시 로그 이벤트는 반환하지 않습니다. 반환된 패턴을 통해 시간 경과에 따른 로그 이벤트의 추세와 변화를 빠르게 확인할 수 있습니다. 비교 쿼리를 실행하고 패턴 결과를 얻으면 원하는 패턴의 샘플 원시 로그 이벤트를 확인할 수 있습니다. 로그 패턴에 대한 자세한 내용은 을 참조하십시오패턴 분석.

비교 쿼리를 실행하면 두 개의 다른 기간, 즉 선택한 원래 쿼리 기간과 비교 기간을 기준으로 쿼리가 분석됩니다. 비교 기간은 항상 원래 쿼리 기간과 동일합니다. 비교를 위한 기본 시간 간격은 다음과 같습니다.

- 이전 기간 — 쿼리 기간 직전의 기간과 비교합니다.
- 이전 날짜 — 쿼리 기간의 하루 전 기간과 비교합니다.
- 이전 주 — 쿼리 기간 1주 전의 기간과 비교합니다.
- 이전 달 — 쿼리 기간 1개월 전의 기간과 비교합니다.

> **Note**
>
> 비교를 사용하는 쿼리에는 합친 시간 범위에서 단일 CloudWatch Logs Insights 쿼리를 실행하는 것과 비슷한 요금이 부과됩니다. 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

비교 쿼리를 실행하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.

2. 탐색 창에서 로그, 로그 인사이트를 선택합니다.

   쿼리 상자에 기본 쿼리가 표시됩니다.

3. 기본 쿼리를 유지하거나 다른 쿼리를 입력합니다.
4. 로그 그룹 선택 드롭다운에서 쿼리할 로그 그룹을 하나 이상 선택합니다.
5. (선택 사항) 시간 간격 선택기를 사용하여 쿼리할 기간을 선택합니다. 기본 쿼리는 이전 1시간의 로그 데이터에 대한 것입니다.
6. 시간 범위 선택기에서 비교를 선택합니다. 그런 다음 원본 로그를 비교할 이전 기간을 선택하고 [적용] 을 선택합니다.
7. 쿼리 실행을 선택합니다.

   쿼리에서 비교 기간의 데이터를 가져오도록 하려면 쿼리에 `diff` 명령이 추가됩니다.

8. 결과를 보려면 패턴 탭을 선택합니다.

   테이블에는 다음 정보가 표시됩니다.

   - 각 패턴 (패턴의 가변 부분이 동적 토큰 <*> 심볼로 대체됨) 자세한 정보는 패턴 분석을 참조하세요.
   - 이벤트 수는 원래의 최신 기간에 해당 패턴을 가진 로그 이벤트의 수입니다.
   - 차이 이벤트 수는 현재 기간의 일치하는 로그 이벤트 수와 비교 기간의 차이입니다. 차이가 양수이면 현재 기간에 해당 이벤트가 더 많다는 뜻입니다.
   - 차이 설명에는 현재 기간과 비교 기간 간의 해당 패턴 변화가 간략하게 요약되어 있습니다.
   - 심각도 유형은,, 와 같은 `FATAL` 로그 이벤트에서 발견된 단어를 기반으로 한 이 패턴의 로그 이벤트의 예상 심각도입니다. `ERROR` `WARN`

9. 목록에 있는 패턴 중 하나를 더 자세히 검사하려면 Inspect 열에서 패턴 중 하나의 아이콘을 선택하십시오.

   패턴 검사 창이 나타나고 다음이 표시됩니다.

   - 패턴. 패턴 내에서 토큰을 선택하여 해당 토큰의 값을 분석하십시오.
   - 쿼리된 시간 범위 동안의 패턴 발생 횟수를 보여주는 히스토그램입니다. 이를 통해 패턴 발생의 급격한 증가와 같은 흥미로운 추세를 파악할 수 있습니다.
   - 로그 샘플 탭에는 선택한 패턴과 일치하는 몇 가지 로그 이벤트가 표시됩니다.
   - 토큰 값 탭에는 선택한 동적 토큰의 값이 표시됩니다 (선택한 경우).

> **Note**
>
> 각 토큰에 대해 최대 10개의 토큰 값이 캡처됩니다. 토큰 수가 정확하지 않을 수 있습니다. CloudWatch 로그는 절대값이 아닌 확률적 카운터를 사용하여 토큰 수를 생성합니다.

- 관련 패턴 탭에는 검사 중인 패턴과 거의 같은 시기에 자주 발생한 다른 패턴이 표시됩니다. 예를 들어, 일반적으로 `ERROR` 메시지 패턴에 추가 세부 정보가 표시된 다른 로그 이벤트가 수반되는 경우 해당 패턴이 여기에 표시됩니다. `INFO`

샘플 쿼리

이 섹션에는 CloudWatch 콘솔에서 실행할 수 있는 일반적이고 유용한 쿼리 명령 목록이 포함되어 있습니다. 쿼리 명령을 실행하는 방법에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 자습서: 샘플 쿼리 실행 및 수정을 참조하십시오.

쿼리 구문에 대한 자세한 내용은 을 참조하십시오CloudWatch 로그 인사이트 쿼리 구문.

주제

- 일반 쿼리
- Lambda 로그에 대한 쿼리
- Amazon VPC 흐름 로그에 대한 쿼리
- Route 53 로그에 대한 쿼리
- CloudTrail 로그 쿼리
- 에 대한 쿼리 Amazon API Gateway
- NAT 게이트웨이에 대한 쿼리
- Apache 서버 로그에 대한 쿼리
- 아마존용 쿼리 EventBridge
- 분석 명령의 예제

일반 쿼리

최근에 추가된 로그 이벤트 25개를 찾습니다.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

시간당 발생한 예외 수 목록을 가져옵니다.

```
filter @message like /Exception/
    | stats count(*) as exceptionCount by bin(1h)
    | sort exceptionCount desc
```

예외에 해당되지 않는 로그 이벤트 목록을 가져옵니다.

```
fields @message | filter @message not like /Exception/
```

server 필드의 각 고유 값에 대한 가장 최근 로그 이벤트 가져옵니다.

```
fields @timestamp, server, severity, message
| sort @timestamp asc
| dedup server
```

각 **severity** 유형에 대한 **server** 필드의 각 고유 값에 대한 가장 최근 로그 이벤트를 가져옵니다.

```
fields @timestamp, server, severity, message
| sort @timestamp desc
| dedup server, severity
```

Lambda 로그에 대한 쿼리

과다 프로비저닝된 메모리 양을 확인합니다.

```
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
```

지연 보고서를 생성합니다.

```
filter @type = "REPORT" |
    stats avg(@duration), max(@duration), min(@duration) by bin(5m)
```

느린 함수 호출을 검색하고 재시도 또는 클라이언트측 코드에서 발생할 수 있는 중복 요청을 제거합니다. 이 쿼리에서 **@duration**은 밀리초 단위입니다.

```
fields @timestamp, @requestId, @message, @logStream
| filter @type = "REPORT" and @duration > 1000
| sort @timestamp desc
| dedup @requestId
| limit 20
```

Amazon VPC 흐름 로그에 대한 쿼리

호스트 간에 상위 15개의 패킷 전송을 찾습니다.

```
stats sum(packets) as packetsTransferred by srcAddr, dstAddr
    | sort packetsTransferred  desc
    | limit 15
```

지정된 서브넷의 호스트에 대해 상위 15개 바이트 전송을 찾습니다.

```
filter isIpv4InSubnet(srcAddr, "192.0.2.0/24")
    | stats sum(bytes) as bytesTransferred by dstAddr
    | sort bytesTransferred desc
    | limit 15
```

데이터 전송 프로토콜로 UDP를 사용하는 IP 주소를 찾습니다.

```
filter protocol=17 | stats count(*) by srcAddr
```

캡처 기간 중 흐름 레코드를 건너뛴 IP 주소를 찾습니다.

```
filter logStatus="SKIPDATA"
    | stats count(*) by bin(1h) as t
    | sort t
```

네트워크 연결 문제 해결에 도움이 되도록 각 연결에 대한 단일 레코드를 찾습니다.

```
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
```

Route 53 로그에 대한 쿼리

시간당 레코드 배포를 쿼리 유형별로 찾습니다.

```
stats count(*) by queryType, bin(1h)
```

요청 수가 가장 많은 DNS 해석기 10개를 찾습니다.

```
stats count(*) as numRequests by resolverIp
    | sort numRequests desc
    | limit 10
```

서버가 DNS 요청을 완료하지 못한 도메인 및 하위 도메인별 레코드 수를 찾습니다.

```
filter responseCode="SERVFAIL" | stats count(*) by queryName
```

CloudTrail 로그 쿼리

각 서비스, 이벤트 유형 및 AWS 리전에 대한 로그 항목 수를 찾습니다.

```
stats count(*) by eventSource, eventName, awsRegion
```

특정 AWS 지역에서 시작 또는 중지된 Amazon EC2 호스트를 찾습니다.

```
filter (eventName="StartInstances" or eventName="StopInstances") and awsRegion="us-
east-2"
```

새로 생성한 IAM 사용자의 AWS 지역, 사용자 이름, ARN을 찾을 수 있습니다.

```
filter eventName="CreateUser"
```

```
    | fields awsRegion, requestParameters.userName, responseElements.user.arn
```

API **UpdateTrail**을 호출하는 중 예외가 발생한 레코드 수를 찾습니다.

```
filter eventName="UpdateTrail" and ispresent(errorCode)
    | stats count(*) by errorCode, errorMessage
```

TLS 1.0 또는 1.1이 사용된 로그 항목을 찾습니다.

```
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
```

TLS 버전 1.0 또는 1.1을 사용한 서비스당 호출 수를 찾습니다.

```
filter tlsDetails.tlsVersion in [ "TLSv1", "TLSv1.1" ]
| stats count(*) as numOutdatedTlsCalls by eventSource
| sort numOutdatedTlsCalls desc
```

에 대한 쿼리 Amazon API Gateway

마지막 10개의 4XX 오류 찾기

```
fields @timestamp, status, ip, path, httpMethod
| filter status>=400 and status<=499
| sort @timestamp desc
| limit 10
```

액세스 로그 그룹에서 가장 오래 실행되는 Amazon API Gateway 요청 10개를 식별하세요. Amazon API Gateway

```
fields @timestamp, status, ip, path, httpMethod, responseLatency
| sort responseLatency desc
```

```
| limit 10
```

액세스 로그 그룹에서 가장 많이 사용되는 API 경로 목록을 반환하세요. Amazon API Gateway

```
stats count(*) as requestCount by path
| sort requestCount desc
| limit 10
```

Amazon API Gateway 액세스 로그 그룹에 대한 통합 지연 보고서를 생성하세요.

```
filter status=200
| stats avg(integrationLatency), max(integrationLatency),
min(integrationLatency) by bin(1m)
```

NAT 게이트웨이에 대한 쿼리

AWS 청구서에서 비용이 정상보다 높은 것으로 확인되면 CloudWatch Logs Insights를 사용하여 상위 기여자를 찾을 수 있습니다. 다음 쿼리 명령에 대한 자세한 내용은 VPC의 NAT 게이트웨이를 통해 트래픽에 가장 많이 기여한 사용자를 찾으려면 어떻게 해야 합니까? 를 참조하십시오. AWS 프리미엄 지원 페이지에서

> **Note**
>
> 다음 쿼리 명령에서 'x.x.x.x'를 NAT 게이트웨이의 프라이빗 IP로 바꾸고 'y.y'를 VPC CIDR 범위의 처음 두 옥텟으로 바꿉니다.

NAT 게이트웨이를 통해 가장 많은 트래픽을 전송하는 인스턴스를 찾습니다.

```
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
```

NAT 게이트웨이의 인스턴스에서 송수신되는 트래픽을 확인합니다.

```
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
```

VPC 인스턴스가 업로드 및 다운로드를 위해 가장 자주 통신하는 인터넷 대상을 결정합니다.

업로드용

```
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
```

다운로드용

```
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
```

Apache 서버 로그에 대한 쿼리

CloudWatch 로그 인사이트를 사용하여 Apache 서버 로그를 쿼리할 수 있습니다. 다음 쿼리에 대한 자세한 내용은 AWS 클라우드 운영 및 마이그레이션 블로그에서 Logs Insights를 통한 Apache 서버 CloudWatch 로그 단순화를 참조하십시오.

가장 관련성이 높은 필드를 찾아서 액세스 로그를 검토하고 애플리케이션의 /admin 경로에서 트래픽을 확인할 수 있습니다.

```
fields @timestamp, remoteIP, request, status, filename| sort @timestamp desc
| filter filename="/var/www/html/admin"
| limit 20
```

상태 코드 '200'(성공)을 사용하여 기본 페이지에 액세스한 고유 GET 요청 수를 찾습니다.

```
fields @timestamp, remoteIP, method, status
| filter status="200" and referrer= http://34.250.27.141/ and method= "GET"
| stats count_distinct(remoteIP) as UniqueVisits
| limit 10
```

Apache 서비스가 다시 시작된 횟수를 찾습니다.

```
fields @timestamp, function, process, message
| filter message like "resuming normal operations"
| sort @timestamp desc
| limit 20
```

아마존용 쿼리 EventBridge

이벤트 세부 정보 유형별로 그룹화된 EventBridge 이벤트 수 가져오기

```
fields @timestamp, @message
| stats count(*) as numberOfEvents by `detail-type`
| sort numberOfEvents desc
```

분석 명령의 예제

glob 표현식을 사용하여 로그 필드 **@message**에서 필드 **@user**, **@method** 및 **@latency**를 추출하고 **@method** 및 **@user**의 고유한 개별 조합에 대한 평균 지연 시간을 반환합니다.

```
parse @message "user=*, method:*, latency := *" as @user,
    @method, @latency | stats avg(@latency) by @method,
    @user
```

정규식을 사용하여 로그 필드 **@message**에서 필드 **@user2**, **@method2** 및 **@latency2**를 추출하고 **@method2** 및 **@user2**의 고유한 개별 조합에 대한 평균 지연 시간을 반환합니다.

```
parse @message /user=(?<user2>.*?), method:(?<method2>.*?),
    latency := (?<latency2>.*?)/ | stats avg(latency2) by @method2,
    @user2
```

loggingTime, **loggingType** 및 **loggingMessage** 필드를 추출하고 **ERROR** 또는 **INFO** 문자열이 포함된 이벤트를 기록하도록 필터링한 다음 **ERROR** 문자열이 포함된 이벤트에 대해 **loggingMessage** 및 **loggingType** 필드만 표시합니다.

```
FIELDS @message
    | PARSE @message "* [*] *" as loggingTime, loggingType, loggingMessage
    | FILTER loggingType IN ["ERROR", "INFO"]
```

```
    | DISPLAY loggingMessage, loggingType = "ERROR" as isError
```

그래프로 로그 데이터 시각화

막대형 차트, 선형 차트, 누적 영역 차트와 같은 시각화를 사용하여 로그 데이터의 패턴을 보다 효율적으로 식별할 수 있습니다. CloudWatch Logs Insights는 해당 `stats` 함수와 하나 이상의 집계 함수를 사용하는 쿼리에 대한 시각화를 생성합니다. 자세한 내용은 stats를 참조하세요.

Logs Insights 쿼리를 저장하고 다시 실행합니다 CloudWatch .

쿼리를 생성한 후 나중에 다시 실행할 수 있도록 저장할 수 있습니다. 쿼리는 폴더 구조에 저장되므로 쿼리를 정리할 수 있습니다. 계정별로 리전당 최대 1,000개의 쿼리를 저장할 수 있습니다.

쿼리를 저장하려면 `logs:PutQueryDefinition` 권한이 있는 역할에 로그인해야 합니다. 저장된 쿼리의 목록을 보려면 `logs:DescribeQueryDefinitions` 권한이 있는 역할에 로그인해야 합니다.

쿼리를 저장하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 쿼리 편집기에서 쿼리를 작성합니다.
4. 저장을 선택합니다.

   저장 버튼이 보이지 않으면 CloudWatch 로그 콘솔의 새 디자인으로 변경해야 합니다. 그렇게 하려면 다음을 수행하세요.

   a. 탐색 창에서 로그 그룹을 선택합니다.
   b. 새 디자인 사용해 보기를 선택합니다.
   c. 탐색 창에서 Insights를 선택하고 이 절차의 3단계로 돌아갑니다.
5. 쿼리의 이름을 입력합니다.
6. (선택 사항) 쿼리를 저장할 폴더를 선택합니다. 새로 생성을 선택하여 폴더를 만듭니다. 새 폴더를 만드는 경우 폴더 이름에 슬래시(/) 문자를 사용하여 폴더 구조를 정의할 수 있습니다. 예를 들어 새 폴더의 이름을 **`folder-level-1/folder-level-2`**로 지정하면 **`folder-level-1`**이라는 최상위 폴더가 만들어지고 그 폴더 안에 **`folder-level-2`**라는 다른 폴더가 만들어집니다. 쿼리가 **`folder-level-2`**에 저장됩니다.
7. (선택 사항) 쿼리의 로그 그룹 또는 쿼리 텍스트를 변경합니다.

8. 저장을 선택합니다.

> **Tip**
>
> `PutQueryDefinition`을 사용하여 저장된 쿼리에 대한 폴더를 만들 수 있습니다. 저장된 쿼리에 대한 폴더를 만들려면 슬래시(/)를 사용하여 원하는 쿼리 이름에 원하는 폴더 이름을 접두사로 붙입니다. <*folder-name*>/<*query-name*>. 이 작업에 대한 자세한 내용은 을 참조하십시오 PutQueryDefinition.

저장된 쿼리를 실행하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 오른쪽에서 쿼리를 선택합니다.
4. 저장된 쿼리 목록에서 쿼리를 선택합니다. 이 쿼리가 쿼리 편집기에 나타납니다.
5. Run(실행)을 선택합니다.

저장된 쿼리의 새 버전을 저장하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 오른쪽에서 쿼리를 선택합니다.
4. 저장된 쿼리 목록에서 쿼리를 선택합니다. 이 쿼리가 쿼리 편집기에 나타납니다.
5. 쿼리를 수정합니다. 작업을 확인하기 위해 쿼리를 실행해야 하는 경우 쿼리 실행을 선택합니다.
6. 새 버전을 저장할 준비가 되면 작업, 다른 이름으로 저장을 선택합니다.
7. 쿼리의 이름을 입력합니다.
8. (선택 사항) 쿼리를 저장할 폴더를 선택합니다. 새로 생성을 선택하여 폴더를 만듭니다. 새 폴더를 만드는 경우 폴더 이름에 슬래시(/) 문자를 사용하여 폴더 구조를 정의할 수 있습니다. 예를 들어 새 폴더의 이름을 **`folder-level-1/folder-level-2`**로 지정하면 **`folder-level-1`**이라는 최상위 폴더가 만들어지고 그 폴더 안에 **`folder-level-2`**라는 다른 폴더가 만들어집니다. 쿼리가 **`folder-level-2`**에 저장됩니다.
9. (선택 사항) 쿼리의 로그 그룹 또는 쿼리 텍스트를 변경합니다.
10. 저장을 선택합니다.

쿼리를 삭제하려면 `logs:DeleteQueryDefinition` 권한이 있는 역할에 로그인해야 합니다.

저장된 쿼리를 편집 또는 삭제하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 오른쪽에서 쿼리를 선택합니다.
4. 저장된 쿼리 목록에서 쿼리를 선택합니다. 이 쿼리가 쿼리 편집기에 나타납니다.
5. 작업, 편집 또는 작업, 삭제를 선택합니다.

대시보드에 쿼리 추가 또는 쿼리 결과 내보내기

쿼리를 실행한 후 CloudWatch 대시보드에 쿼리를 추가하거나 결과를 클립보드에 복사할 수 있습니다.

대시보드에 추가한 쿼리는 대시보드를 로드할 때마다 그리고 대시보드를 새로 고칠 때마다 실행됩니다. 이러한 쿼리는 동시 CloudWatch Logs Insights 쿼리 한도인 30개에 포함됩니다.

대시보드에 쿼리 결과를 추가하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 로그 그룹을 하나 이상 선택하고 쿼리를 실행합니다.
4. 대시보드에 추가(Add to dashboard)를 선택합니다.
5. 대시보드를 선택하거나 새로 생성을 선택하여 쿼리 결과에 대한 새 대시보드를 생성합니다.
6. 쿼리 결과에 사용할 위젯 유형을 선택합니다.
7. 위젯의 이름을 입력합니다.
8. 대시보드에 추가(Add to dashboard)를 선택합니다.

쿼리 결과를 클립보드로 복사하거나 쿼리 결과를 다운로드하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 로그 그룹을 하나 이상 선택하고 쿼리를 실행합니다.
4. 결과 내보내기를 선택한 다음 원하는 옵션을 선택합니다.

실행 중인 쿼리 또는 쿼리 기록 보기

현재 진행 중인 쿼리와 최신 쿼리 기록을 볼 수 있습니다.

현재 실행 중인 쿼리에는 대시보드에 추가한 쿼리가 포함됩니다. 대시보드에 추가된 쿼리를 포함하여 계정당 동시 CloudWatch Logs Insights 쿼리는 30개로 제한됩니다.

최근 쿼리 기록을 보려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그(Logs)를 선택한 다음, 로그 인사이트(Logs Insights)를 선택합니다.
3. 새 디자인의 CloudWatch Logs 콘솔을 사용하는 경우 [History] 를 선택합니다. 이전 디자인을 사용하는 경우 작업, 이 계정에 대한 쿼리 기록 보기를 선택합니다.

   최근 쿼리 목록이 나타납니다. 쿼리를 선택하고 실행을 선택하여 이 중 하나를 다시 실행할 수 있습니다.

   [상태] 아래에 현재 실행 중인 모든 쿼리에 대해 [진행 중] 이 표시됩니다. CloudWatch

쿼리 결과를 다음과 같이 암호화합니다. AWS Key Management Service

기본적으로 Logs는 기본 CloudWatch Logs 서버 측 암호화 방법을 사용하여 CloudWatch Logs Insights 쿼리의 저장된 결과를 암호화합니다. CloudWatch 대신 AWS KMS 키를 사용하여 이러한 결과를 암호화하도록 선택할 수 있습니다. AWS KMS 키를 암호화 결과와 연결하면 CloudWatch Logs는 해당 키를 사용하여 계정에 저장된 모든 쿼리 결과를 암호화합니다.

나중에 쿼리 결과에서 키를 분리하면 CloudWatch 로그는 이후 쿼리를 위해 기본 암호화 방법으로 돌아갑니다. 하지만 키가 연결된 동안 실행된 쿼리는 여전히 해당 키로 암호화됩니다. CloudWatch 로그에서 계속 키를 참조할 수 있기 때문에 KMS 키의 연결이 끊긴 후에도 CloudWatch 로그에서 해당 결과를 반환할 수 있습니다. 하지만 나중에 키를 비활성화하면 CloudWatch Logs는 해당 키로 암호화된 쿼리 결과를 읽을 수 없습니다.

> **Important**
>
> CloudWatch 로그는 대칭 KMS 키만 지원합니다. 비대칭 키를 사용하여 쿼리 결과를 암호화하지 마세요. 자세한 내용은 대칭 및 비대칭 키 사용을 참조하세요.

Limits

- 다음 단계를 수행하려면 `kms:CreateKey`, `kms:GetKeyPolicy` 및 `kms:PutKeyPolicy` 권한이 있어야 합니다.
- 쿼리 결과에서 키를 연결하거나 연결 해제하고 난 후 이러한 변경이 적용되기까지 최대 5분의 시간이 소요될 수 있습니다.
- 관련 키에 대한 CloudWatch 로그 액세스를 취소하거나 연결된 KMS 키를 삭제하면 CloudWatch Logs의 암호화된 데이터를 더 이상 검색할 수 없습니다.
- CloudWatch 콘솔을 사용하여 키를 연결할 수 없으며, 또는 Logs API를 사용해야 합니다. AWS CLI CloudWatch

1단계: 생성 AWS KMS key

다음 create-key 명령을 사용하여 KMS 키 생성:

```
aws kms create-key
```

이 명령의 출력 화면에는 키의 키 ID와 Amazon 리소스 이름(ARN)이 포함됩니다. 다음은 예 출력입니다.

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

2단계: KMS 키에 대한 권한 설정

기본적으로 모든 KMS 키는 비공개입니다. 리소스 소유자만 이를 사용하여 데이터를 암호화 및 해독할 수 있습니다. 그러나 리소스 소유자가 원한다면 다른 사용자 및 리소스에 키에 대한 액세스 권한을 부여할 수 있습니다. 이 단계에서는 CloudWatch 로그 서비스 주체에게 키 사용 권한을 부여합니다. 이 서비스 주체는 키가 저장된 동일한 AWS 지역에 있어야 합니다.

가장 좋은 방법은 지정한 AWS 계정으로만 키 사용을 제한하는 것입니다.

먼저 다음 get-key-policy명령어를 `policy.json` 사용하여 KMS 키의 기본 정책을 저장합니다.

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

텍스트 편집기에서 `policy.json` 파일을 열고 다음 설명 중 하나에서 굵은 글꼴로 표시된 섹션을 추가합니다. 기존 설명과 새 설명을 쉼표로 구분합니다. 이러한 명령문은 `Condition` 섹션을 사용하여 AWS KMS 키의 보안을 강화합니다. 자세한 정보는 AWS KMS 키 및 암호화 컨텍스트을 참조하세요.

이 예제의 `Condition` 섹션에서는 AWS KMS 키 사용을 지정된 계정의 CloudWatch Logs Insights 쿼리 결과로 제한합니다.

```
{
 "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::account_ID: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:region:account_ID:query-result:*"
                },
                "StringEquals": {
                    "aws:SourceAccount": "Your_account_ID"
                }
            }
        }
    ]
}
```

마지막으로 다음 put-key-policy명령어를 사용하여 업데이트된 정책을 추가합니다.

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

3단계: KMS 키를 쿼리 결과와 연결

KMS 키를 계정의 쿼리 결과와 연결

다음과 같이 disassociate-kms-key 명령을 실행합니다.

```
aws logs associate-kms-key --resource-identifier "arn:aws:logs:region:account-id:query-
result:*" --kms-key-id "key-arn"
```

4단계: 계정의 쿼리 결과에서 키 연결 해제

쿼리 결과와 연결된 KMS 키의 연결을 끊으려면 다음 disassociate-kms-key명령을 사용합니다.

```
aws logs disassociate-kms-key --resource-identifier "arn:aws:logs:region:account-
id:query-result:*"
```

자연어를 사용하여 CloudWatch Logs Insights 쿼리를 생성하고 업데이트하십시오.

> **Note**
>
> 이 기능은 일반적으로 미국 동부 (버지니아 북부), 미국 서부 (오레곤), 아시아 태평양 (도쿄) 에서 로그용으로 CloudWatch 사용할 수 있습니다.

CloudWatch 로그는 자연어 쿼리 기능을 지원하므로 CloudWatch 로그 인사이트 및 CloudWatch 메트릭 인사이트에 대한 쿼리를 생성하고 업데이트하는 데 도움이 됩니다.

이 기능을 사용하면 찾고 있는 로그 데이터에 대해 질문하거나 평이한 영어로 CloudWatch 로그 데이터에 대해 설명할 수 있습니다. 자연어 기능은 사용자가 입력한 프롬프트를 기반으로 쿼리를 생성하고 쿼리 작동 방식에 line-by-line 대한 설명을 제공합니다. 쿼리를 업데이트하여 데이터를 더 자세히 조사할 수도 있습니다.

환경에 따라 “전송된 바이트 기준 상위 100개의 소스 IP 주소는 무엇입니까?” 와 같은 프롬프트를 입력할 수 있습니다. 및 “가장 느린 Lambda 함수 요청 10개 찾기”

이 기능을 사용하여 CloudWatch Logs Insights 쿼리를 생성하려면 CloudWatch Logs Insights 쿼리 편집기를 열고 쿼리하려는 로그 그룹을 선택한 다음 쿼리 생성을 선택합니다.

> **Important**
>
> 자연어 쿼리 기능을 사용하려면, CloudWatchLogsFullAccessCloudWatchLogsReadOnlyAccessAdministratorAccess, 또는 ReadOnlyAccess정책을 사용해야 합니다.
> 신규 또는 기존 고객 관리형 또는 인라인 정책에 `cloudwatch:GenerateQuery` 작업을 포함시킬 수도 있습니다.

쿼리 예제

이 섹션의 예제에서는 자연어 기능을 사용하여 쿼리를 생성하고 업데이트하는 방법을 설명합니다.

> **Note**
>
> CloudWatch Logs Insights 쿼리 편집기 및 구문에 대한 자세한 내용은 CloudWatch Logs Insights 쿼리 구문을 참조하십시오.

예제: 자연어 쿼리 생성

자연어를 사용하여 쿼리를 생성하려면 프롬프트를 입력하고 새 쿼리 생성을 선택합니다. 이 예제는 기본 검색을 수행하는 쿼리를 보여줍니다.

프롬프트

다음은 가장 느린 Lambda 함수 호출 10개를 검색하도록 기능을 지시하는 프롬프트의 예입니다.

```
Find the 10 slowest requests
```

Query

다음은 프롬프트를 기반으로 자연 언어 기능이 생성하는 쿼리의 예입니다. 쿼리 전에 댓글에 프롬프트가 어떻게 나타나는지 확인합니다. 쿼리가 끝나면 쿼리 작동 방식을 설명하는 설명을 읽을 수 있습니다.

```
# 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**
>
> 프롬프트 표시 및 쿼리 작동 방식에 대한 설명을 끄려면 편집기의 톱니바퀴 아이콘을 사용합니다.

예제: 자연어 쿼리 업데이트

초기 프롬프트를 편집한 다음, 쿼리 업데이트를 선택하여 쿼리를 업데이트할 수 있습니다.

업데이트된 프롬프트

다음 예제는 이전 프롬프트의 업데이트된 버전을 보여줍니다. 가장 느린 Lambda 함수 호출 10개를 검색하는 프롬프트 대신, 이제 이 프롬프트는 속도가 가장 느린 Lambda 함수 호출 20개를 검색하고 추가 로그 이벤트를 위한 다른 열을 포함하도록 기능을 지시합니다.

```
Show top 20 slowest requests instead and display requestId as a column
```

업데이트된 쿼리

다음은 업데이트된 쿼리의 예제입니다. 업데이트된 쿼리가 나오기 전에 댓글에 업데이트된 프롬프트가 어떻게 나타나는지 확인합니다. 쿼리가 끝나면 원래 쿼리가 어떻게 업데이트되었는지에 대한 설명을 읽을 수 있습니다.

```
# 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.
```

서비스 개선을 위한 데이터 사용 거부

AI 모델을 학습시키고 관련 쿼리를 생성하기 위해 제공하는 자연어 프롬프트 데이터는 서비스를 제공하고 유지하는 용도로만 사용됩니다. 이 데이터는 Logs Insights의 품질을 개선하는 데 사용될 수 있습니다. CloudWatch 고객의 신뢰와 개인 정보 보호는 물론 콘텐츠 보안도 당사의 최우선 과제입니다. 자세한 내용은 AWS 서비스 약관과 AWS 책임감 있는 AI 정책을 참조하세요.

AI 서비스 옵트아웃 정책을 생성하여 콘텐츠가 자연어 쿼리의 개발 또는 품질 향상에 사용되는 것을 거부할 수 있습니다. 쿼리 생성 기능을 포함하여 모든 CloudWatch Logs AI 기능에 대한 데이터 수집을 거부하려면 CloudWatch 로그에 대한 옵트아웃 정책을 만들어야 합니다. 자세한 내용은 AWS Organizations 사용 설명서의 AI 서비스 옵트아웃 정책을 참조하세요.

로그 이상 탐지

각 로그 그룹에 대해 로그 이상 탐지기를 만들 수 있습니다. 예외 항목 탐지기는 로그 그룹에 수집된 로그 이벤트를 스캔하고 로그 데이터에서 이상을 찾습니다. 이상 탐지는 기계 학습과 패턴 인식을 사용하여 일반적인 로그 콘텐츠의 기준을 설정합니다.

로그 그룹에 대한 예외 항목 탐지기를 만든 후에는 로그 그룹에서 지난 2주간의 로그 이벤트를 학습에 사용하여 훈련합니다. 교육 기간은 최대 15분이 소요될 수 있습니다. 교육이 완료되면 들어오는 로그를 분석하여 이상 징후를 식별하기 시작합니다. 그러면 로그 콘솔에 이상 징후가 표시되어 검사할 수 있습니다. CloudWatch

CloudWatch 로그 패턴 인식은 로그의 정적 및 동적 콘텐츠를 식별하여 로그 패턴을 추출합니다. 패턴은 많은 수의 로그 이벤트가 종종 몇 가지 패턴으로 압축될 수 있기 때문에 대규모 로그 세트를 분석하는 데 유용합니다.

예를 들어, 세 가지 로그 이벤트의 다음 샘플을 참조하십시오.

```
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
```

이전 샘플에서는 세 개의 로그 이벤트가 모두 한 가지 패턴을 따릅니다.

```
<*> <*> [INFO] Calling DynamoDB to store for resource id <*>
```

패턴 내의 필드를 토큰이라고 합니다. 요청 ID 또는 타임스탬프와 같이 패턴 내에서 달라지는 필드를 동적 토큰이라고 합니다. 동적 토큰은 CloudWatch Logs에 <*> 패턴이 표시되는 시점으로 표시됩니다. 동적 토큰에 대해 발견된 각각의 다른 값을 토큰 값이라고 합니다.

동적 토큰의 일반적인 예로는 오류 코드, 타임스탬프, 요청 ID 등이 있습니다.

로그 이상 탐지는 이러한 패턴을 사용하여 이상을 찾습니다. 이상 탐지기 모델 교육 기간이 지나면 알려진 추세와 비교하여 로그를 평가합니다. 이상 감지기는 큰 변동을 이상 현상으로 표시합니다.

로그 이상 탐지기 생성에는 요금이 부과되지 않습니다.

이상 및 패턴의 심각도 및 우선 순위

로그 이상 탐지기에서 발견된 각 예외 항목에는 우선 순위가 할당됩니다. 발견된 각 패턴에는 심각도가 할당됩니다.

- 우선 순위는 자동으로 계산되며, 패턴의 심각도 수준과 예상 값과의 편차 정도에 따라 결정됩니다. 예를 들어 특정 토큰 값이 갑자기 500% 증가하면 심각도가 높더라도 해당 예외 항목이 HIGH 우선 순위로 지정될 수 있습니다. NONE
- 심각도는FATAL,, ERROR 등의 패턴에서 발견된 키워드만을 기준으로 합니다. WARN 이러한 키워드가 하나도 발견되지 않으면 패턴의 심각도가 로 표시됩니다NONE.

예외 항목 가시성 시간

예외 항목 탐지기를 만들 때 해당 탐지기의 최대 예외 항목 가시성 기간을 지정합니다. 콘솔에 예외 항목이 표시되고 API 작업에 의해 반환되는 일수입니다. ListAnomalies 이 기간이 경과한 이상 징후가 계속 발생하면 자동으로 정상 동작으로 받아들여지고 이상 탐지기 모델은 이를 이상 징후로 표시하는 것을 중단합니다.

이상 감지기를 만들 때 가시성 시간을 조정하지 않으면 21일이 기본값으로 사용됩니다.

예외 항목 억제

예외 항목이 발견된 후 일시적 또는 영구적으로 억제하도록 선택할 수 있습니다. 예외 항목을 억제하면 예외 항목 탐지기가 지정한 시간 동안 이 발생을 예외 항목으로 플래깅하는 것을 중지합니다. 예외를 억제할 때 해당 특정 예외만 억제하거나 예외가 발견된 패턴과 관련된 모든 예외를 억제하도록 선택할 수 있습니다.

콘솔에서는 숨겨진 예외 항목을 계속 볼 수 있습니다. 억제를 중단하도록 선택할 수도 있습니다.

자주 묻는 질문(FAQ)

내 데이터를 AWS 사용하여 기계 학습 알고리즘을 학습시켜 사용하거나 다른 고객을 위해 AWS 사용하나요?

아니요. 교육을 통해 생성된 이상 탐지 모델은 로그 그룹의 로그 이벤트를 기반으로 하며 해당 로그 그룹과 해당 계정 내에서만 사용됩니다. AWS

예외 항목 탐지에 적합한 로그 이벤트 유형은 무엇입니까?

로그 이상 탐지는 애플리케이션 로그 및 대부분의 로그 항목이 일반적인 패턴에 맞는 기타 유형의 로그에 적합합니다. INFO, ERROR, DEBUG와 같은 로그 수준 또는 심각도 키워드가 포함된 이벤트가 있는 로그 그룹은 로그 이상 탐지에 특히 적합합니다.

로그 이상 탐지는 로그와 같이 JSON 구조가 매우 긴 로그 이벤트에는 적합하지 않습니다. CloudTrail 패턴 분석은 로그 줄의 처음 1500자까지만 분석하므로 이 제한을 초과하는 문자는 건너뛰게 됩니다.

VPC 흐름 로그와 같은 감사 또는 액세스 로그도 예외 항목 탐지의 성공률이 떨어집니다. 이상 탐지는 애플리케이션 문제를 찾기 위한 것이므로 네트워크 또는 액세스 이상에는 적합하지 않을 수 있습니다.

예외 항목 탐지기가 특정 로그 그룹에 적합한지 여부를 판단하려면 로그 패턴 분석을 사용하여 그룹 내 CloudWatch 로그 이벤트의 패턴 수를 찾아보십시오. 패턴 수가 약 300개를 넘지 않는 경우 이상 탐지가 잘 작동할 수 있습니다. 패턴 분석에 대한 자세한 내용은 을 참조하십시오. 패턴 분석

예외 항목으로 플래그가 지정되는 것은 무엇입니까?

다음과 같은 경우 로그 이벤트가 예외 항목으로 플래그가 지정될 수 있습니다.

- 이전에 로그 그룹에서 볼 수 없었던 패턴을 가진 로그 이벤트입니다.
- 알려진 패턴과 크게 달라졌습니다.
- 개별 일반 값 집합이 있는 동적 토큰의 새 값입니다.
- 동적 토큰의 값 발생 횟수가 크게 변경되었습니다.

위의 모든 항목이 예외 항목으로 표시될 수 있지만, 그렇다고 해서 응용 프로그램이 제대로 작동하지 않는 것은 아닙니다. 예를 들어 higher-than-usual 여러 `200` 성공 값이 예외 항목으로 표시될 수 있습니다. 이와 같은 경우에는 문제를 나타내지 않는 이러한 예외를 억제하는 방안을 고려할 수 있습니다.

마스킹되는 민감한 데이터는 어떻게 되나요?

민감한 데이터로 마스킹된 로그 이벤트의 모든 부분은 예외 항목이 있는지 스캔되지 않습니다. 민감한 데이터를 마스킹하는 방법에 대한 자세한 내용은 마스킹을 통한 민감한 로그 데이터 보호 지원을 참조하십시오.

로그 그룹에서 이상 항목 탐지를 활성화합니다.

다음 단계를 사용하여 CloudWatch 콘솔을 사용하여 로그 그룹에서 이상을 스캔하는 로그 이상 탐지기를 만들 수 있습니다.

프로그래밍 방식으로 이상 탐지기를 만들 수도 있습니다. 자세한 내용은 을 참조하십시오. CreateLogAnomalyDetector

로그 이상 탐지기 생성하기

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.

2. 로그, 로그 예외 항목을 선택합니다.
3. 예외 항목 탐지기 생성을 선택합니다.
4. 이 이상 탐지기를 생성할 로그 그룹을 선택합니다.
5. 예외 항목 탐지기 이름에 탐지기 이름을 입력합니다.
6. (선택 사항) 평가 빈도를 기본값인 5분에서 변경합니다. 로그 그룹이 새 로그를 수신하는 빈도에 따라 이 값을 설정합니다. 예를 들어, 로그 그룹이 10분마다 일괄적으로 새 로그 이벤트를 수신하는 경우 평가 빈도를 15분으로 설정하는 것이 적절할 수 있습니다.
7. (선택 사항) 특정 단어나 문자열이 포함된 로그 이벤트에서만 이상을 찾도록 예외 항목 탐지기를 구성하려면 필터 패턴을 선택합니다.

   그런 다음 예외 항목 탐지 필터 패턴에 패턴을 입력합니다. 패턴 구문에 대한 자세한 내용은 지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail에 대한 필터 패턴 구문

   (선택 사항) 필터 패턴을 테스트하려면 로그 이벤트 메시지에 일부 로그 메시지를 입력한 다음 [Test Pattern] 을 선택합니다.
8. (선택 사항) 예외 항목 표시 기간을 기본값에서 변경하거나 이 예외 항목 탐지기에 AWS KMS 키를 연결하려면 고급 구성을 선택합니다.

   a. 예외 항목 가시성 기간을 기본값에서 변경하려면 최대 예외 항목 가시성 기간 (일) 에 새 값을 입력합니다.

   b. AWS KMS 키를 이 이상 탐지기와 연결하려면 KMS 키 ARN에 ARN을 입력합니다. 키를 할당하면 이 탐지기에서 찾은 이상 정보가 키와 함께 암호화됩니다. 사용자는 이 키에 대한 권한을 가지고 있어야 하며, 이상 탐지기가 발견한 이상 징후와 관련된 정보를 검색할 수 있는 권한이 있어야 합니다.

      또한 CloudWatch 로그 서비스 주체에게 키 사용 권한이 있는지 확인해야 합니다. 자세한 정보는 이상 탐지기 및 그 결과를 다음과 같이 암호화합니다. AWS KMS을 참조하세요.
9. 예외 항목 탐지 활성화를 선택합니다.

   예외 항목 탐지기가 생성되고 로그 그룹이 수집하는 로그 이벤트를 기반으로 모델 학습을 시작합니다. 약 15분 후 이상 탐지가 활성화되어 이상 징후를 찾아 표면화하기 시작합니다.

발견된 이상 현상 보기

로그 예외 항목 탐지기를 하나 이상 만든 후 CloudWatch 콘솔을 사용하여 탐지기가 발견한 이상 징후를 볼 수 있습니다.

프로그래밍 방식으로 이상 현상을 볼 수 있습니다. 자세한 내용은 을 참조하십시오. ListAnomalies

모든 로그 이상 탐지기에서 발견된 이상 징후를 보려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 콘솔을 엽니다. CloudWatch
2. 로그, 로그 예외 항목을 선택합니다.

   로그 예외 테이블이 나타납니다. 로그 예외 항목 옆의 상단에 있는 숫자는 표에 나열된 로그 예외 항목의 수를 나타냅니다. 표의 각 행에는 다음 정보가 표시됩니다.

   - 예외 항목 열에는 예외 항목에 대한 간략한 요약이 표시됩니다. 이러한 요약은 로그에서 생성됩니다. CloudWatch
   - 예외 항목의 우선순위. 우선 순위는 로그 이벤트의 변화량, 로그 이벤트에서 `Exception` 발생하는 것과 같은 키워드 등에 따라 자동으로 계산됩니다.
   - 예외 현상의 기반이 되는 로그 패턴. 패턴에 대한 자세한 내용은 을 참조하십시오. 로그 이상 탐지
   - 예외 로그 트렌드는 패턴과 일치하는 로그의 양을 나타내는 히스토그램을 표시합니다.
   - 마지막 탐지 시간은 이 예외가 발견된 가장 최근 시간을 표시합니다.
   - 최초 탐지 시간은 이 예외 항목이 처음 발견된 시간을 표시합니다.
   - 예외 항목 탐지기는 이 예외 항목과 관련된 로그 이벤트가 포함된 로그 그룹의 이름을 표시합니다. 이 이름을 선택하면 로그 그룹 세부 정보 페이지를 볼 수 있습니다.
3. 예외 항목 하나를 더 자세히 검사하려면 해당 행의 라디오 버튼을 선택하십시오.

   패턴 검사 창이 나타나고 다음이 표시됩니다.

   - 이 예외 현상의 기반이 되는 패턴입니다. 패턴 내에서 토큰을 선택하여 해당 토큰의 값을 분석하십시오.
   - 쿼리된 시간 범위 동안의 예외 항목 발생 수를 보여주는 히스토그램입니다.
   - 로그 샘플 탭에는 예외 현상에 속하는 몇 가지 로그 이벤트가 표시됩니다.
   - 토큰 값 탭에는 선택한 동적 토큰의 값이 표시됩니다 (선택한 경우).

   > **Note**
   >
   > 각 토큰에 대해 최대 10개의 토큰 값이 캡처됩니다. 토큰 수가 정확하지 않을 수 있습니다. CloudWatch 로그는 절대값이 아닌 확률적 카운터를 사용하여 토큰 수를 생성합니다.

4. 예외 항목을 숨기려면 해당 행에서 라디오 버튼을 선택한 후 다음을 수행하십시오.

   a. 조치, 예외 항목 억제를 선택합니다.

   b. 그런 다음 예외 항목을 억제할 기간을 지정합니다.

   c. 이 패턴과 관련된 모든 예외를 억제하려면 패턴 억제를 선택합니다.

   d. [예외 항목 억제] 를 선택합니다.

단일 로그 그룹에서 발견된 예외 항목을 보려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 로그, 로그 그룹을 선택합니다.
3. 로그 그룹 이름을 선택한 다음 예외 항목 탐지 탭을 선택합니다.

   예외 항목 탐지 테이블이 나타납니다. 로그 예외 항목 옆의 상단에 있는 숫자는 표에 나열된 로그 예외 항목의 수를 나타냅니다. 표의 각 행에는 다음 정보가 표시됩니다.

   - 예외 항목 열에는 예외 항목에 대한 간략한 요약이 표시됩니다. 이러한 요약은 로그에서 생성됩니다. CloudWatch
   - 예외 항목의 우선순위. 우선 순위는 로그 이벤트의 변화량, 로그 이벤트에서 `Exception` 발생하는 것과 같은 키워드 등에 따라 자동으로 계산됩니다.
   - 예외 현상의 기반이 되는 로그 패턴. 패턴에 대한 자세한 내용은 을 참조하십시오. 로그 이상 탐지
   - 예외 로그 트렌드는 패턴과 일치하는 로그의 양을 나타내는 히스토그램을 표시합니다.
   - 마지막 탐지 시간은 이 예외가 발견된 가장 최근 시간을 표시합니다.
   - 최초 탐지 시간은 이 예외 항목이 처음 발견된 시간을 표시합니다.
4. 예외 항목 하나를 더 자세히 조사하려면 해당 행에서 라디오 버튼을 선택합니다.

   패턴 검사 창이 나타나고 다음이 표시됩니다.

   - 이 예외 현상의 기반이 되는 패턴입니다. 패턴 내에서 토큰을 선택하여 해당 토큰의 값을 분석하십시오.
   - 쿼리된 시간 범위 동안의 예외 항목 발생 수를 보여주는 히스토그램입니다.
   - 로그 샘플 탭에는 예외 현상에 속하는 몇 가지 로그 이벤트가 표시됩니다.
   - 토큰 값 탭에는 선택한 동적 토큰의 값이 표시됩니다 (선택한 경우).

> **Note**
>
> 각 토큰에 대해 최대 10개의 토큰 값이 캡처됩니다. 토큰 수가 정확하지 않을 수 있습니다. CloudWatch 로그는 절대값이 아닌 확률적 카운터를 사용하여 토큰 수를 생성합니다.

5. 예외 항목을 숨기려면 해당 행에서 라디오 버튼을 선택한 후 다음을 수행하십시오.

   a. 조치, 예외 항목 억제를 선택합니다.

   b. 그런 다음 예외 항목을 억제할 기간을 지정합니다.

   c. 이 패턴과 관련된 모든 예외를 억제하려면 패턴 억제를 선택합니다.

   d. [예외 항목 억제] 를 선택합니다.

로그 이상 탐지기에 대한 경보 생성

로그 그룹의 로그 이상 탐지기에 대한 경보를 생성할 수 있습니다. 지정된 기간 동안 로그 그룹에서 지정된 수의 예외가 발견되면 경보가 ALARM 상태로 전환되도록 지정할 수 있습니다. 필터를 사용하여 지정된 우선 순위의 예외 항목만 경보에서 계산되도록 할 수도 있습니다.

로그 이상 탐지기에 대한 경보를 만들려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그, 로그 예외 항목을 선택합니다.

   로그 이상 탐지기 테이블이 나타납니다.
3. 경보를 설정하려는 예외 항목 탐지기의 라디오 버튼을 선택하고 Create alarm (Create alarm) 을 선택합니다.

   CloudWatch 알람 생성 마법사가 나타납니다. 이 LogAnomalyDetector필드에는 선택한 이상 탐지기 이름이 표시됩니다. 지표 이름 필드가 표시됩니다. AnomalyCount
4. (선택 사항) 이 경보를 예외 항목 우선 순위로 필터링하려면 다음 중 하나를 수행하십시오.
   - 경보가 우선순위가 높은 예외 항목만 계산하도록 하려면 for를 입력합니다. **HIGH** LogAnomalyPriority
   - 우선 순위가 높거나 중간 정도인 예외 항목만 경보로 계산하도록 하려면 for를 입력합니다. **MEDIUM** LogAnomalyPriority

우선 순위 수준에 대한 자세한 내용은 을 참조하십시오. 이상 및 패턴의 심각도 및 우선 순위

5. 경보에 정적 또는 지표 이상 탐지 임계값을 사용하도록 선택하십시오. 이 선택에 따라 경보 임계값 설정 방법이 결정됩니다. 정적 임계값은 알람 임계값이 사용자가 선택한 고정적이고 일정한 수치임을 의미합니다. 예외 항목 탐지 임계값은 일반적인 값의 범위를 CloudWatch 결정하는 것을 의미하며, 실제 수가 이 대역의 임계값을 초과하면 경보가 트리거됩니다. 로그 이상 탐지 경보에는 예외 항목 감지를 선택할 필요가 없습니다. 지표 이상 탐지에 대한 자세한 내용은 예외 항목 탐지 사용을 참조하십시오. CloudWatch

6. For Whenever is ***your-metric-name***. [크게], [크게/같음], [낮음/같음] 또는 [낮음] 을 선택합니다. :에 임곗값에 대한 숫자를 지정합니다. 기간으로 지정된 시간 동안 예외 항목 탐지기가 이 수보다 많은 경보를 발견하면 경보 **ALARM** 상태가 됩니다.

7. 추가 구성을 선택합니다. 경보에 대한 데이터 포인트에서 경보를 트리거하기 위해 평가 기간(데이터 포인트)이 ALARM 상태로 유지해야 하는 기간을 지정합니다. 두 값이 일치하는 경우 다수의 연속 기간이 위반되면 ALARM 상태가 되는 경보가 생성됩니다.

   N개 중 M번째 경보를 생성하려면 두 번째 값의 숫자보다 작은 값을 첫 번째 값에 지정합니다. 자세한 내용은 경보 평가를 참조하십시오.

8. 누락 데이터 처리(Missing data treatment)에서 일부 데이터 포인트가 누락된 경우 경보가 어떻게 동작할지 선택합니다. 자세한 내용은 CloudWatch 경보가 누락된 데이터를 처리하는 방법 구성을 참조하십시오.

9. 다음을 선택합니다.

10. 알림에서 알림 추가를 선택한 다음, 경보가 ALARMOK, 또는 INSUFFICIENT_DATA 상태로 전환될 때 알리도록 Amazon SNS 주제를 지정합니다.

    a. (선택 사항) 동일한 경보 상태 또는 다른 경보 상태에 대해 여러 개의 알림을 보내려면 Add notification(알림 추가)을 선택합니다.

    > **Note**
    >
    > 경보 상태가 되는 경우와 함께 데이터 부족 상태가 되는 경우에도 조치를 취하도록 경보를 설정하는 것이 좋습니다. 데이터 소스에 연결되는 Lambda 함수와 관련된 많은 문제로 인해 경보가 데이터 부족 상태로 전환될 수 있기 때문입니다.

    b. (선택 사항) Amazon SNS 알림을 보내지 않으려면 제거를 선택합니다.

11. (선택 사항) 경보가 Amazon EC2 Auto Scaling, Amazon EC2, AWS Systems Manager티켓 등에 대한 작업을 수행하도록 하려면 적절한 버튼을 선택하고 경보 상태 및 조치를 지정하십시오.

> **Note**
>
> 경보는 ALARM 상태일 때만 Systems Manager 작업을 수행할 수 있습니다. Systems Manager 작업에 대한 자세한 내용은 생성 구성 CloudWatch OpsItems 및 인시던트 생성을 참조하십시오.

12. 다음을 선택합니다.
13. 이름 및 설명에서 경보의 이름과 설명을 입력하고 다음을 선택합니다. 이름에는 UTF-8 문자만 포함해야 하며 ASCII 제어 문자는 포함할 수 없습니다. 설명에는 마크다운 형식이 포함될 수 있으며, 마크다운 형식은 CloudWatch 콘솔의 알람 세부 정보 탭에만 표시됩니다. 마크다운은 런북이나 기타 내부 리소스에 대한 링크를 추가하는 데 유용할 수 있습니다.

> **Tip**
>
> 경보 이름에는 UTF-8 문자만 포함되어야 합니다. ASCII 제어 문자를 포함할 수 없습니다.

14. 미리 보기 및 생성에서 정보 및 조건이 원하는 내용인지 확인한 다음 경보 생성을 선택합니다.

로그 이상 탐지기가 게시한 지표

CloudWatch 로그는 지표를 지표에 게시합니다. AnomalyCount CloudWatch 이 지표는 AWS/Logs 네임스페이스에 게시됩니다.

AnomalyCount지표는 다음 차원으로 게시됩니다.

- LogAnomalyDetector— 이상 탐지기 이름
- LogAnomalyPriority— 이상 현상의 우선 순위

이상 탐지기 및 그 결과를 다음과 같이 암호화합니다. AWS KMS

이상 탐지기 데이터는 항상 로그에서 암호화됩니다. CloudWatch 기본적으로 CloudWatch Logs는 저장된 데이터에 대해 서버 측 암호화를 사용합니다. 대신 AWS Key Management Service 를 사용하여 이를 암호화할 수 있습니다. 이렇게 하면 키를 사용하여 암호화가 수행됩니다. AWS KMS KMS 키를 이상 탐지기와 연결하면 예외 항목 탐지기 수준에서 암호화를 사용할 AWS KMS 수 있습니다.

> **Important**
>
> CloudWatch 로그는 대칭 KMS 키만 지원합니다. 비대칭 키를 사용하여 로그 그룹의 데이터를 암호화하지 마세요. 자세한 내용은 대칭 및 비대칭 키 사용을 참조하세요.

Limits

- 다음 단계를 수행하려면 `kms:CreateKey`, `kms:GetKeyPolicy` 및 `kms:PutKeyPolicy` 권한이 있어야 합니다.
- 이상 탐지기에 키를 연결하거나 연결을 끊은 후 작업이 적용되는 데 최대 5분이 걸릴 수 있습니다.
- 관련 키에 대한 CloudWatch 로그 액세스를 취소하거나 연결된 KMS 키를 삭제하면 Logs의 암호화된 데이터를 더 이상 검색할 수 없습니다. CloudWatch

1단계: 키 생성 AWS KMS

다음 create-key 명령을 사용하여 KMS 키 생성:

```
aws kms create-key
```

이 명령의 출력 화면에는 키의 키 ID와 Amazon 리소스 이름(ARN)이 포함됩니다. 다음은 예 출력입니다.

```
{
    "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"
        ]
```

```
    }
}
```

2단계: KMS 키에 대한 권한 설정

기본적으로 모든 AWS KMS 키는 비공개입니다. 리소스 소유자만 이를 사용하여 데이터를 암호화 및 해독할 수 있습니다. 그러나 리소스 소유자가 원한다면 다른 사용자 및 리소스에 KMS 키에 대한 액세스 권한을 부여할 수 있습니다. 이 단계에서는 CloudWatch 로그 서비스 주체에게 키 사용 권한을 부여합니다. 이 서비스 보안 주체는 KMS 키가 저장된 동일한 AWS 지역에 있어야 합니다.

가장 좋은 방법은 KMS 키 사용을 지정한 AWS 계정이나 예외 항목 탐지기로만 제한하는 것이 좋습니다.

먼저 다음 명령을 사용하여 KMS 키의 기본 정책을 저장합니다. `policy.json` get-key-policy

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

텍스트 편집기에서 `policy.json` 파일을 열고 다음 설명 중 하나에서 굵은 글꼴로 표시된 섹션을 추가합니다. 기존 설명과 새 설명을 쉼표로 구분합니다. 이러한 명령문은 `Condition` 섹션을 사용하여 AWS KMS 키의 보안을 강화합니다. 자세한 정보는 AWS KMS 키 및 암호화 컨텍스트을 참조하세요.

이 예제의 `Condition` 섹션에서는 AWS KMS 키 사용을 지정된 계정으로만 제한하지만 모든 이상 탐지에 사용할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::Your_account_ID:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": "logs.REGION.amazonaws.com"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*",
      "Condition": {
        "ArnLike": {
          "kms:EncryptionContext:aws:logs:arn":
 "arn:aws:logs:REGION:Your_account_ID:anomaly-detector:*"
        }
      }
    },
    {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.REGION.amazonaws.com"
        },
        "Action": [
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*",
            "kms:DescribeKey"
        ],
        "Resource": "*",
        "Condition": {
            "ArnLike": {
                "kms:EncryptionContext:aws-crypto-ec:aws:logs:arn":
 "arn:aws:logs:REGION:Your_account_ID:anomaly-detector:*"
            }
        }
    }
  ]
}
```

마지막으로 다음 put-key-policy명령을 사용하여 업데이트된 정책을 추가합니다.

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

3단계: KMS 키를 예외 항목 탐지기와 연결

콘솔에서 또는 API를 사용하여 KMS 키를 만들 때 이상 탐지기에 KMS 키를 연결할 수 있습니다. AWS CLI

4단계: 이상 탐지기에서 키 연결 해제

키가 이상 탐지기와 연결된 후에는 키를 업데이트할 수 없습니다. 키를 제거하는 유일한 방법은 예외 항목 탐지기를 삭제한 다음 다시 생성하는 것입니다.

로그 그룹 및 로그 스트림 작업

로그 스트림은 동일한 소스를 공유하는 로그 이벤트 시퀀스입니다. 로그의 개별 로그 소스는 각각 별도의 로그 스트림을 구성합니다. CloudWatch

로그 그룹은 동일한 보존 기간, 모니터링 및 액세스 제어 설정을 공유하는 로그 스트림 그룹입니다. 로그 그룹을 정의하고 각 그룹에 배치할 스트림을 지정할 수 있습니다. 하나의 로그 그룹이 가질 수 있는 로그 스트림의 수는 제한이 없습니다.

이 단원에 나오는 절차를 사용하여 로그 그룹 및 로그 스트림 작업을 수행합니다.

CloudWatch Logs에서 로그 그룹을 생성합니다.

Amazon CloudWatch Logs 사용 설명서의 이전 섹션에 있는 단계를 사용하여 Amazon EC2 인스턴스에 CloudWatch Logs 에이전트를 설치하면 해당 프로세스의 일부로 로그 그룹이 생성됩니다. CloudWatch 콘솔에서 직접 로그 그룹을 생성할 수도 있습니다.

사용자 그룹을 생성하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 작업을 선택한 후 로그 그룹 생성을 선택합니다.
4. 로그 그룹의 이름을 입력한 다음 로그 그룹 생성을 선택합니다.

> **Tip**
>
> 탐색 창의 즐겨찾기 및 최근 항목(Favorites and recents) 메뉴에서 대시보드와 경보뿐만 아니라 로그 그룹을 즐겨찾기에 추가할 수 있습니다. 최근 방문(Recently visited) 열에서 즐겨찾기에 추가하려는 로그 그룹 위에 마우스 포인터를 놓고 그 옆에 있는 별 기호를 선택합니다.

로그 그룹에 로그 보내기

CloudWatch Logs는 여러 AWS 서비스로부터 로그 이벤트를 자동으로 수신합니다. 다음 방법 중 하나를 사용하여 다른 로그 이벤트를 CloudWatch Logs로 보낼 수도 있습니다.

- CloudWatch 에이전트 - 통합 CloudWatch 에이전트는 지표와 로그를 모두 Logs에 CloudWatch 보낼 수 있습니다. CloudWatch 에이전트 설치 및 사용에 대한 자세한 내용은 Amazon 사용 설명서의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집을 참조하십시오. CloudWatch
- AWS CLI—로그 이벤트를 Logs에 일괄 put-log-events업로드합니다. CloudWatch
- 프로그래밍 방식 — PutLogEventsAPI를 사용하면 프로그래밍 방식으로 로그 이벤트 배치를 로그에 업로드할 수 있습니다. CloudWatch

Logs로 전송된 로그 데이터 보기 CloudWatch

Logs 에이전트가 Logs로 전송한 내용을 stream-by-stream 기준으로 CloudWatch 로그 데이터를 보고 스크롤할 수 있습니다. CloudWatch 확인할 로그 데이터에 대해 시간 범위를 지정할 수 있습니다.

로그 데이터를 보려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹에서 스트림을 확인할 로그 그룹을 선택합니다.
4. 로그 그룹 목록에서 보려는 로그 그룹의 이름을 선택합니다.
5. 로그 스트림 목록에서 보려는 로그 스트림의 이름을 선택합니다.
6. 로그 데이터의 표시 방법을 변경하려면 다음 중 하나를 수행합니다.

   - 단일 로그 이벤트를 확장하려면 해당 로그 이벤트 옆에 있는 화살표를 선택합니다.
   - 로그 이벤트 목록에서 모든 로그 이벤트를 확장하고 이를 일반 텍스트 버전으로 보려면 텍스트를 선택합니다.
   - 로그 이벤트를 필터링하려면 검색 필드에 원하는 검색 필터를 입력합니다. 자세한 정보는 필터를 사용하여 로그 이벤트에서 지표 생성 섹션을 참조하세요.
   - 지정된 날짜 및 시간 범위의 로그 데이터를 보려면 검색 필터 옆의 날짜 및 시간 옆에 있는 화살표를 선택합니다. 날짜 및 시간 범위를 지정하려면 절대를 선택합니다. 미리 정의된 분, 시간, 일 또는 주 수를 선택하려면 상대를 선택합니다. UTC와 현지 시간대 간을 전환할 수도 있습니다.

Live Tail을 사용하여 거의 실시간으로 로그 보기

CloudWatch Logs Live Tail을 사용하면 수집되는 새 로그 이벤트의 스트리밍 목록을 확인하여 인시던트를 신속하게 해결할 수 있습니다. 모은 로그를 거의 실시간으로 보고 필터링하고 강조 표시할 수 있

으므로 문제를 빠르게 탐지하고 해결할 수 있습니다. 지정한 항을 기반으로 로그를 필터링하고 원하는 항목을 빠르게 찾을 수 있도록 지정된 항이 포함된 로그를 강조 표시할 수도 있습니다.

Live Tail 세션은 세션 사용 시간(분)별로 비용이 발생합니다. 요금에 대한 자세한 내용은 Amazon CloudWatch 요금의 로그 탭을 참조하십시오.

> **Note**
>
> Live Tail은 표준 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

다음 섹션에서는 콘솔에서 Live Tail을 사용하는 방법을 설명합니다. 프로그래밍 방식으로 라이브 테일 세션을 시작할 수도 있습니다. 자세한 내용은 StartLiveTail를 참조하세요. SDK 예제는 SDK를 사용하여 라이브 테일 세션 시작을 참조하십시오. AWS

Live Tail 세션 시작

CloudWatch 콘솔을 사용하여 Live Tail 세션을 시작할 수 있습니다. 다음 절차에서는 왼쪽 탐색 창의 Live tail 옵션을 사용하여 Live Tail 세션을 시작하는 방법을 설명합니다. 로그 그룹 페이지 또는 CloudWatch 로그 인사이트 페이지에서 라이브 테일 세션을 시작할 수도 있습니다.

> **Note**
>
> 데이터 보호 정책을 사용하여 Live Tail로 보고 있는 로그 그룹의 민감한 데이터를 마스킹하는 경우 민감한 데이터는 항상 Live Tail 세션에서 마스킹된 상태로 나타납니다. 로그 그룹의 민감한 데이터 마스킹에 대한 자세한 내용은 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원 섹션을 참조하세요.

Live Tail 세션 시작

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그, Live tail을 선택합니다.
3. 로그 그룹 선택에서 Live Tail 세션에서 이벤트를 보려는 로그 그룹을 선택합니다. 로그 그룹을 10개까지 선택할 수 있습니다.
4. (선택 사항) 로그 그룹을 하나만 선택한 경우 로그 이벤트를 보려는 로그 스트림을 하나 이상 선택하여 Live Tail 세션을 추가로 필터링할 수 있습니다. 이렇게 하려면 로그 스트림 선택 아래의 드롭

다운 목록에서 로그 스트림 이름을 선택합니다. 또는 로그 스트림 선택 아래의 두 번째 상자를 사용하여 로그 스트림 이름 접두사를 입력할 수 있습니다. 그러면 접두사와 일치하는 이름의 모든 로그 스트림이 선택됩니다.

5. (선택 사항) 특정 단어나 다른 문자열을 포함하는 로그 이벤트만 표시하려면 `Add filter patterns`에 단어나 문자열을 입력합니다.

   예를 들어 `Warning`이라는 단어가 포함된 로그 이벤트만 표시하려면 **`Warning`**을 입력합니다. 필터 필드는 대소문자를 구분합니다. 이 필드에 여러 항과 패턴 연산자를 포함할 수 있습니다.

   - **`error 404`**는 `error`와 `404`를 모두 포함하는 로그 이벤트만 표시합니다.
   - **`?Error ?error`**는 `Error` 또는 `error`를 포함하는 로그 이벤트를 표시합니다.
   - **`-INFO`**는 `INFO`를 포함하지 않는 모든 로그 이벤트를 표시합니다.
   - **`{ $.eventType = "UpdateTrail" }`**은 이벤트 유형 필드의 값이 `UpdateTrail`인 모든 JSON 로그 이벤트를 표시합니다.

   정규식(regex)을 사용하여 필터링할 수도 있습니다.

   - **`%ERROR%`**은(는) regex를 사용하여 ERROR 키워드로 구성된 모든 로그 이벤트를 표시
   - **`{ $.names = %Steve% }`**은(는) Steve가 속성 `"name"`에 속해 있는 경우 regex를 사용하여 JSON 로그 이벤트를 표시
   - **`[ w1 = %abc%, w2 ]`**은(는) regex를 사용하여 첫 단어가 abc인 공백으로 구분된 로그 이벤트를 표시

   패턴 구문에 대한 자세한 내용은 필터 패턴 구문을 참조하세요.

6. (선택 사항) 표시된 로그 이벤트 중 일부를 강조 표시하려면 Live Tail 아래에 검색하고 강조 표시할 항을 입력합니다. 강조 표시 항을 한 번에 하나씩 입력합니다. 강조 표시할 항을 여러 개 추가하면 각 항을 나타내는 다른 색상이 할당됩니다. 강조 표시기는 지정된 항이 포함된 로그 이벤트의 왼쪽에 표시되며, 전체 로그 이벤트를 보기 위해 기본 창에서 로그 이벤트를 확장할 때도 항 자체 아래에 표시됩니다.

   강조 표시와 함께 필터링을 사용하여 문제를 신속하게 해결할 수 있습니다. 예를 들어 이벤트를 필터링하여 `Error`가 포함된 이벤트만 표시한 다음 `404`가 포함된 이벤트를 강조 표시할 수도 있습니다.

7. 세션을 시작하려면 필터 적용을 선택합니다.

   일치하는 로그 이벤트가 창에 나타나기 시작합니다. 다음 정보도 표시됩니다.

- 타이머는 Live Tail 세션이 활성화된 시간을 표시합니다.
- 이벤트/초는 설정한 필터와 일치하는 초당 모은 로그 이벤트 수를 표시합니다.
- 많은 이벤트가 필터와 일치하여 세션이 너무 빨리 스크롤되는 것을 방지하기 위해 CloudWatch 로그에는 일치하는 일부 이벤트만 표시될 수 있습니다. 이 경우 화면에 표시되는 일치하는 이벤트의 비율이 % 표시됨에 표시됩니다.

8. 현재 표시된 내용을 조사하기 위해 이벤트 흐름을 일시 중지하려면 이벤트 창의 아무 곳이나 클릭합니다.
9. 세션 중 다음을 사용하여 각 로그 이벤트에 대한 추가 세부 정보를 볼 수 있습니다.

    - 기본 창에 로그 이벤트의 전체 텍스트를 표시하려면 해당 로그 이벤트 옆에 있는 화살표를 선택합니다.
    - 보조 창에 로그 이벤트의 전체 텍스트를 표시하려면 해당 로그 이벤트 옆에 있는 + 돋보기를 선택합니다. 이벤트 흐름이 일시 중지되고 보조 창이 나타납니다.

      보조 창에 로그 이벤트 텍스트를 표시하면 해당 텍스트를 기본 창의 다른 이벤트와 비교하는 데 유용할 수 있습니다.

10. Live Tail 세션을 중지하려면 중지를 선택합니다.
11. 세션을 다시 시작하려면 필요에 따라 필터 패널을 사용하여 필터링 기준을 수정하고 필터 적용을 선택합니다. 그런 다음 시작을 선택합니다.

필터 패턴을 사용하여 로그 데이터 검색

지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail에 대한 필터 패턴 구문를 사용하여 로그 데이터를 검색할 수 있습니다. 로그 그룹 내의 모든 로그 스트림을 검색하거나 를 사용하여 특정 로그 스트림을 AWS CLI 검색할 수도 있습니다. 각각의 검색이 실행되면서 데이터의 다음 페이지를 검색하거나 검색을 계속할 수 있도록 발견된 데이터의 첫 페이지와 토큰이 반환됩니다. 결과가 반환되지 않은 경우에는 계속 검색을 할 수 있습니다.

검색 범위를 제한하기 위해 쿼리하고자 하는 시간 범위를 설정할 수 있습니다. 처음에는 시간 범위를 넓게 잡아서 관심 있는 로그 줄이 어디에 있는지 확인한 다음, 시간 범위를 좁혀서 관심 있는 시간 범위로 로그에 대한 뷰의 범위를 지정할 수 있습니다.

로그에서 추출된 지표에서 해당 로그로 직접 피벗을 적용할 수도 있습니다.

CloudWatch Cross-Account Observability에서 모니터링 계정으로 설정된 계정에 로그인한 경우 이 모니터링 계정에 연결된 소스 계정의 로그 이벤트를 검색하고 필터링할 수 있습니다. 자세한 내용은 계정 CloudWatch 간 옵저버빌리티를 참조하십시오.

콘솔을 사용하여 로그 항목 검색

콘솔을 이용하여 지정된 기준을 충족하는 로그 항목을 검색할 수 있습니다.

콘솔을 이용하여 로그 항목을 검색하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹에서 검색할 로그 스트림이 포함된 로그 그룹의 이름을 선택합니다.
4. 로그 스트림에서 검색할 로그 스트림의 이름을 선택합니다.
5. 로그 이벤트에서 사용할 필터 구문을 입력합니다.

콘솔을 이용하여 시간 범위에 대한 모든 로그 항목을 검색하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹에서 검색할 로그 스트림이 포함된 로그 그룹의 이름을 선택합니다.
4. 로그 그룹 검색을 선택합니다.
5. 로그 이벤트에서 날짜 및 시간 범위를 선택하고 필터 구문을 입력합니다.

를 사용하여 로그 항목을 검색합니다. AWS CLI

를 사용하여 지정된 기준에 맞는 로그 항목을 검색할 수 AWS CLI있습니다.

를 사용하여 로그 항목을 검색하려면 AWS CLI

명령 프롬프트에서 다음 filter-log-events명령을 실행합니다. `--filter-pattern`을 사용하여 지정된 필터 패턴으로 결과를 제한하고 `--log-stream-names`을 사용하여 지정된 로그 스트림으로 결과를 제한합니다.

```
aws logs filter-log-events --log-group-name my-group [--log-stream-
names LIST_OF_STREAMS_TO_SEARCH] [--filter-pattern VALID_METRIC_FILTER_PATTERN]
```

를 사용하여 지정된 시간 범위의 로그 항목을 검색하려면 AWS CLI

명령 프롬프트에서 다음 filter-log-events명령을 실행합니다.

```
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]
```

지표에서 로그로 피벗 적용

콘솔의 다른 부분에서 특정 로그 항목으로 이동할 수 있습니다.

대시보드 위젯에서 로그로 이동하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 대시보드를 선택합니다.
3. 대시보드를 선택합니다.
4. 위젯에서 로그 보기 아이콘을 선택한 다음 이 시간 범위에서 로그 보기를 선택합니다. 지표 필터가 하나 이상 있는 경우 목록에서 하나를 선택합니다. 지표 필터의 수가 목록에 표시할 수 있는 것보다 많은 경우에는 더 많은 지표 필터를 선택하고 지표 필터를 선택 또는 검색합니다.

지표에서 로그로 이동하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표(Metrics)를 선택합니다.
3. 모든 지표 탭의 검색 필드에 지표 이름을 입력하고 Enter 키를 누릅니다.
4. 검색 결과에서 지표를 하나 이상 선택합니다.
5. 작업과 로그 보기를 선택합니다. 지표 필터가 하나 이상 있는 경우 목록에서 하나를 선택합니다. 지표 필터의 수가 목록에 표시할 수 있는 것보다 많은 경우에는 더 많은 지표 필터를 선택하고 지표 필터를 선택 또는 검색합니다.

문제 해결

검색 완료에 너무 많은 시간이 소요

로그 데이터가 많은 경우에는 검색을 완료하는 데 긴 시간이 소요될 수 있습니다. 다음과 같은 방법으로 검색 속도를 높일 수 있습니다.

- 를 사용하는 AWS CLI경우 원하는 로그 스트림으로만 검색을 제한할 수 있습니다. 예를 들어, 로그 그룹에 1000개의 로그 스트림이 있는데 관련성이 있다고 알고 있는 세 개의 로그 스트림만 보려는 경우 를 사용하여 로그 그룹 내의 로그 스트림 3개로만 검색을 제한할 수 있습니다. AWS CLI
- 더 짧고 세분화된 시간 범위를 사용하면 검색할 데이터양을 줄이고 쿼리 속도를 높일 수 있습니다.

로그에서 CloudWatch 로그 데이터 보존을 변경하십시오.

기본적으로 로그 데이터는 CloudWatch Logs에 무기한 저장됩니다. 하지만 로그 그룹에서 로그 데이터를 저장할 기간을 구성할 수 있습니다. 현재 보존 기간 설정보다 오래된 데이터는 모두 삭제됩니다. 언제든지 로그 그룹별로 로그 보존 기간을 변경할 수 있습니다.

> **Note**
>
> CloudWatch Logs 보존 설정에 도달해도 로그 이벤트를 즉시 삭제하지 않습니다. 일반적으로 로그 이벤트가 삭제되기까지 최대 72시간이 걸리지만 드물게 그보다 더 오래 걸릴 수 있습니다.
> 즉, 만료 날짜가 지났지만 실제로 삭제되지 않은 로그 이벤트를 포함하는 경우 더 긴 보존 설정을 갖도록 로그 그룹을 변경하면 새 보존 날짜에 도달한 후 해당 로그 이벤트가 삭제되는 데 최대 72시간이 걸립니다. 로그 데이터를 영구적으로 삭제하려면 이전 보존 기간이 종료된 후 72시간이 경과하거나 이전 로그 이벤트가 삭제되었음을 확인할 때까지 로그 그룹을 낮은 보존 설정으로 유지합니다.
> 로그 이벤트가 보존 설정에 도달하면 삭제 대상으로 표시됩니다. 삭제 대상으로 표시된 후에는 이후에 실제로 삭제되지 않더라도 더 이상 아카이브 스토리지 비용에 추가되지 않습니다. 삭제 대상으로 표시된 로그 이벤트는 API를 통해 `storedBytes` 값을 검색하여 로그 그룹에 저장 중인 바이트 수를 확인할 때도 포함되지 않습니다.

로그 보존 기간 설정을 변경하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 로그, 로그 그룹을 선택합니다.
3. 업데이트할 로그 그룹을 찾습니다.
4. 해당 로그 그룹의 보존 열에서 현재 보존 설정 (예: 만료 안 함) 을 선택합니다.
5. 보존 설정에서 만료 이후 이벤트에 대해 로그 보존 값을 선택한 다음 저장을 선택합니다.

Amazon Logs의 태그 CloudWatch 로그 그룹

Amazon CloudWatch Logs에서 생성한 로그 그룹에 태그 형태로 자체 메타데이터를 할당할 수 있습니다. 태그는 로그 그룹에 대해 정의된 키-값 페어입니다. 태그를 사용하면 AWS 리소스를 관리하고 청구 데이터를 비롯한 데이터를 정리할 수 있는 간단하면서도 강력한 방법이 됩니다.

> **Note**
>
> 태그를 사용하여 로그 그룹 및 대상을 포함한 CloudWatch 로그 리소스에 대한 액세스를 제어할 수 있습니다. 로그 그룹과 로그 스트림 간의 계층적 관계로 인해 로그 스트림에 대한 액세스는 로그 그룹 수준에서 제어됩니다. 태그를 사용하여 액세스를 제어하는 방법에 대한 자세한 내용은 태그를 사용하여 Amazon Web Services 리소스에 대한 액세스 제어를 참조하세요.

내용

태그 기본 사항

AWS CLI, 또는 CloudWatch Logs API를 사용하여 AWS CloudFormation 다음 작업을 완료할 수 있습니다.

- 생성 시 로그 그룹에 태그를 추가합니다.
- 기존 로그 그룹에 태그를 추가합니다.
- 로그 그룹에 대한 태그를 나열합니다.
- 로그 그룹에서 태그를 제거합니다.

태그를 사용하여 로그 그룹을 분류할 수 있습니다. 예를 들어 용도, 소유자 또는 환경을 기준별로 로그 그룹을 분류할 수 있습니다. 각 태그에 대해 키와 값이 정의되기 때문에 특정 요구를 충족하는 사용자 지정 범주 세트를 생성할 수 있습니다. 예를 들어, 태그 세트를 정의하여 소유자와 연관 애플리케이션에 따라 로그 그룹을 추적할 수 있습니다. 다음은 태그의 몇 가지 예제입니다.

- 프로젝트: 프로젝트 이름
- 소유자: 이름
- 용도: 로드 테스트
- 애플리케이션: 애플리케이션 이름
- 환경: 프로덕션

태그 지정을 사용하여 비용 추적

태그를 사용하여 AWS 비용을 분류하고 추적할 수 있습니다. 로그 그룹을 비롯한 AWS 리소스에 태그를 적용하면 태그별로 집계된 사용량 및 비용이 AWS 비용 할당 보고서에 포함됩니다. 비즈니스 범주를 나타내는 태그(예: 비용 센터, 애플리케이션 이름 또는 소유자)를 적용하여 여러 서비스에 대한 비용을 정리할 수 있습니다. 자세한 내용은 AWS Billing 사용 설명서의 사용자 지정 결제 보고서에 비용 할당 태그 사용을 참조하세요.

태그 제한

태그에 적용되는 제한은 다음과 같습니다.

기본 제한

- 로그 그룹당 최대 태그 수는 50개입니다.
- 태그 키와 값은 대소문자를 구분합니다.
- 삭제된 로그 그룹에 대해 태그를 변경하거나 편집할 수 없습니다.

태그 키 제한

- 각 태그 키는 고유해야 합니다. 이미 사용 중인 키를 가진 태그를 추가하면 기존 키-값 페어에 새 태그가 덮어쓰기 됩니다.
- 이 접두사는 에서 사용하도록 예약되어 `aws:` 있으므로 태그 키를 로 시작할 수 없습니다. AWS AWS 사용자 대신 이 접두사로 시작하는 태그를 생성하지만 편집하거나 삭제할 수는 없습니다.
- 태그 키의 길이는 유니코드 1~128자여야 합니다.
- 태그 키의 문자로는 유니코드 문자, 숫자, 공백 그리고 `_ . / = + - @` 같은 특수 문자가 허용됩니다.

태그 값 제한

- 태그 값의 길이는 유니코드 0~255자여야 합니다.
- 태그 값은 공백 상태로 둘 수 있습니다. 아니면 유니코드 문자, 숫자, 공백 그리고 _ . / = + - @ 같은 특수 문자를 사용할 수 있습니다.

를 사용하여 로그 그룹에 태그를 지정합니다. AWS CLI

AWS CLI를 사용하여 태그를 추가, 나열 및 제거할 수 있습니다. 예제는 다음 설명서를 참조하세요.

create-log-group

로그 그룹을 생성합니다. 로그 그룹 생성 시 태그를 선택에 따라 추가할 수 있습니다.

tag-resource

지정된 Logs 리소스에 하나 이상의 태그 (키-값 쌍) 를 할당합니다. CloudWatch

list-tags-for-resource

Logs 리소스에 CloudWatch 연결된 태그를 표시합니다.

untag-resource

지정된 CloudWatch Logs 리소스에서 하나 이상의 태그를 제거합니다.

CloudWatch Logs API를 사용하여 로그 그룹에 태그 지정

CloudWatch Logs API를 사용하여 태그를 추가, 나열, 제거할 수 있습니다. 예제는 다음 설명서를 참조하세요.

CreateLogGroup

로그 그룹을 생성합니다. 로그 그룹 생성 시 태그를 선택에 따라 추가할 수 있습니다.

TagResource

지정된 CloudWatch Logs 리소스에 하나 이상의 태그 (키-값 쌍) 를 할당합니다.

ListTagsForResource

Logs 리소스에 CloudWatch 연결된 태그를 표시합니다.

UntagResource

지정된 CloudWatch Logs 리소스에서 하나 이상의 태그를 제거합니다.

다음을 사용하여 Logs의 로그 데이터를 암호화합니다. CloudWatch AWS Key Management Service

로그 그룹 데이터는 CloudWatch 로그에서 항상 암호화됩니다. 기본적으로 Logs는 저장된 CloudWatch 로그 데이터에 대해 서버 측 암호화를 사용합니다. 대신 AWS Key Management Service 를 사용하여 이를 암호화할 수 있습니다. 이렇게 하면 키를 사용하여 암호화가 수행됩니다. AWS KMS KMS 키를 로그 그룹과 연결하면 로그 그룹을 생성할 때 또는 로그 그룹이 생성된 후에 KMS 키를 로그 그룹과 연결하여 암호화를 사용할 AWS KMS 수 있습니다.

> **Important**
>
> CloudWatch 이제 Logs는 키로 사용하고 `kms:EncryptionContext:aws:logs:arn` 로그 그룹의 ARN을 해당 키의 값으로 사용하는 암호화 컨텍스트를 지원합니다. KMS 키로 이미 암호화된 로그 그룹이 있고 단일 계정 및 로그 그룹에서 키를 사용하도록 제한하려면 IAM 정책에 조건을 포함하는 새 KMS 키를 할당해야 합니다. 자세한 정보는 AWS KMS 키 및 암호화 컨텍스트을 참조하세요.

KMS 키를 로그 그룹에 연결하고 나면 로그 데이터에서 새로 수집된 모든 데이터를 이 키를 사용해 암호화할 수 있습니다. 이 데이터는 보존 기간 내내 암호화된 형식으로 저장됩니다. CloudWatch 로그는 요청될 때마다 이 데이터를 해독합니다. CloudWatch 암호화된 데이터가 요청될 때마다 로그에 KMS 키에 대한 권한이 있어야 합니다.

나중에 KMS 키를 로그 그룹에서 분리하면 Logs는 CloudWatch Logs 기본 암호화 방법을 사용하여 새로 수집된 데이터를 암호화합니다. CloudWatch KMS 키로 암호화된 이전에 수집된 모든 데이터는 KMS 키로 암호화된 상태로 유지됩니다. CloudWatch 로그에서 계속 키를 참조할 수 있기 때문에 KMS 키의 연결이 끊긴 후에도 CloudWatch 로그에서 해당 데이터를 반환할 수 있습니다. 하지만 나중에 키를 비활성화하면 CloudWatch Logs는 해당 키로 암호화된 로그를 읽을 수 없습니다.

> **Important**
>
> CloudWatch 로그는 대칭 KMS 키만 지원합니다. 비대칭 키를 사용하여 로그 그룹의 데이터를 암호화하지 마세요. 자세한 내용은 대칭 및 비대칭 키 사용을 참조하세요.

Limits

- 다음 단계를 수행하려면 `kms:CreateKey`, `kms:GetKeyPolicy` 및 `kms:PutKeyPolicy` 권한이 있어야 합니다.
- 로그 그룹에서 키를 연결하거나 연결 해제하고 난 후 이러한 변경이 적용되기까지 최대 5분의 시간이 소요될 수 있습니다.
- 관련 키에 대한 CloudWatch 로그 액세스를 취소하거나 연결된 KMS 키를 삭제하면 CloudWatch Logs의 암호화된 데이터를 더 이상 검색할 수 없습니다.
- 콘솔을 사용하여 KMS 키를 로그 그룹과 연결할 수 없습니다. CloudWatch

1단계: 키 생성 AWS KMS

다음 create-key 명령을 사용하여 KMS 키 생성:

```
aws kms create-key
```

이 명령의 출력 화면에는 키의 키 ID와 Amazon 리소스 이름(ARN)이 포함됩니다. 다음은 예 출력입니다.

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

2단계: KMS 키에 대한 권한 설정

기본적으로 모든 AWS KMS 키는 비공개입니다. 리소스 소유자만 이를 사용하여 데이터를 암호화 및 해독할 수 있습니다. 그러나 리소스 소유자가 원한다면 다른 사용자 및 리소스에 KMS 키에 대한 액세스 권한을 부여할 수 있습니다. 이 단계에서는 CloudWatch 로그 서비스 주체에게 키 사용 권한을 부여합니다. 이 서비스 보안 주체는 KMS 키가 저장된 동일한 AWS 지역에 있어야 합니다.

KMS 키 사용을 지정한 AWS 계정 또는 로그 그룹으로만 제한하는 것이 가장 좋습니다.

먼저 다음 get-key-policy명령을 `policy.json` 사용하여 KMS 키의 기본 정책을 저장합니다.

```
aws kms get-key-policy --key-id key-id --policy-name default --output text > ./
policy.json
```

텍스트 편집기에서 `policy.json` 파일을 열고 다음 설명 중 하나에서 굵은 글꼴로 표시된 섹션을 추가합니다. 기존 설명과 새 설명을 쉼표로 구분합니다. 이러한 명령문은 `Condition` 섹션을 사용하여 AWS KMS 키의 보안을 강화합니다. 자세한 정보는 AWS KMS 키 및 암호화 컨텍스트을 참조하세요.

이 예제의 `Condition` 섹션에서는 키를 단일 로그 그룹 ARN으로 제한합니다.

```
{
 "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::Your_account_ID: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": {
                    "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:region:account-
id:log-group:log-group-name"
                }
            }
        }
    ]
}
```

이 예제의 Condition 섹션에서는 AWS KMS 키 사용을 지정된 계정으로 제한하지만 모든 로그 그룹에 사용할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::Your_account_ID: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": {
                "ArnLike": {
                    "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:region:account-
id:*"
                }
            }
        }
    ]
}
```

마지막으로 다음 put-key-policy명령을 사용하여 업데이트된 정책을 추가합니다.

```
aws kms put-key-policy --key-id key-id --policy-name default --policy file://
policy.json
```

3단계: KMS 키를 로그 그룹에 연결

로그 그룹을 생성할 때와 그 이후에 KMS 키를 로그 그룹에 연결할 수 있습니다.

로그 그룹에 이미 연결된 KMS 키가 있는지 확인하려면 다음 describe-log-groups명령을 사용하십시오.

```
aws logs describe-log-groups --log-group-name-prefix "log-group-name-prefix"
```

출력에 kmsKeyId 필드가 포함된 경우 로그 그룹은 해당 필드의 값에 대해 표시된 키와 연결됩니다.

생성 시 로그 그룹에 KMS 키를 연결

다음과 같이 create-log-group 명령을 실행합니다.

```
aws logs create-log-group --log-group-name my-log-group --kms-key-id "key-arn"
```

기존 로그 그룹에 KMS 키를 연결

다음과 같이 associate-kms-key 명령을 실행합니다.

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id "key-arn"
```

4단계: 로그 그룹에서 키 연결 해제

로그 그룹과 연결된 KMS 키를 분리하려면 다음 명령을 사용합니다. disassociate-kms-key

```
aws logs disassociate-kms-key --log-group-name my-log-group
```

AWS KMS 키 및 암호화 컨텍스트

AWS Key Management Service 키와 암호화된 로그 그룹의 보안을 강화하기 위해 이제 CloudWatch Logs는 로그 그룹 ARN을 로그 데이터를 암호화하는 데 사용되는 암호화 컨텍스트의 일부로 넣습니다. 암호화 컨텍스트는 추가 인증 데이터로 사용되는 키-값 쌍의 집합입니다. 암호화 컨텍스트를 사용하면 IAM 정책 조건을 사용하여 AWS 계정 및 로그 그룹별로 AWS KMS 키에 대한 액세스를 제한할 수 있습니다. 자세한 내용은 암호화 컨텍스트 및 IAM JSON 정책 요소: 조건을 참조하세요.

암호화된 각 로그 그룹에 대해 서로 다른 KMS 키를 사용하는 것이 좋습니다.

이전에 암호화한 로그 그룹이 있고 해당 로그 그룹에만 작동하는 새 KMS 키를 사용하도록 로그 그룹을 변경하려는 경우 다음 단계를 수행합니다.

KMS 키를 해당 로그 그룹으로 제한하는 정책에 따라 사용하도록 암호화된 로그 그룹 변환

1. 다음 명령을 입력하여 로그 그룹의 현재 키의 ARN을 찾습니다.

   ```
   aws logs describe-log-groups
   ```

   출력에는 다음 줄이 포함됩니다. ARN을 기록해 둡니다. 7단계에서 사용해야 합니다.

   ```
   ...
   "kmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/01234567-89ab-
   cdef-0123-456789abcdef"
   ...
   ```

2. 다음 명령을 입력하여 새 KMS 키 생성:

   ```
   aws kms create-key
   ```

3. 다음 명령을 입력하여 새 키의 정책을 `policy.json` 파일에 저장합니다.

   ```
   aws kms get-key-policy --key-id new-key-id --policy-name default --output text > ./
   policy.json
   ```

4. 텍스트 편집기를 사용하여 `policy.json`을 열고 `Condition` 표현식을 정책에 추가합니다.

   ```
   {
   ```

```
    "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT-ID: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": {
                "ArnLike": {
                    "kms:EncryptionContext:aws:logs:arn":
 "arn:aws:logs:REGION:ACCOUNT-ID:log-
group:LOG-GROUP-NAME"
                }
            }
        }
    ]
}
```

5. 다음 명령을 입력하여 업데이트된 정책을 새 KMS 키에 추가:

```
aws kms put-key-policy --key-id new-key-ARN --policy-name default --policy file://
policy.json
```

6. 다음 명령을 입력하여 정책을 로그 그룹과 연결합니다.

```
aws logs associate-kms-key --log-group-name my-log-group --kms-key-id new-key-ARN
```

CloudWatch 이제 로그는 새 키를 사용하여 모든 새 데이터를 암호화합니다.

7. 그런 다음 이전 키에서 Decrypt를 제외한 모든 권한을 취소합니다. 먼저 다음 명령을 입력하여 이전 정책을 검색합니다.

```
aws kms get-key-policy --key-id old-key-ARN --policy-name default --output text
 > ./policy.json
```

8. 텍스트 편집기를 사용하여 policy.json을 열고 Action 목록에서 kms:Decrypt*를 제외한 모든 값을 제거합니다.

```
{
    "Version": "2012-10-17",
    "Id": "key-default-1",
    "Statement": [
        {
            "Sid": "Enable IAM User Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::Your_account_ID:root"
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "logs.region.amazonaws.com"
            },
            "Action": [
                "kms:Decrypt*"
            ],
            "Resource": "*"
        }
    ]
}
```

9. 다음 명령을 입력하여 업데이트된 정책을 이전 키에 추가합니다.

```
aws kms put-key-policy --key-id old-key-ARN --policy-name default --policy file://
policy.json
```

마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원

로그 그룹 데이터 보호 정책을 사용하면 CloudWatch 로그에 수집된 민감한 데이터를 보호할 수 있습니다. 이러한 정책을 사용하면 계정의 로그 그룹에서 모은 로그 이벤트에 나타나는 민감한 데이터를 감사하고 마스킹할 수 있습니다.

데이터 보호 정책을 만들면 기본적으로 선택한 데이터 식별자와 일치하는 민감한 데이터가 CloudWatch Logs Insights, 지표 필터, 구독 필터를 비롯한 모든 송신 지점에서 마스킹됩니다. `logs:Unmask` IAM 권한이 있는 사용자만 마스킹되지 않은 데이터를 볼 수 있습니다.

계정의 모든 로그 그룹에 대한 데이터 보호 정책을 생성할 수 있으며 개별 로그 그룹에 대한 데이터 보호 정책을 생성할 수도 있습니다. 전체 계정에 대한 정책을 생성하면 기존 로그 그룹과 향후 생성되는 로그 그룹 모두에 적용됩니다.

전체 계정에 대한 데이터 보호 정책을 생성하고 단일 로그 그룹에 대한 정책도 생성하는 경우 두 정책 모두 해당 로그 그룹에 적용됩니다. 두 정책 중 하나에 지정된 모든 관리형 데이터 식별자는 해당 로그 그룹에서 감사되고 마스킹됩니다.

> **Note**
>
> 민감한 데이터 마스킹은 Standard 로그 클래스의 로그 그룹에만 지원됩니다. 계정의 모든 로그 그룹에 대해 데이터 보호 정책을 만들면 표준 로그 클래스의 로그 그룹에만 적용됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

각 로그 그룹에는 로그 그룹 수준의 데이터 보호 정책이 하나만 있을 수 있지만 해당 정책은 감사하고 마스킹할 여러 관리형 데이터 식별자를 지정할 수 있습니다. 데이터 보호 정책의 한도는 30,720자입니다.

> **Important**
>
> 중요한 데이터는 로그 그룹에 수집될 때 감지되고 마스킹 처리됩니다. 데이터 보호 정책을 설정하면 해당 시간 이전에 로그 그룹에 수집된 로그 이벤트는 마스킹 처리되지 않습니다.

CloudWatch 로그는 재무 데이터, 개인 건강 정보 (PHI) 및 개인 식별 정보 (PII) 를 보호하기 위해 선택할 수 있는 사전 구성된 데이터 유형을 제공하는 다양한 관리형 데이터 식별자를 지원합니다. CloudWatch 로그 데이터 보호를 통해 패턴 매칭 및 기계 학습 모델을 활용하여 민감한 데이터를 탐지할 수 있습니다. 일부 유형의 관리형 데이터 식별자의 경우 민감한 데이터와 가까운 특정 키워드를 찾아야만 탐지 효과가 달라집니다. 또한 사용자 지정 데이터 식별자를 사용하여 특정 사용 사례에 맞는 데이터 식별자를 만들 수 있습니다.

선택한 데이터 식별자와 일치하는 민감한 데이터가 감지되면 측정항목이 생성됩니다. CloudWatch 이는 LogEventsWithFindings지표이며 AWS/Logs 네임스페이스에서 내보냅니다. 이 메트릭을 사용하여 CloudWatch 경보를 생성하고 그래프와 대시보드로 시각화할 수 있습니다. 데이터 보호에서 내보낸 지표는 판매된 지표이며 무료입니다. CloudWatch 로그가 전송하는 지표에 대한 자세한 내용은 을 CloudWatch 참조하십시오. CloudWatch 메트릭을 사용한 모니터링

각 관리형 데이터 식별자는 특정 국가 또는 지역의 신용 카드 번호, AWS 비밀 액세스 키 또는 여권 번호와 같은 특정 유형의 민감한 데이터를 탐지하도록 설계되었습니다. 데이터 보호 정책을 생성할 때 이러한 식별자를 사용하여 로그 그룹에서 수집한 로그를 분석하고, 감지되면 조치를 취하도록 구성할 수 있습니다.

CloudWatch 로그 데이터 보호는 관리형 데이터 식별자를 사용하여 다음 범주의 민감한 데이터를 탐지할 수 있습니다.

- 자격 증명 (예: 개인 키 또는 AWS 비밀 액세스 키)
- 금융 정보(예: 신용 카드 번호)
- 개인 식별 정보(PII)(예: 운전면허증 또는 사회 보장 번호)
- 보호 대상 건강 정보(PHI)(예: 건강 보험 또는 의료 식별 번호)
- 디바이스 식별자(예: IP 주소 또는 MAC 주소)

보호할 수 있는 데이터 유형에 대한 자세한 내용은 보호할 수 있는 데이터 유형 섹션을 참조하세요.

목차

데이터 보호 정책 이해

주제

데이터 보호 정책이란 무엇입니까?

CloudWatch 로그는 데이터 보호 정책을 사용하여 검사하려는 민감한 데이터와 해당 데이터를 보호하기 위해 취하려는 조치를 선택합니다. 관심 있는 민감한 데이터를 선택하려면 데이터 식별자를 사용합니다. CloudWatch 그런 다음 로그 데이터 보호 기능이 기계 학습 및 패턴 매칭을 사용하여 민감한 데이터를 탐지합니다. 발견된 데이터 식별자에 따라 조치를 취하기 위해 감사와 비식별화 작업을 정의할 수 있습니다. 이러한 연산을 사용하여 발견된(또는 발견되지 않은) 민감한 데이터를 로깅하고 로그 이벤트를 확인할 때 민감한 데이터를 마스킹할 수 있습니다.

데이터 보호 정책은 어떻게 구성되어 있습니까?

다음 그림처럼 데이터 보호 정책 문서는 다음 요소를 포함합니다.

- 문서 상단에 위치하는 정책 전반의 선택적 정보
- 감사 및 비식별화 작업을 정의하는 문 하나

로그 CloudWatch 로그 그룹당 하나의 데이터 보호 정책만 정의할 수 있습니다. 데이터 보호 정책은 하나 이상의 거부 또는 비식별 명령문과 하나의 감사 명령문을 가질 수 있습니다.

데이터 보호 정책의 JSON 속성

데이터 보호 정책에는 식별을 위해 다음과 같은 기본 정책 정보가 필요합니다.

- Name(이름) – 정책 이름입니다.
- Description(설명)(선택 사항) – 정책 설명입니다.
- Version(버전) - 정책 언어 버전입니다. 현재 버전은 2021-06-01입니다.
- Statement(명령문) - 데이터 보호 정책 조치를 지정하는 명령문 목록입니다.

```
{
  "Name": "CloudWatchLogs-PersonalInformation-Protection",
  "Description": "Protect basic types of sensitive data",
  "Version": "2021-06-01",
  "Statement": [
        ...
  ]
}
```

정책 명령문을 위한 JSON 속성

정책 명령문은 데이터 보호 작업에 대한 탐지 컨텍스트를 설정합니다.

- Sid(선택 사항) - 명령문 식별자입니다.
- DataIdentifier— CloudWatch Logs가 스캔해야 하는 민감한 데이터. 이러한 데이터로는 이름, 주소 또는 전화번호가 있습니다.
- 작업 — 후속 조치 (감사 또는 식별 제거). CloudWatch 로그는 민감한 데이터를 발견하면 이러한 작업을 수행합니다.

```
{
  ...
  "Statement": [
    {
      "Sid": "audit-policy",
      "DataIdentifier": [
        "arn:aws:dataprotection::aws:data-identifier/Address"
      ],
      "Operation": {
        "Audit": {
          "FindingsDestination": {}
        }
      }
    },
```

정책 명령문 작업을 위한 JSON 속성

정책 명령문은 다음 데이터 보호 작업 중 하나를 설정합니다.

- Audit - 로깅을 중단하지 않고 지표 및 결과 보고서를 내보냅니다. 일치하는 문자열은 Logs가 CloudWatch AWS/Logs 네임스페이스에 게시하는 LogEventsWithFindings메트릭을 증가시킵니다. CloudWatch 해당 지표를 사용하여 경보를 생성할 수 있습니다.

  결과 보고서의 예는 감사 결과 보고서를 참조하세요.

  Logs가 보내는 메트릭에 대한 자세한 내용은 을 CloudWatch 참조하십시오. CloudWatch CloudWatch 메트릭을 사용한 모니터링
- De-identify -로깅을 중단하지 않고 중요한 데이터를 마스킹합니다.

데이터 보호 정책을 생성하거나 사용하는 데 필요한 IAM 권한

로그 그룹에 대한 데이터 보호 정책을 사용하려면 다음 표에 나와 있는 특정 권한이 있어야 합니다. 권한은 계정 전체 데이터 보호 정책과 단일 로그 그룹에 적용되는 데이터 보호 정책에 따라 다릅니다.

계정 수준 데이터 보호 정책에 필요한 권한

> **Note**
>
> Lambda 함수 내에서 이러한 작업을 수행하는 경우 Lambda 실행 역할 및 권한 경계에 다음 권한도 포함되어야 합니다.

Operation	IAM 권한 필요	Resource
감사 대상이 없는 데이터 보호 정책 생성	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
CloudWatch 로그를 감사 대상으로 삼아 데이터 보호 정책을 생성하세요.	`logs:PutAccountPolicy`	*

Operation	IAM 권한 필요	Resource
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*
	`logs:PutResourcePolicy`	*
	`logs:DescribeResourcePolicies`	*
	`logs:DescribeLogGroups`	*
Firehose를 감사 대상으로 사용하여 데이터 보호 정책을 만드세요.	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*
	`firehose:TagDeliveryStream`	`arn:aws:logs:::deliverystream/` *YOUR_DELIVERY_STREAM*
Amazon S3를 감사 대상으로 사용하여 데이터 보호 정책 생성	`logs:PutAccountPolicy`	*
	`logs:PutDataProtectionPolicy`	*
	`logs:CreateLogDelivery`	*

Operation	IAM 권한 필요	Resource
	s3:GetBucketPolicy	arn:aws:s3::: *YOUR_BUCKET*
	s3:PutBucketPolicy	arn:aws:s3::: *YOUR_BUCKET*
지정된 로그 그룹에서 마스킹된 로그 이벤트 마스킹 해제	logs:Unmask	arn:aws:logs:::log-group:*
기존 데이터 보호 정책 보기	logs:GetDataProtectionPolicy	*
데이터 보호 정책 삭제	logs:DeleteAccountPolicy	*
	logs:DeleteDataProtectionPolicy	*

데이터 보호 감사 로그가 이미 대상으로 전송되고 있는 경우, 동일한 대상으로 로그를 전송하는 다른 정책에는 logs:PutDataProtectionPolicy 및 logs:CreateLogDelivery 권한만 있으면 됩니다.

단일 로그 그룹에 대한 데이터 보호 정책에 필요한 권한

> **Note**
>
> Lambda 함수 내에서 이러한 작업을 수행하는 경우 Lambda 실행 역할 및 권한 경계에 다음 권한도 포함되어야 합니다.

Operation	IAM 권한 필요	Resource
감사 대상이 없는 데이터 보호 정책 생성	logs:PutDataProtectionPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :*

Operation	IAM 권한 필요	Resource
CloudWatch 로그를 감사 대상으로 삼아 데이터 보호 정책을 생성하세요.	logs:PutDataProtectionPolicy logs:CreateLogDelivery logs:PutResourcePolicy logs:DescribeResourcePolicies logs:DescribeLogGroups	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * * * *
Firehose를 감사 대상으로 사용하여 데이터 보호 정책을 만드세요.	logs:PutDataProtectionPolicy logs:CreateLogDelivery firehose:TagDeliveryStream	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * arn:aws:logs:::deliverystream/ *YOUR_DELIVERY_STREAM*
Amazon S3를 감사 대상으로 사용하여 데이터 보호 정책 생성	logs:PutDataProtectionPolicy logs:CreateLogDelivery s3:GetBucketPolicy s3:PutBucketPolicy	arn:aws:logs:::log-group: *YOUR_LOG_GROUP* :* * arn:aws:s3::: *YOUR_BUCKET* arn:aws:s3::: *YOUR_BUCKET*

Operation	IAM 권한 필요	Resource
마스킹된 로그 이벤트 마스크 해제	`logs:Unmask`	`arn:aws:logs:::log-group:` *YOUR_LOG_GROUP* `:*`
기존 데이터 보호 정책 보기	`logs:GetDataProtectionPolicy`	`arn:aws:logs:::log-group:` *YOUR_LOG_GROUP* `:*`
데이터 보호 정책 삭제	`logs:DeleteDataProtectionPolicy`	`arn:aws:logs:::log-group:` *YOUR_LOG_GROUP* `:*`

데이터 보호 감사 로그가 이미 대상으로 전송되고 있는 경우, 동일한 대상으로 로그를 전송하는 다른 정책에는 `logs:PutDataProtectionPolicy` 및 `logs:CreateLogDelivery` 권한만 있으면 됩니다.

샘플 데이터 보호 정책

다음 샘플 정책을 사용하면 사용자는 세 가지 유형의 감사 대상 모두에 감사 결과를 전송할 수 있는 데이터 보호 정책을 생성, 확인 및 삭제할 수 있습니다. 사용자는 마스크되지 않은 데이터를 볼 수 없습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "YOUR_SID_1",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:PutResourcePolicy",
                "logs:DescribeLogGroups",
                "logs:DescribeResourcePolicies"
            ],
            "Resource": "*"
        },
        {
            "Sid": "YOUR_SID_2",
```

```
            "Effect": "Allow",
            "Action": [
                "logs:GetDataProtectionPolicy",
                "logs:DeleteDataProtectionPolicy",
                "logs:PutDataProtectionPolicy",
                "s3:PutBucketPolicy",
                "firehose:TagDeliveryStream",
                "s3:GetBucketPolicy"
            ],
            "Resource": [
                "arn:aws:firehose:::deliverystream/YOUR_DELIVERY_STREAM",
                "arn:aws:s3:::YOUR_BUCKET",
                "arn:aws:logs:::log-group:YOUR_LOG_GROUP:*"
            ]
        }
    ]
}
```

계정 전체 데이터 보호 정책 생성

CloudWatch 로그 콘솔 또는 AWS CLI 명령을 사용하여 계정 내 모든 로그 그룹의 민감한 데이터를 마스킹하는 데이터 보호 정책을 만들 수 있습니다. 이는 현재 로그 그룹과 향후 생성하는 로그 그룹 모두에 영향을 미칩니다.

> **Important**
>
> 중요한 데이터는 로그 그룹에 수집될 때 감지되고 마스킹 처리됩니다. 데이터 보호 정책을 설정하면 해당 시간 이전에 로그 그룹에 수집된 로그 이벤트는 마스킹 처리되지 않습니다.

주제

- 콘솔
- AWS CLI

콘솔

콘솔을 사용하여 계정 전체 데이터 보호 정책 생성

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.

2. 탐색 창에서 설정을 선택합니다. 목록 하단 근처에 있습니다.
3. 로그 탭을 선택합니다.
4. 구성을 선택합니다.
5. 관리형 데이터 식별자의 경우 모든 로그 그룹에 대해 감사하고 마스킹하려는 데이터 유형을 선택합니다. 선택 상자에 입력하여 원하는 식별자를 찾을 수 있습니다.

   로그 데이터 및 비즈니스와 관련된 데이터 식별자만 선택하는 것이 좋습니다. 여러 유형의 데이터를 선택하면 오탐이 발생할 수 있습니다.

   보호할 수 있는 데이터 유형에 대한 자세한 내용은 보호할 수 있는 데이터 유형 섹션을 참조하세요.
6. (선택 사항) 사용자 지정 데이터 식별자를 사용하여 다른 유형의 데이터를 감사하고 마스킹하려면 사용자 지정 데이터 식별자 추가를 선택합니다. 그런 다음 로그 이벤트에서 해당 유형의 데이터를 검색하는 데 사용할 데이터 유형의 이름과 정규 표현식을 입력합니다. 자세한 정보는 사용자 지정 데이터 식별자을 참조하세요.

   단일 데이터 보호 정책에는 최대 10개의 사용자 지정 데이터 식별자가 포함될 수 있습니다. 사용자 지정 데이터 식별자를 정의하는 각 정규 표현식은 200자 이하여야 합니다.
7. (선택 사항) 감사 결과를 전송할 서비스를 하나 이상 선택합니다. 이러한 서비스에 감사 결과를 보내지 않기로 선택하더라도 선택한 중요한 데이터 유형은 여전히 마스킹 처리됩니다.
8. Activate data protection(데이터 보호 활성화)을 선택합니다.

AWS CLI

를 AWS CLI 사용하여 데이터 보호 정책을 만들려면

1. 텍스트 편집기를 사용하여 `DataProtectionPolicy.json` 정책 파일을 생성합니다. 정책 구문에 대한 자세한 내용은 다음 섹션을 참조하세요.
2. 다음 명령을 입력합니다.

```
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
```

AWS CLI 또는 API 작업을 위한 데이터 보호 정책 구문

AWS CLI 명령 또는 API 작업에 사용할 JSON 데이터 보호 정책을 생성할 때는 정책에 두 개의 JSON 블록이 포함되어야 합니다.

- 첫 번째 블록은 `DataIdentifer` 배열과 `Audit` 액션이 있는 `Operation` 속성을 모두 포함해야 합니다. `DataIdentifer` 배열은 마스킹 처리하려는 중요한 데이터 유형을 나열합니다. 사용 가능한 옵션에 대한 자세한 내용은 보호할 수 있는 데이터 유형 섹션을 참조하세요.

  중요한 데이터 용어를 찾으려면 `Audit` 액션이 있는 `Operation` 속성이 필요합니다. 이 `Audit` 액션에는 `FindingsDestination` 객체가 포함되어야 합니다. 선택적으로 이 `FindingsDestination` 객체를 사용하여 감사 결과 보고서를 보낼 하나 이상의 대상을 나열할 수 있습니다. 로그 그룹, Amazon Data Firehose 스트림, S3 버킷과 같은 대상을 지정하는 경우 해당 목적지는 이미 존재해야 합니다. 감사 결과 보고서의 예는 감사 결과 보고서를 참조하세요.
- 두 번째 블록은 `DataIdentifer` 배열과 `Deidentify` 액션이 있는 `Operation` 속성을 모두 포함해야 합니다. `DataIdentifer` 배열은 정책의 첫 번째 블록에 있는 `DataIdentifer` 배열과 정확히 일치해야 합니다.

  `Deidentify` 액션이 있는 `Operation` 속성은 실제로 데이터를 마스킹 처리하며 해당 속성은 `"MaskConfig": {}` 객체를 포함해야 합니다. `"MaskConfig": {}` 객체는 비어 있어야 합니다.

다음은 관리형 데이터 식별자만 사용하는 데이터 보호 정책의 예입니다. 이 정책은 이메일 주소와 미국 운전면허증을 마스킹합니다.

사용자 지정 데이터 식별자를 지정하는 정책에 대한 자세한 내용은 을 참조하십시오데이터 보호 정책에서 사용자 지정 데이터 식별자 사용.

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

단일 로그 그룹에 대한 데이터 보호 정책 생성

CloudWatch 로그 콘솔 또는 AWS CLI 명령을 사용하여 민감한 데이터를 마스킹하는 데이터 보호 정책을 만들 수 있습니다.

각 로그 그룹에 하나의 데이터 보호 정책을 할당할 수 있습니다. 각 데이터 보호 정책은 여러 유형의 정보를 감사할 수 있습니다. 각 데이터 보호 정책은 하나의 감사 진술을 포함할 수 있습니다.

주제

- 콘솔
- AWS CLI

콘솔

콘솔을 사용하여 데이터 보호 정책을 생성하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 로그, 로그 그룹을 선택합니다.
3. 로그 그룹의 이름을 선택합니다.
4. Actions(작업), Create data protection policy(데이터 보호 정책 생성)를 선택합니다.
5. 관리 데이터 식별자의 경우 이 로그 그룹에서 감사하고 마스킹하려는 데이터 유형을 선택합니다. 선택 상자에 입력하여 원하는 식별자를 찾을 수 있습니다.

   로그 데이터 및 비즈니스와 관련된 데이터 식별자만 선택하는 것이 좋습니다. 여러 유형의 데이터를 선택하면 오탐이 발생할 수 있습니다.

   관리 데이터 식별자를 사용하여 보호할 수 있는 데이터 유형에 대한 자세한 내용은 을 참조하십시오. 보호할 수 있는 데이터 유형
6. (선택 사항) 사용자 지정 데이터 식별자를 사용하여 다른 유형의 데이터를 감사하고 마스킹하려면 사용자 지정 데이터 식별자 추가를 선택합니다. 그런 다음 로그 이벤트에서 해당 유형의 데이터를 검색하는 데 사용할 데이터 유형의 이름과 정규 표현식을 입력합니다. 자세한 정보는 사용자 지정 데이터 식별자을 참조하세요.

   단일 데이터 보호 정책에는 최대 10개의 사용자 지정 데이터 식별자가 포함될 수 있습니다. 사용자 지정 데이터 식별자를 정의하는 각 정규 표현식은 200자 이하여야 합니다.
7. (선택 사항) 감사 결과를 전송할 서비스를 하나 이상 선택합니다. 이러한 서비스에 감사 결과를 보내지 않기로 선택하더라도 선택한 중요한 데이터 유형은 여전히 마스킹 처리됩니다.
8. Activate data protection(데이터 보호 활성화)을 선택합니다.

AWS CLI

를 AWS CLI 사용하여 데이터 보호 정책을 만들려면

1. 텍스트 편집기를 사용하여 `DataProtectionPolicy.json` 정책 파일을 생성합니다. 정책 구문에 대한 자세한 내용은 다음 섹션을 참조하세요.
2. 다음 명령을 입력합니다.

```
aws logs put-data-protection-policy --log-group-identifier "my-log-group" --policy-
document file:///Path/DataProtectionPolicy.json --region us-west-2
```

AWS CLI 또는 API 작업을 위한 데이터 보호 정책 구문

AWS CLI 명령 또는 API 작업에 사용할 JSON 데이터 보호 정책을 생성할 때는 정책에 두 개의 JSON 블록이 포함되어야 합니다.

- 첫 번째 블록은 `DataIdentifer` 배열과 `Audit` 액션이 있는 `Operation` 속성을 모두 포함해야 합니다. `DataIdentifer` 배열은 마스킹 처리하려는 중요한 데이터 유형을 나열합니다. 사용 가능한 옵션에 대한 자세한 내용은 보호할 수 있는 데이터 유형 섹션을 참조하세요.

  중요한 데이터 용어를 찾으려면 `Audit` 액션이 있는 `Operation` 속성이 필요합니다. 이 `Audit` 액션에는 `FindingsDestination` 객체가 포함되어야 합니다. 선택적으로 이 `FindingsDestination` 객체를 사용하여 감사 결과 보고서를 보낼 하나 이상의 대상을 나열할 수 있습니다. 로그 그룹, Amazon Data Firehose 스트림, S3 버킷과 같은 대상을 지정하는 경우 해당 목적지는 이미 존재해야 합니다. 감사 결과 보고서의 예는 감사 결과 보고서를 참조하세요.
- 두 번째 블록은 `DataIdentifer` 배열과 `Deidentify` 액션이 있는 `Operation` 속성을 모두 포함해야 합니다. `DataIdentifer` 배열은 정책의 첫 번째 블록에 있는 `DataIdentifer` 배열과 정확히 일치해야 합니다.

  `Deidentify` 액션이 있는 `Operation` 속성은 실제로 데이터를 마스킹 처리하며 해당 속성은 `"MaskConfig": {}` 객체를 포함해야 합니다. `"MaskConfig": {}` 객체는 비어 있어야 합니다.

다음은 이메일 주소와 미국 운전면허증을 마스킹 처리하는 데이터 보호 정책의 예입니다.

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

마스킹 처리되지 않은 데이터 보기

마스킹 처리되지 않은 데이터를 보려면 사용자에게 `logs:Unmask` 권한이 있어야 합니다. 이 권한이 있는 사용자는 다음과 같은 방법으로 마스킹 처리되지 않은 데이터를 볼 수 있습니다.

- 로그 스트림에서 이벤트를 볼 때는 Display(표시), Unmask(마스크 해제)를 선택합니다.
- unmask (@message) 명령이 포함된 CloudWatch 로그 인사이트 쿼리를 사용하세요. 다음 예제 쿼리는 스트림의 가장 최근 로그 이벤트 20개를 마스킹 처리되지 않은 상태로 표시합니다.

```
fields @timestamp, @message, unmask(@message)
| sort @timestamp desc
| limit 20
```

CloudWatch Logs Insights 명령에 대한 자세한 내용은 을 참조하십시오CloudWatch 로그 인사이트 쿼리 구문.

- unmask매개 변수와 함께 GetLogEventsor FilterLogEvents연산을 사용하십시오.

CloudWatchLogsFullAccess정책에는 logs:Unmask 권한이 포함됩니다. 권한이 CloudWatchLogsFullAccess없는 사용자에게 권한을 logs:Unmask 부여하려면 해당 사용자에게 사용자 지정 IAM 정책을 연결할 수 있습니다. 자세한 내용은 사용자(콘솔)에게 권한 추가를 참조하세요.

감사 결과 보고서

Logs, Amazon S3 또는 Firehose에 감사 보고서를 작성하도록 CloudWatch CloudWatch 로그 데이터 보호 감사 정책을 설정하는 경우, 이러한 결과 보고서는 다음 예와 유사합니다. CloudWatch 로그는 민감한 데이터가 포함된 각 로그 이벤트에 대해 하나의 결과 보고서를 작성합니다.

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

보고서의 필드는 다음과 같습니다.

- resourceArn 필드는 중요한 데이터가 발견된 로그 그룹을 표시합니다.
- dataIdentifiers 객체는 감사 중인 중요한 데이터 유형 중 하나에 대한 결과 정보를 표시합니다.

- `name` 필드는 이 섹션에서 보고하는 중요한 데이터의 유형을 식별합니다.
- `count` 필드는 해당 유형의 중요한 데이터가 로그 이벤트에서 나타나는 횟수를 표시합니다.
- `start` 및`end` 필드는 로그 이벤트에서 중요한 데이터의 각 항목이 나타나는 위치를 문자 수별로 표시합니다.

이전 예에서는 하나의 로그 이벤트에서 두 개의 이메일 주소를 발견한 보고서를 보여 줍니다. 첫 번째 이메일 주소는 로그 이벤트의 13번째 문자에서 시작하여 26번째 문자에서 끝납니다. 두 번째 이메일 주소는 30번째 문자부터 43번째 문자까지입니다. 이 로그 이벤트에는 두 개의 이메일 주소가 있지만 `LogEventsWithFindings` 지표는 중요한 데이터가 나타난 횟수가 아닌 중요한 데이터를 포함하는 로그 이벤트 수를 계산하므로 지표의 값은 하나씩만 증가합니다.

에 의해 보호되는 버킷으로 감사 결과를 전송하는 데 필요한 키 정책 AWS KMS

Amazon S3-관리형 키(SSE-S3)를 사용한 서버 측 암호화 또는 KMS 키(SSE-KMS)를 사용한 서버 측 암호화를 활성화하여 Amazon S3 버킷의 데이터를 보호할 수 있습니다. 자세한 내용은 Amazon S3 사용 설명서의 서버 측 암호화를 사용하여 데이터 보호를 참조하세요.

SSE-S3로 보호되는 버킷에 감사 결과를 보내는 경우 추가 구성이 필요하지 않습니다. Amazon S3는 암호화 키를 처리합니다.

SSE-KMS로 보호되는 버킷에 감사 결과를 보내는 경우 로그 전달 계정이 S3 버킷에 쓸 수 있도록 KMS에 대한 키 정책을 업데이트해야 합니다. SSE-KMS와 함께 사용하는 데 필요한 키 정책에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 Amazon S3 설명서를 참조하십시오.

보호할 수 있는 데이터 유형

이 섹션에는 CloudWatch 로그 데이터 보호 정책에서 보호할 수 있는 데이터 유형에 대한 정보가 포함되어 있습니다. CloudWatch 로그 관리 데이터 식별자는 금융 데이터, 개인 건강 정보 (PHI) 및 개인 식별 정보 (PII) 를 보호하기 위해 사전 구성된 데이터 유형을 제공합니다. 또한 사용자 지정 데이터 식별자를 사용하여 특정 사용 사례에 맞는 데이터 식별자를 만들 수 있습니다.

목차

- 금융 정보
  - 금융 데이터 유형을 위한 데이터 식별자 ARN
- 보호 대상 건강 정보(PHI)
  - 보호 대상 건강 정보(PHI) 데이터 유형에 대한 데이터 식별자 ARN
- 개인 식별 정보(PII)
  - 운전면허증 식별 번호 키워드
  - 국가별 식별 번호 키워드
  - 여권 번호 키워드
  - 납세자 식별 및 참조 번호 키워드
  - 개인 식별 정보(PII) 데이터 식별자 ARN
- 사용자 지정 데이터 식별자
  - 사용자 지정 데이터 식별자란?
  - 사용자 지정 데이터 식별자 제약
  - 콘솔에서 사용자 지정 데이터 식별자 사용
  - 데이터 보호 정책에서 사용자 지정 데이터 식별자 사용

CloudWatch 민감한 데이터 유형의 관리 데이터 식별자를 기록합니다.

이 섹션에는 관리형 데이터 식별자를 사용하여 보호할 수 있는 데이터 유형과 이러한 각 데이터 유형과 관련된 국가 및 지역에 대한 정보가 포함되어 있습니다.

일부 민감한 데이터 유형의 경우 CloudWatch Logs data Protection은 데이터 근처에 있는 키워드를 검색하여 해당 키워드를 찾은 경우에만 일치하는 키워드를 찾습니다. 키워드가 특정 유형의 데이터와 인접해야 하는 경우 키워드는 일반적으로 데이터로부터 30자 이내(포함) 이내여야 합니다.

키워드에 공백이 포함된 경우 CloudWatch 로그 데이터 보호는 공백이 없거나 공백 대신 밑줄 (_) 또는 하이픈 (-) 이 포함된 키워드 변형을 자동으로 검색합니다. 경우에 따라 CloudWatch Logs는 키워드의 일반적인 변형을 처리하기 위해 키워드를 확장하거나 축약하기도 합니다.

다음 표에는 CloudWatch Logs가 관리형 데이터 식별자를 사용하여 탐지할 수 있는 자격 증명, 기기, 금융, 의료 및 보호 대상 건강 정보 (PHI) 의 유형이 나열되어 있습니다. 이는 개인 식별 정보(PII)로도 인정될 수 있는 특정 유형의 데이터에 추가됩니다.

언어 및 지역에 구애받지 않는 지원되는 식별자

식별자	범주
Address	개인
AwsSecretKey	보안 인증 정보
CreditCardExpiration	금융
CreditCardNumber	금융
CreditCardSecurityCode	금융
EmailAddress	개인
IpAddress	개인
LatLong	개인
Name	개인
OpenSshPrivateKey	보안 인증
PgpPrivateKey	보안 인증
PkcsPrivateKey	보안 인증
PuttyPrivateKey	보안 인증 정보
VehicleIdentificationNumber	개인

리전별 데이터 식별자에는 식별자 이름, 하이픈, 두 글자(ISO 3166-1 alpha-2) 코드를 포함해야 합니다. 예를 들어 DriversLicense-US입니다.

두 글자로 된 국가 또는 리전 코드를 포함해야 하는 지원되는 식별자

식별자	범주	국가 및 언어
BankAccountNumber	금융	DE, ES, FR, GB, IT
CepCode	개인	BR

식별자	범주	국가 및 언어
Cnpj	개인	BR
CpfCode	개인	BR
DriversLicense	개인	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	상태	미국
ElectoralRollNumber	개인	GB
HealthInsuranceCardNumber	상태	EU
HealthInsuranceClaimNumber	상태	미국
HealthInsuranceNumber	상태	FR
HealthcareProcedureCode	상태	미국
IndividualTaxIdentificationNumber	개인	미국
InseeCode	개인	FR
MedicareBeneficiaryNumber	상태	미국
NationalDrugCode	상태	미국
NationalIdentificationNumber	개인	DE, ES, IT
NationalInsuranceNumber	개인	GB
NationalProviderId	상태	미국
NhsNumber	상태	GB

식별자	범주	국가 및 언어
NieNumber	개인	ES
NifNumber	개인	ES
PassportNumber	개인	CA, DE, ES, FR, GB, IT, US
PermanentResidenceNumber	개인	CA
PersonalHealthNumber	상태	CA
PhoneNumber	개인	BR, DE, ES, FR, GB, IT, US
PostalCode	개인	CA
RgNumber	개인	BR
SocialInsuranceNumber	개인	CA
Ssn	개인	ES, US
TaxId	개인	DE, ES, FR, GB
ZipCode	개인	미국

보안 인증 정보

CloudWatch 로그 데이터 보호는 다음 유형의 자격 증명을 찾을 수 있습니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
AWS 비밀 액세스 키	`AwsSecretKey`	`aws_secret_access_key` , `credentials` , `secret access key`, `secret key`, `set-awscredential`	모두

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
OpenSSH 프라이빗 키	`OpenSSHPrivateKey`	None	모두
PGP 프라이빗 키	`PgpPrivateKey`	None	모두
Pkcs 프라이빗 키	`PkcsPrivateKey`	None	모두
PuTTY 프라이빗 키	`PuttyPrivateKey`	None	모두

보안 인증 데이터 유형을 위한 데이터 식별자 ARN

다음은 데이터 보호 정책에 추가할 수 있는 데이터 식별자의 Amazon 리소스 이름(ARN) 목록입니다.

보안 인증 데이터 식별자 ARN
`arn:aws:dataprotection::aws:data-identifier/AwsSecretKey`
`arn:aws:dataprotection::aws:data-identifier/OpenSshPrivateKey`
`arn:aws:dataprotection::aws:data-identifier/PgpPrivateKey`
`arn:aws:dataprotection::aws:data-identifier/PkcsPrivateKey`
`arn:aws:dataprotection::aws:data-identifier/PuttyPrivateKey`

디바이스 식별자

CloudWatch 로그 데이터 보호는 다음 유형의 기기 식별자를 찾을 수 있습니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
IP 주소	`IpAddress`	None	모두

디바이스 데이터 유형을 위한 데이터 식별자 ARN

다음은 데이터 보호 정책에 추가할 수 있는 데이터 식별자의 Amazon 리소스 이름(ARN) 목록입니다.

디바이스 데이터 식별자 ARN
`arn:aws:dataprotection::aws:data-identifier/IpAddress`

금융 정보

CloudWatch 로그 데이터 보호는 다음 유형의 재무 정보를 찾을 수 있습니다.

데이터 보호 정책을 설정하면 Logs는 CloudWatch 로그 그룹이 위치한 지리적 위치에 관계없이 사용자가 지정한 데이터 식별자를 검색합니다. 이 표에서 Countries and regions(국가 및 리전) 열에 있는 정보는 해당 국가 및 리전에 적합한 키워드를 검색하기 위해 데이터 식별자에 두 글자로 된 국가 코드를 추가해야 하는지 그 여부를 지정합니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
은행 계좌 번호	`BankAccountNumber`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 은행 계좌 번호 키워드 표를 참조하세요.	프랑스, 독일, 이탈리아, 스페인, 영국	여기에는 국가 코드와 같은 요소를 포함하여 최대 34개의 영숫자로 구성된 국제 은행 계좌

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
				번호 (IBAN) 가 포함됩니다.
신용 카드 유효 기간	`CreditCardExpiration`	`exp d`, `exp m`, `exp y`, `expiration` , `expiry`	모두	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
신용 카드 번호	CreditCardNumber	account number, american express, amex, bank card, card, card number, card num, cc #, ccn, check card, credit, credit card#, dankort, debit, debit card, diners club, discover, electron, japanese card bureau, jcb, mastercard , mc, pan, payment account number, payment card number, pcn, union pay, visa	모두	적발 시 데이터는 Luhn check 공식을 준수하는 13~19자리 시퀀스여야 하며 아메리칸 익스프레스, 단코트, 다이너스 클럽, 디스커버, 일렉트론, 일본 카드 뷰로(JCB), 마스터카드, 비자

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
				등의 신용 카드 유형에 표준 카드 번호 접두사를 사용해야 합니다. UnionPay
신용 카드 인증 코드	`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`	모두	

은행 계좌 번호 키워드

국가 코드와 같은 요소를 포함하여 최대 34개의 영숫자로 구성된 국제 은행 계좌 번호(IBAN)를 탐지하려면 다음 키워드를 사용하세요.

국가	키워드
프랑스	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
독일	account code, account number, accountno# , accountnumber# , bankleitzahl , bban, customer account id, customer account number, customer bank account id, geheimzahl , iban, kartennummer , kontonummer , kreditkartennummer , sepa
이탈리아	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
스페인	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
영국	account code, account number, accountno# , accountnumber# , bban, customer account ID, customer account number, customer bank account id, iban, sepa
미국	bank account, bank acct, checking account, checking acct, deposit account, deposit acct, savings account, savings acct, chequing account, chequing acct

CloudWatch 신용 카드 발급 기관이 공개 테스트를 위해 예약한 다음과 같은 시퀀스는 로그에 기록되지 않습니다.

```
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.
```

금융 데이터 유형을 위한 데이터 식별자 ARN

다음은 데이터 보호 정책에 추가할 수 있는 데이터 식별자의 Amazon 리소스 이름(ARN) 목록입니다.

금융 데이터 식별자 ARN
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-DE
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-ES
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-FR
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-GB
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-IT
arn:aws:dataprotection::aws:data-identifier/BankAccountNumber-US
arn:aws:dataprotection::aws:data-identifier/CreditCardExpiration
arn:aws:dataprotection::aws:data-identifier/CreditCardNumber
arn:aws:dataprotection::aws:data-identifier/CreditCardSecurityC ode

보호 대상 건강 정보(PHI)

CloudWatch 로그 데이터 보호는 다음 유형의 보호된 건강 정보 (PHI) 를 찾을 수 있습니다.

데이터 보호 정책을 설정하면 Logs는 CloudWatch 로그 그룹이 위치한 지리적 위치에 관계없이 사용자가 지정한 데이터 식별자를 검색합니다. 이 표에서 Countries and regions(국가 및 리전) 열에 있는 정보는 해당 국가 및 리전에 적합한 키워드를 검색하기 위해 데이터 식별자에 두 글자로 된 국가 코드를 추가해야 하는지 그 여부를 지정합니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
마약단속국(DEA) 등록 번호	`DrugEnforcementAgencyNumber`	`dea number`, `dea registration`	미국
건강 보험 카드 번호(EHIC)	`HealthInsuranceCardNumber`	`assicurazione sanitaria numero`, `carta assicurazione numero`, `carte d’assurance maladie `, `carte européenne d'assurance maladie `, `ceam`, `ehic`, `ehic#`, `finlandehicnumber# `, `gesundheitskarte `, `hälsokort `, `health card`, `health card number`, `health insurance card`, `health insurance number`, `insurance card number`, `krankenversicherungskarte `, `krankenversicherungsnummer `, `medical account number`, `numero conto medico`, `numéro d’assurance maladie `, `numéro`	유럽 연합

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
		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 , sjukförsäkring nummer, sjukförsäkringskort , suomi ehic-numero , tarjeta de salud, terveyskortti , tessera sanitaria assicurazione numero , versicherungsnummer	
건강 보험 청구 번호(HICN)	HealthInsuranceClaimNumber	health insurance claim number, hic no, hic no., hic number, hic#, hicn, hicn#, hicno#	미국
건강 보험 또는 의료 식별 번호	HealthInsuranceNumber	carte d'assuré social, carte vitale, insurance card	프랑스
Healthcare Common Procedure Coding System(HCPCS) 코드	HealthcareProcedureCode	current procedural terminology , hcpcs, healthcare common procedure coding system	미국

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전
메디케어 수혜자 번호(MBN)	MedicareBeneficiaryNumber	mbi, medicare beneficiary	미국
국가 의약품 코드(NDC)	NationalDrugCode	national drug code, ndc	미국
국가 공급자 식별자(NPI)	NationalProviderId	hipaa, n.p.i., national provider, npi	미국
국가 의료 서비스(NHS) 번호	NhsNumber	national health service, NHS	영국
개인 건강 번호	PersonalHealthNumber	canada healthcare number, msp number, care number, phn, soins de santé	캐나다

보호 대상 건강 정보(PHI) 데이터 유형에 대한 데이터 식별자 ARN

다음은 보호 대상 건강 정보(PHI) 데이터 보호 정책에서 사용할 수 있는 데이터 식별자 Amazon 리소스 이름(ARN)을 나열합니다.

PHI 데이터 식별자 ARN
arn:aws:dataprotection::aws:data-identifier/DrugEnforcementAgencyNumber-US
arn:aws:dataprotection::aws:data-identifier/HealthcareProcedureCode-US
arn:aws:dataprotection::aws:data-identifier/HealthInsuranceCardNumber-EU

PHI 데이터 식별자 ARN
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

개인 식별 정보(PII)

CloudWatch 로그 데이터 보호는 다음 유형의 개인 식별 정보 (PII) 를 찾을 수 있습니다.

데이터 보호 정책을 설정하면 Logs는 CloudWatch 로그 그룹이 위치한 지리적 위치에 관계없이 사용자가 지정한 데이터 식별자를 검색합니다. 이 표에서 Countries and regions(국가 및 리전) 열에 있는 정보는 해당 국가 및 리전에 적합한 키워드를 검색하기 위해 데이터 식별자에 두 글자로 된 국가 코드를 추가해야 하는지 그 여부를 지정합니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
생년월일	DateOfBirth	dob, date of birth, birthdate , birth	모두	지원에는 모든 숫

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
		`date`, `birthday`, `b-day`, `bday`		자, 숫자 및 월 이름의 조합과 같은 대부분의 날짜 형식이 포함됩니다. 날짜 구성 요소는 공백, 슬래시(/) 또는 하이픈(-)으로 구분할 수 있습니다.
Código de Endereçamento Postal(CEP)	`CepCode`	`cep`, `código de endereçamento postal`, `codigo de endereçamento postal`	브라질	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
Cadastro Nacional da Pessoa Jurídica(CNPJ)	Cnpj	cadastro nacional da pessoa jurídica, cadastro nacional da pessoa juridica, cnpj	브라질	
Cadastro de Pessoas Físicas(CPF)	CpfCode	Cadastro de pessoas fisicas, cadastro de pessoas físicas, cadastro de pessoa física, cadastro de pessoa fisica, cpf	브라질	
운전면허증 식별 번호	DriversLicense	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 운전면허증 식별 번호 표를 참조하세요.	여러 국가. 자세한 내용은 운전면허증 식별 번호 표를 참조하세요.	
선거인단 번호	ElectoralRollNumber	electoral #, electoral number, electoral roll #, electoral roll no., electoral roll number, electoral rollno	영국	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
개인 납세자 식별	`IndividualTaxIdentificationNumber`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 개인 납세자 식별 번호 표를 참조하세요.	브라질, 프랑스, 독일, 스페인, 영국	
국립 통계 경제 연구소(INSEE)	`InseeCode`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 국가 식별 번호용 키워드 표를 참조하세요.	프랑스	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
국적 식별 번호	`NationalIdentificationNumber`	예. 자세한 내용은 이 섹션 뒷부분의 국가 식별 번호용 키워드 표를 참조하세요.	독일, 이탈리아, 스페인	여기에는 DNI(Documento Nacional de Identidad) 식별자(스페인), Codice fiscale 코드(이탈리아) 및 국가 신분증 번호(독일어)가 포함됩니다.
National Insurance Number(NINO)	`NationalInsuranceNumber`	`insurance no.`, `insurance number`, `insurance#` , `national insurance number`, `nationalinsurance#` , `nationalinsurancenumber` , `nin`, `nino`	영국	–

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
Número de identidad de extranjero(NIE)	`NieNumber`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 개인 납세자 식별 번호 표를 참조하세요.	스페인	
Número de Identificación Fiscal(NIF)	`NifNumber`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 개인 납세자 식별 번호 표를 참조하세요.	스페인	
여권 번호	`PassportNumber`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 여권 번호 키워드 표를 참조하세요.	캐나다, 프랑스, 독일, 이탈리아, 스페인, 영국, 미국	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
영주권 번호	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	캐나다	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
전화번호	PhoneNumber	브라질: 키워드에 cel, celular, fone, móvel, número residencial , numero residencial , telefone을 포함합니다. 기타: cell, contact, fax, fax number, mobile, phone, phone number, tel, telephone , telephone number	브라질, 캐나다, 프랑스, 독일, 이탈리아, 스페인, 영국, 미국	여기에는 미국 내 수신자 부담 전화번호와 팩스 번호가 포함됩니다. 키워드가 데이터에 근접한 경우 번호에 국가 코드를 포함할 필요가 없습니다. 키워드가 데이터에 근접하지 않은 경우 숫자에 국가

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
				코드가 포함되지 않아도 됩니다.
우편 번호	PostalCode	None	캐나다	
Registro Geral(RG)	RgNumber	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 개인 납세자 식별 번호 표를 참조하세요.	브라질	
Social Insurance Number(SIN)	SocialInsuranceNumber	canadian id, numéro d'assurance sociale, social insurance number, sin	캐나다	
사회 보장 번호(SSN)	Ssn	스페인 – número de la seguridad social, social security no., social security no. número de la seguridad social, social security number, socialsecurityno# , ssn, ssn# 미국 – social security, ss#, ssn	스페인, 미국	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
납세자 식별 번호 또는 참조 번호	`TaxId`	예. 국가마다 다른 키워드가 적용됩니다. 자세한 내용은 이 섹션 뒷부분의 개인 납세자 식별 번호 표를 참조하세요. .	프랑스, 독일, 스페인, 영국	여기에는 TIN(프랑스), Steueridentifikationsnummer(독일), CIF(스페인), UTR 및 TRN(영국)이 포함됩니다.
우편 번호	`ZipCode`	`zip code`, `zip+4`	미국	미국 우편번호입니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
우편 주소	`Address`	None	호주, 캐나다, 프랑스, 독일, 이탈리아, 스페인, 영국, 미국	키워드가 필수 항목은 아니지만 검색하려면 주소에 도시 또는 장소의 이름과 우편번호를 포함해야 합니다.
전자 메일 주소	`EmailAddress`	None	모두	

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
위성 항법 시스템(GPS) 좌표	LatLong	coordinate , coordinates , lat long, latitude longitude , location, position	모두	CloudWatch 위도 및 경도 좌표가 쌍으로 저장되고 DD (십진도) 형식 (예: 41.948614, -87.655311) 인 경우 로그는 GPS 좌표를 탐지할 수 있습니다. 지원에는 10진수 도 (DDM) 형식의 좌표(예: 41°56.9168'N

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
				87°39.3187'W) 또는 도, 분, 초(DMS) 형식의 좌표(예: 41°56'55.0104"N 87°39'19.1196"W)가 포함되지 않습니다.
전체 이름	`Name`	None	모두	CloudWatch 로그는 전체 이름만 감지할 수 있습니다. 라틴 문자 집합만 지원합니다.

데이터 유형	데이터 식별자 ID	필수 키워드	국가 및 리전	참고
차량 식별 번호(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	모두	CloudWatch 로그는 17자 시퀀스로 구성되고 ISO 3779 및 3780 표준을 준수하는 VIN을 탐지할 수 있습니다. 이 표준은 전 세계에서 사용할 수 있도록 설계되었습니다.

운전면허증 식별 번호 키워드

다양한 유형의 운전면허증 식별 번호를 감지하려면 CloudWatch Logs에서 숫자 근처에 키워드가 있어야 합니다. 다음 표에는 CloudWatch Logs가 특정 국가 및 지역에서 인식하는 키워드가 나열되어 있습니다.

국가 또는 리전	키워드
호주	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
오스트리아	führerschein, fuhrerschein, führerschein republik österreich, fuhrerschein republik osterreich
벨기에	fuehrerschein, fuehrerschein- nr, fuehrerscheinnummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrerscheinnummer, führerscheinnummer, numéro permis conduire, permis de conduire, rijbewijs, rijbewijsnummer
불가리아	превозно средство, свидетелство за управление на моторно, свидетелство за управление на мпс, сумпс, шофьорска книжка
캐나다	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,

국가 또는 리전	키워드
	drivers permit number, driving licence, driving license, driving permit, permis de conduire
크로아티아	vozačka dozvola
사이프러스	άδεια οδήγησης
체코 공화국	číslo licence, císlo licence řidiče, číslo řidičského průkazu, ovladače lic., povolení k jízdě, povolení řidiče, řidiči povolení, řidičský prúkaz, řidičský průkaz
덴마크	kørekort, kørekortnummer
에스토니아	juhi litsentsi number, juhiloa number, juhiluba, juhiluba number
핀란드	ajokortin numero, ajokortti, förare lic., körkort, körkort nummer, kuljettaja lic., permis de conduire
프랑스	permis de conduire
독일	fuehrerschein, fuehrerschein- nr, fuehrerschein nummer, fuhrerschein, führerschein, fuhrerschein- nr, führerschein- nr, fuhrerschein nummer, führerscheinnummer
그리스	δεια οδήγησης, adeia odigisis
헝가리	illesztőprogramok lic, jogosítvány, jogsi, licencszám, vezető engedély, vezetői engedély
아일랜드	ceadúnas tiomána
이탈리아	patente di guida, patente di guida numero, patente guida, patente guida numero

국가 또는 리전	키워드
라트비아	autovadītāja apliecība, licences numurs, vadītāja apliecība, vadītāja apliecības numurs, vadītāja atļauja, vadītāja licences numurs, vadītāji lic.
리투아니아	vairuotojo pažymėjimas
룩셈부르크	fahrerlaubnis, führerschäin
몰타	liċenzja tas-sewqan
네덜란드	permis de conduire, rijbewijs, rijbewijsnummer
폴란드	numer licencyjny, prawo jazdy, zezwolenie na prowadzenie
포르투갈	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
루마니아	numărul permisului de conducere, permis de conducere
슬로바키아	číslo licencie, číslo vodičského preukazu, ovládače lic., povolenia vodičov, povolenie jazdu, povolenie na jazdu, povolenie vodiča, vodičský preukaz
슬로베니아	vozniško dovoljenje

국가 또는 리전	키워드
스페인	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
스웨덴	ajokortin numero, dlno# ajokortti, drivere lic., förare lic., körkort, körkort nummer, körkortsn ummer, kuljettajat lic.
영국	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
미국	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

국가별 식별 번호 키워드

CloudWatch Logs에서 다양한 유형의 국가 식별 번호를 검색하려면 해당 번호와 가까운 위치에 키워드가 있어야 합니다. 여기에는 Documento Nacional de Identidad(DNI) 식별자(스페인), 프랑스 국립 통계 및 경제 연구소(INSEE) 코드, 독일 신분증 번호 및 Registro Geral(RG) 번호(브라질)이 포함됩니다.

다음 표에는 CloudWatch Logs가 특정 국가 및 지역에서 인식하는 키워드가 나열되어 있습니다.

국가 또는 리전	키워드
브라질	registro geral, rg
프랑스	assurance sociale, carte nationale d’identité, 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#
독일	ausweisnummer, id number, identification number, identity number, insurance number, personal id, personalausweis
이탈리아	codice fiscal, dati anagrafici, ehic, health card, health insurance card, p. iva, partita i.v.a., personal data, tax code, tessera sanitaria
스페인	dni, dni#, dninúmero#, documento nacional de identidad, identidad único, identidadúnico#, insurance number, national identification number, national identity, nationalid#, nationalidno#, número nacional identidad, personal identification number, personal identity no, unique identity number, uniqueid#

여권 번호 키워드

다양한 유형의 여권 번호를 감지하려면 CloudWatch Logs에서 숫자 근처에 키워드가 있어야 합니다. 다음 표에는 CloudWatch Logs가 특정 국가 및 지역에서 인식하는 키워드가 나열되어 있습니다.

국가 또는 리전	키워드
캐나다	passeport, passeport#, passport, passport#, passportno, passportno#
프랑스	numéro de passeport, passeport, passeport #, passeport #, passeportn °, passeport n °, passeportNon, passeport non
독일	ausstellungsdatum, ausstellungsort, geburtsdatum, passport, passports, reisepass, reisepass–nr, reisepassnummer
이탈리아	italian passport number, numéro passeport, numéro passeport italien, passaporto, passaporto italiana, passaporto numero, passport number, repubblica italiana passaporto
스페인	españa pasaporte, libreta pasaporte, número pasaporte, pasaporte, passport, passport book, passport no, passport number, spain passport
영국	passeport #, passeport n °, passeportNon, passeport non, passeportn °, passport #, passport no, passport number, passport#, passportid
미국	passport, travel document

납세자 식별 및 참조 번호 키워드

다양한 유형의 납세자 식별 번호 및 참조 번호를 검색하려면 CloudWatch Logs에서 숫자 근처에 키워드가 있어야 합니다. 다음 표에는 CloudWatch Logs가 특정 국가 및 지역에서 인식하는 키워드가 나열되어 있습니다.

국가 또는 리전	키워드
브라질	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
프랑스	numéro d'identification fiscale, tax id, tax identification number, tax number, tin, tin#
독일	identifikationsnummer, steuer id, steueride ntifikationsnummer, steuernummer, tax id, tax identification number, tax number
스페인	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#
영국	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
미국	개인 납세자 식별 번호(itin, i.t.i.n.)

개인 식별 정보(PII) 데이터 식별자 ARN

다음 표는 데이터 보호 정책에 추가할 수 있는 개인 식별 정보(PII) 데이터 식별자의 Amazon 리소스 이름(ARN) 목록입니다.

PII 및 데이터 식별자 ARN
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
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

PII 및 데이터 식별자 ARN
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
arn:aws:dataprotection::aws:data-identifier/DriversLicense-US
arn:aws:dataprotection::aws:data-identifier/ElectoralRollNumber-GB
arn:aws:dataprotection::aws:data-identifier/EmailAddress

PII 및 데이터 식별자 ARN
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
arn:aws:dataprotection::aws:data-identifier/PassportNumber-IT
arn:aws:dataprotection::aws:data-identifier/PassportNumber-US
arn:aws:dataprotection::aws:data-identifier/PermanentResidenceNumber-CA

PII 및 데이터 식별자 ARN
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
arn:aws:dataprotection::aws:data-identifier/VehicleIdentificationNumber
arn:aws:dataprotection::aws:data-identifier/ZipCode-US

사용자 지정 데이터 식별자

주제

사용자 지정 데이터 식별자란?

사용자 지정 데이터 식별자(CDI)를 통해 데이터 보호 정책에 사용할 수 있는 사용자 지정 정규 표현식을 정의할 수 있습니다. 사용자 지정 데이터 식별자를 사용하면 관리형 데이터 식별자로는 제공할 수 없는 비즈니스별 개인 식별 정보(PII) 사용 사례를 대상으로 지정할 수 있습니다. 예를 들어 사용자 지정 데이터 식별자를 사용하여 회사별 직원 ID를 찾을 수 있습니다. 사용자 지정 데이터 식별자는 관리형 데이터 식별자와 함께 사용할 수 있습니다.

사용자 지정 데이터 식별자 제약

CloudWatch 로그 사용자 지정 데이터 식별자에는 다음과 같은 제한이 있습니다.

- 각 데이터 보호 정책에 최대 10개의 사용자 지정 데이터 식별자가 지원됩니다.
- 사용자 지정 데이터 식별자 이름의 최대 길이는 128자입니다. 다음 문자가 지원됩니다.
  - 영숫자: (a-zA-Z0-9)
  - 기호: ('_' | '-')
- RegEx의 최대 길이는 200자입니다. 다음 문자가 지원됩니다.
  - 영숫자: (a-zA-Z0-9)
  - 기호: ('_' | '#' | '=' | '@' |'/' | ';' | ',' | '-' | ' ')
  - RegEx 예약 문자: ('^' | '$' | '?' | '[' | ']' | '{' | '}' | '|' | '\\' | '*' | '+' | '.')
- 사용자 지정 데이터 식별자는 관리형 데이터 식별자와 동일한 이름을 공유할 수 없습니다.
- 사용자 지정 데이터 식별자는 계정 수준 데이터 보호 정책 또는 로그 그룹 수준 데이터 보호 정책에서 지정할 수 있습니다. 관리형 데이터 식별자와 마찬가지로, 계정 수준 정책에 정의된 사용자 지정 데이터 식별자는 로그 그룹 수준 정책에 정의된 사용자 지정 데이터 식별자와 함께 작동합니다.

콘솔에서 사용자 지정 데이터 식별자 사용

CloudWatch 콘솔을 사용하여 데이터 보호 정책을 만들거나 편집할 때 사용자 지정 데이터 식별자를 지정하려면 데이터 식별자의 이름과 정규 표현식만 입력하면 됩니다. 예를 들어 이름을 입력하고 **Employee_ID EmployeeID-\d{9}** 정규 표현식으로 입력할 수 있습니다. 이 정규 표현식은 로그 이벤트를 탐지하고 뒤에 9개의 숫자가 있는 로그 이벤트를 탐지하고 EmployeeID- 마스킹합니다. 예제: EmployeeID-123456789

데이터 보호 정책에서 사용자 지정 데이터 식별자 사용

AWS CLI 또는 AWS API를 사용하여 사용자 지정 데이터 식별자를 지정하는 경우 데이터 보호 정책을 정의하는 데 사용되는 JSON 정책에 데이터 식별자 이름과 정규 표현식을 포함해야 합니다. 다음 데이터 보호 정책은 회사별 직원 ID를 포함하는 로그 이벤트를 탐지하고 마스킹합니다.

1. 데이터 보호 정책 내에 Configuration 블록을 생성합니다.
2. 사용자 지정 데이터 식별자에 Name을 입력합니다. 예를 들어 **EmployeeId**입니다.
3. 사용자 지정 데이터 식별자에 Regex을 입력합니다. 예를 들어 **EmployeeID-\d{9}**입니다. 이 정규 표현식은 뒤에 9자리 숫자가 EmployeeID- 포함된 로그 이벤트와 일치합니다. EmployeeID- 예제: EmployeeID-123456789
4. 정책 설명에서 다음 사용자 지정 데이터 식별자를 참조하세요.

```
{
    "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. (선택 사항) 필요에 따라 사용자 지정 데이터 식별자를 `Configuration` 블록에 계속 추가합니다. 데이터 보호 정책은 현재 최대 10개의 사용자 지정 데이터 식별자를 지원합니다.

필터를 사용하여 로그 이벤트에서 지표 생성

지표 필터를 하나 이상 생성하여 로그로 들어오는 CloudWatch 로그 데이터를 검색하고 필터링할 수 있습니다. 지표 필터는 CloudWatch Logs로 전송되는 로그 데이터에서 찾을 용어와 패턴을 정의합니다. CloudWatch 로그는 이러한 지표 필터를 사용하여 로그 데이터를 그래프로 표시하거나 경보를 설정할 수 있는 수치 CloudWatch 지표로 변환합니다.

로그 필터에서 지표를 생성할 때 차원과 단위를 메트릭에 할당하도록 선택할 수도 있습니다. 단위를 지정하는 경우 필터를 생성할 때 올바른 단위를 지정해야 합니다. 필터 단위를 나중에 변경하는 것은 효과가 없습니다.

> **Note**
>
> 지표 필터는 표준 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

이러한 지표를 보거나 경보를 설정할 때는 백분위수 CloudWatch 통계를 비롯한 모든 유형의 통계를 사용할 수 있습니다.

> **Note**
>
> 지표 값이 음수가 아닌 경우에만 지표의 백분위수 통계가 지원됩니다. 지표 필터를 음수를 보고할 수 있도록 설정하면 값이 음수인 경우에는 해당 지표의 백분위수 통계를 사용할 수 없습니다. 자세한 내용은 백분위수를 참조하세요.

필터는 데이터를 소급해서 필터링하지 않습니다. 필터는 필터가 생성된 이후에 발생한 이벤트에 대한 지표 데이터 요소만 게시합니다. 필터링된 결과는 처음 50개 줄을 반환하는데, 필터링된 결과에 대한 타임스탬프가 지표 생성 시간보다 이른 경우에는 결과가 표시되지 않습니다.

내용

- 지표 필터 삭제

개념

각 지표 필터는 다음과 같은 키 요소들로 이루어져 있습니다.

기본값

로그를 수집했지만 일치하는 로그를 찾을 수 없는 기간 동안 지표 필터에 보고되는 값입니다. 이 값을 0으로 설정하면 이러한 모든 기간에서 데이터가 보고되어 일치하는 데이터가 없는 기간 때문에 지표가 '불규칙'해지는 것을 방지할 수 있습니다. 1분 동안 로그가 수집되지 않은 경우에는 아무 값도 보고되지 않습니다.

지표 필터로 생성한 지표에 차원을 할당하는 경우 해당 지표에 기본값을 할당할 수 없습니다.

차원

차원은 지표를 추가로 정의하는 키-값 쌍입니다. 지표 필터에서 생성된 지표에 차원을 할당할 수 있습니다. 차원은 지표에 대한 고유한 식별자의 일부이므로 로그에서 고유한 이름/값 쌍을 추출할 때마다 해당 지표의 새로운 변형이 생성되는 것입니다.

필터 패턴

Logs가 각 CloudWatch 로그 이벤트의 데이터를 해석하는 방법을 상징적으로 설명하는 것입니다. 예를 들어 로그 항목에는 타임스탬프, IP 주소, 문자열 등이 포함될 수 있습니다. 이러한 패턴을 사용하여 로그 파일에서 검색할 내용을 지정합니다.

척도 이름

모니터링된 로그 정보를 게시해야 하는 CloudWatch 지표의 이름. 예를 들어, 라는 지표에 게시할 수 ErrorCount 있습니다.

지표 네임스페이스

새 CloudWatch 지표의 대상 네임스페이스.

지표 값

일치하는 로그가 발견될 때마다 지표에 게시하는 숫자 값입니다. 예를 들어, 특정 단어(예: "Error")의 출현 횟수를 계산할 경우 각 출현마다 이 값이 "1"이 됩니다. 전송된 바이트를 계산하는 경우 로그 이벤트에서 발견된 실제 바이트 수만큼 증가시킬 수 있습니다.

지표 필터에 대한 필터 패턴 구문

> **Note**
>
> 측정항목 필터의 차이점 - CloudWatch Logs Insights 쿼리
> 지표 필터는 일치하는 CloudWatch 로그를 찾을 때마다 지표 필터에 지정된 숫자 값이 추가된다는 점에서 Logs Insights 쿼리와 다릅니다. 자세한 정보는 지표 필터에 대한 지표 값 구성을 참조하세요.
> Amazon CloudWatch Logs Insights 쿼리 언어를 사용하여 로그 그룹을 쿼리하는 방법에 대한 자세한 내용은 을 참조하십시오CloudWatch 로그 인사이트 쿼리 구문.
> 일반 필터 패턴 예제
> 지표 필터와 구독 필터 및 필터 로그 이벤트에 적용할 수 있는 일반 필터 패턴 구문에 대한 자세한 내용은 지표 필터, 구독 필터 및 필터 로그 이벤트의 필터 패턴 구문을 참조하세요. 여기에 다음 예제가 포함되어 있습니다.
>
> - 지원되는 정규식 구문
> - 비정형 로그 이벤트에서 일치하는 용어 검색
> - JSON 로그 이벤트에서 일치하는 단어 검색
> - 공백으로 구분된 로그 이벤트에서 일치하는 용어 검색

지표 필터를 사용하면 Logs로 CloudWatch 들어오는 로그 데이터를 검색 및 필터링하고, 필터링된 로그 데이터에서 지표 관찰을 추출하고, 데이터 포인트를 CloudWatch Logs 지표로 변환할 수 있습니다. CloudWatch Logs로 전송되는 로그 데이터에서 검색할 용어와 패턴을 정의합니다. 지표 필터는 로그 그룹에 할당이 되고, 로그 그룹에 할당된 모든 필터는 로그 스트림에 적용됩니다.

지표 필터와 일치하는 용어가 검색되면 지표 개수가 지정된 숫자 값만큼 증가합니다. 예를 들어 지표 필터를 생성하여 로그 이벤트에서 ERROR라는 단어의 개수를 계산할 수 있습니다.

지표에 측정 단위 및 차원을 할당할 수 있습니다. 예를 들어 로그 이벤트 내 ERROR라는 단어의 개수를 계산하는 지표 필터를 만드는 경우, `ErrorCode`라는 차원을 지정하여 단어 ERROR를 포함하는 총 로그 이벤트 수를 표시하고 보고된 오류 코드를 기준으로 데이터를 필터링할 수 있습니다.

> **Tip**
>
> 지표에 측정 단위를 할당하는 경우 올바른 단위를 지정해야 합니다. 나중에 단위를 변경하면 변경 사항이 적용되지 않을 수 있습니다. CloudWatch 지원하는 장치의 전체 목록은 Amazon CloudWatch API 참조를 참조하십시오 MetricDatum.

주제

지표 필터에 대한 지표 값 구성

지표 필터를 생성할 때 필터 패턴을 정의하고 지표의 값과 기본값을 지정합니다. 지표 값을 숫자, 명명된 식별자 또는 숫자 식별자로 설정할 수 있습니다. 기본값을 지정하지 않으면 메트릭 필터가 일치하는 값을 찾지 못해도 데이터를 보고하지 않습니다. CloudWatch 값이 0인 경우에도 기본값을 지정하는 것이 좋습니다. 기본값을 설정하면 데이터를 더 정확하게 CloudWatch 보고할 수 있고 지표가 잘못 집계되는 것을 CloudWatch 방지할 수 있습니다. CloudWatch 1분마다 지표 값을 집계하고 보고합니다.

지표 필터가 로그 이벤트에서 일치하는 항목을 찾으면 지표 값을 기준으로 지표 수가 증가합니다. 지표 필터가 일치하는 항목을 찾지 못하면 지표의 기본값을 CloudWatch 보고합니다. 예를 들어 로그 그룹이 1분마다 두 개의 레코드를 게시하고, 지표 값은 1, 기본값은 0입니다. 처음 1분 동안 지표 필터가 두 로그 레코드 모두에서 일치하는 항목을 찾을 경우 이 기간(분)의 지표 값은 2입니다. 두 번째 1분 동안 지표 필터가 두 레코드 모두에서 일치하는 항목을 찾지 못할 경우 해당 기간(분)의 기본값은 0입니다. 지표 필터가 생성하는 지표에 차원을 할당하는 경우 해당 지표에 대한 기본값을 지정할 수 없습니다.

정적 값 대신 로그 이벤트에서 추출한 값으로 지표를 증가시키도록 지표 필터를 설정할 수도 있습니다. 자세한 내용은 로그 이벤트의 값을 사용하여 지표 값 증가 섹션을 참조하세요.

JSON 값 또는 공백으로 구분된 로그 이벤트의 지표로 차원 게시

CloudWatch 콘솔 또는 AWS CLI를 사용하여 JSON 및 공백으로 구분된 로그 이벤트가 생성하는 지표와 함께 차원을 게시하는 지표 필터를 생성할 수 있습니다. 차원은 이름/값 쌍이며 JSON 및 공백으로 구분된 필터 패턴에만 사용할 수 있습니다. JSON 및 공백으로 구분된 지표 필터를 최대 3개의 차원으로 생성할 수 있습니다. 차원에 대한 자세한 내용 및 지표에 차원을 할당하는 방법에 대한 자세한 내용은 다음 섹션을 참조하세요.

- Amazon CloudWatch 사용 설명서의 치수
- 예: Amazon CloudWatch Logs 사용 설명서에서 Apache 로그에서 필드를 추출하고 차원을 할당합니다.

> **Important**
>
> 차원에는 사용자 지정 지표와 동일한 요금을 수집하는 값이 포함되어 있습니다. 예기치 않은 요금이 청구되는 것을 방지하려면 `IPAddress` 또는 `requestID`처럼 높은 카디널리티 필드를 차원으로 지정하지 마세요.
>
> 로그 이벤트에서 지표를 추출하면 사용자 지정 지표에 대한 요금이 청구됩니다. 실수로 높은 요금이 부과되는 것을 방지하기 위해 일정 시간 내에 지정한 차원에 대해 1000개의 다른 이름/값 쌍을 생성하는 경우 Amazon이 지표 필터를 사용 중지할 수 있습니다.
>
> 예상 요금을 알려주는 결제 경보를 생성할 수 있습니다. 자세한 내용은 예상 AWS 요금 모니터링을 위한 결제 경보 생성을 참조하십시오.

JSON 로그 이벤트에서 지표로 차원 게시

다음 예제에는 JSON 지표 필터에서 차원을 지정하는 방법을 설명하는 코드 조각이 포함되어 있습니다.

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**
>
> 예제 JSON 로그 이벤트로 예제 지표 필터를 테스트하는 경우, 예제 JSON 로그를 한 줄에 입력해야 합니다.

Example: Metric filter

지표 필터는 JSON 로그 이벤트에 속성 `eventType` 및 `"sourceIPAddress"`가 포함될 때마다 지표를 증가시킵니다.

```
{ $.eventType = "*" && $.sourceIPAddress != 123.123.* }
```

JSON 지표 필터를 생성할 때 지표 필터에 있는 어떤 속성도 차원으로 지정할 수 있습니다. 예를 들어, `eventType`을 차원으로 설정하려면 다음을 사용합니다.

```
"eventType" : $.eventType
```

예제 지표에는 이름이 `"eventType"`인 차원이 포함되어 있고, 예제 로그 이벤트의 차원 값은 `"UpdateTrail"`입니다.

공백으로 구분된 로그 이벤트에서 지표로 차원 게시

다음 예제에는 공백으로 구분된 지표 필터에서 차원을 지정하는 방법을 설명하는 코드 조각이 포함되어 있습니다.

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]
```

지표 필터는 공백으로 구분된 로그 이벤트에 필터에 지정된 필드가 포함된 경우 지표를 증가시킵니다. 예를 들어 지표 필터는 공백으로 구분된 로그 이벤트 예제에서 다음 필드 및 값을 찾습니다.

```
{
   "$bytes": "1534",
   "$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"
}
```

공백으로 구분된 지표 필터를 생성할 때 지표 필터에 있는 어떤 속성도 차원으로 지정할 수 있습니다. 예를 들어, `server`을 차원으로 설정하려면 다음을 사용합니다.

```
"server" : $server
```

예제 지표 필터에는 이름이 `server`인 차원이 있고, 예제 로그 이벤트의 차원 값은 `"Prod"`입니다.

Example: Match terms with AND (&&) and OR (||)

논리 연산자 AND('&&') 및 OR('||')를 사용하여 조건을 포함하는 공백으로 구분된 지표 필터를 생성할 수 있습니다. 다음 지표 필터는 이벤트의 첫 번째 단어가 WARN을 포함하는 모든 상위 문자열 또는 ERROR인 로그 이벤트를 반환합니다.

```
[w1=ERROR || w1=%WARN%, w2]
```

로그 이벤트의 값을 사용하여 지표 값 증가

로그 이벤트에서 찾은 숫자 값을 게시하는 지표 필터를 생성할 수 있습니다. 이 섹션의 절차에서는 다음 예제 지표 필터를 사용하여 JSON 로그 이벤트의 숫자 값을 지표에 게시하는 방법을 보여 줍니다.

```
{ $.latency = * } metricValue: $.latency
```

로그 이벤트에 값을 게시하는 지표 필터 생성

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 로그(Logs)를 선택한 다음, 로그 그룹(Log groups)을 선택합니다.
3. 로그 그룹을 선택하거나 생성합니다.

   로그 그룹을 생성하는 방법에 대한 자세한 내용은 Amazon Logs 사용 설명서의 CloudWatch Logs에서 CloudWatch 로그 그룹 생성을 참조하십시오.
4. 작업(Actions)을 선택한 후 지표 필터 생성(Create metric filter)을 선택합니다.
5. 필터 패턴(Filter Pattern)에 **`{ $.latency = * }`**를 입력하고 다음(Next)을 선택합니다.
6. 지표 이름(Metric Name)에 myMetric을 입력합니다.
7. 지표 값(Metric Value)에 **`$.latency`**를 입력합니다.
8. (선택 사항) 기본값(Default Value)에 0을 입력하고 다음(Next)을 선택합니다.

   값이 0인 경우에도 기본값을 지정하는 것이 좋습니다. 기본값을 설정하면 데이터를 더 정확하게 CloudWatch 보고하고 지표가 불안정하게 집계되는 것을 CloudWatch 방지할 수 있습니다. CloudWatch 1분마다 지표 값을 집계하고 보고합니다.
9. 지표 필터 생성(Create metric filter)을 선택합니다.

예제 지표 필터가 예제 JSON 로그 이벤트의 용어 `"latency"`와 일치하고 숫자 값 50을 지표 myMetric에 게시합니다.

```
{
"latency": 50,
"requestType": "GET"
```

```
}
```

지표 필터 생성

다음 절차 및 예제에서는 지표 필터를 생성하는 방법을 보여줍니다.

예제

- 로그 그룹에 대한 지표 필터 생성
- 예제: 로그 이벤트 수 계산
- 예제: 단어의 출현 횟수 계산
- 예제: HTTP 404 코드 수 계산
- 예제: HTTP 4xx 코드 수 계산
- 예제: Apache 로그에서 필드 추출 후 차원 할당

로그 그룹에 대한 지표 필터 생성

로그 그룹에 대한 지표 필터를 만들려면 다음 단계를 따릅니다. 지표는 해당 몇 개의 데이터 포인트가 있기 전에는 표시되지 않습니다.

콘솔을 CloudWatch 사용하여 지표 필터를 만들려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 로그(Logs)를 선택한 다음, 로그 그룹(Log groups)을 선택합니다.
3. 로그 그룹의 이름을 선택합니다.
4. `Actions`를 선택한 다음 Create metric filter(지표 필터 생성)를 선택합니다.
5. Filter pattern(필터 패턴)에 필터 패턴을 입력합니다. 자세한 정보는 지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail에 대한 필터 패턴 구문을 참조하세요.
6. (선택 사항) 필터 패턴을 테스트하려면 Test Pattern(테스트 패턴)에 패턴을 테스트할 로그 이벤트를 하나 이상 입력합니다. 각 로그 이벤트는 한 줄에 형식이 지정되어야 합니다. 줄 바꿈은 Log event messages(로그 이벤트 메시지) 상자에서 로그 이벤트를 구분하는 데 사용됩니다.
7. Next(다음)를 선택하고 지표 필터의 이름을 입력합니다.
8. 메트릭 세부 정보에서 메트릭 네임스페이스에 메트릭이 게시될 CloudWatch 네임스페이스의 이름을 입력합니다. 네임스페이스가 아직 없는 경우 Create new(새로 생성)가 선택되는지 확인하세요.
9. 지표 이름에 새 지표의 이름을 입력합니다.

10. 메트릭 필터가 필터에 포함된 키워드의 발생을 계산하는 경우 지표 값에 1을 입력합니다. 이렇게 하면 키워드 중 하나를 포함하는 각 로그 이벤트에 대해 메트릭이 1씩 증가합니다.

    또는 토큰(예: **$size**)을 입력합니다. 이렇게 하면 size 필드가 포함된 모든 로그 이벤트에 대해 size 필드의 수치 값만큼씩 지표가 증가합니다.

11. (선택 사항) 단위에서 지표에 할당할 단위를 선택합니다. 단위를 지정하지 않을 경우 단위는 None으로 설정됩니다.

12. (선택 사항) 지표에 대한 세 개의 차원에 대해 각각 이름과 토큰을 입력합니다. 지표 필터가 생성하는 지표에 차원을 할당하는 경우 해당 지표에 대한 기본값을 할당할 수 없습니다.

    > **Note**
    >
    > 차원은 JSON 또는 공백으로 구분된 지표 필터에서만 지원됩니다.

13. 지표 필터 생성을 선택합니다. 생성한 지표 필터를 탐색 창에서 찾을 수 있습니다. Logs(로그)를 선택한 후 Log groups(로그 그룹)를 선택합니다. 지표 필터를 생성한 로그 그룹의 이름을 선택한 다음 Metric filters(지표 필터) 탭을 선택합니다.

예제: 로그 이벤트 수 계산

가장 간단한 유형의 로그 이벤트 모니터링은 발생하는 로그 이벤트의 수를 계산하는 것입니다. 모든 이벤트의 수를 유지하거나 "하트비트" 스타일 모니터를 생성하거나 단순히 지표 필터 생성을 연습하기 위해 계산을 원할 수 있습니다.

다음 CLI 예제에서는 MyAppAccessCount 라는 메트릭 필터를 로그 그룹 MyApp /access.log 에 적용하여 네임스페이스에 EventCount 메트릭을 생성합니다. CloudWatch MyNamespace 이 필터는 모든 로그 이벤트 콘텐츠와 일치하며 지표를 "1"씩 늘리도록 구성되어 있습니다.

콘솔을 사용하여 메트릭 필터를 만들려면 CloudWatch

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹의 이름을 선택합니다.
4. 작업, 지표 필터 생성을 선택합니다.
5. 필터 패턴 및 테스트할 로그 데이터 선택을 비워 둡니다.
6. 다음을 선택한 후 필터 이름에 **EventCount**를 입력합니다.

7. 지표 세부 정보의 지표 네임스페이스에 **MyNameSpace**를 입력합니다.
8. 지표 이름에 **MyAppEventCount**를 입력합니다.
9. 지표 값이 1인지 확인합니다. 이는 모든 로그 이벤트에 대해 개수가 1씩 증가하도록 지정합니다.
10. 기본값에 0을 입력하고 다음을 선택합니다. 기본값을 지정할 경우 로그 이벤트가 발생하지 않는 기간에도 데이터가 보고되므로 때때로 데이터가 존재하지 않아 지표가 불규칙해지는 것을 방지할 수 있습니다.
11. 지표 필터 생성을 선택합니다.

를 사용하여 메트릭 필터를 만들려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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
```

모든 이벤트 데이터를 게재하여 이 새로운 정책을 테스트할 수 있습니다. 지표에 게시된 데이터 포인트를 확인해야 MyAppAccessEventCount 합니다.

를 사용하여 이벤트 데이터를 게시하려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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"
```

예제: 단어의 출현 횟수 계산

로그 이벤트에는 연산의 성공 또는 실패 횟수같이 계산이 필요한 중요한 메시지가 종종 포함됩니다. 예를 들어 지정된 연산이 실패하면 오류가 발생하고 로그 파일에 오류가 기록될 수 있습니다. 이들 항목을 모니터링하여 오류의 트렌드를 이해하고 싶을 수 있습니다.

아래 예제에서는 Error라는 단어를 모니터링하기 위한 지표 필터가 생성됩니다. 정책이 생성되어 로그 그룹 MyApp /message.log 에 추가되었습니다. CloudWatch 로그는 오류가 포함된 모든 이벤트에 대해 값이 " ErrorCount 1"인 데이터 포인트를 MyApp/message.log 네임스페이스의 CloudWatch 사용자 지정 지표에 게시합니다. Error라는 단어가 포함된 이벤트가 없으면 값 0이 게시됩니다. CloudWatch콘솔에서 이 데이터를 그래프로 표시할 때는 반드시 합계 통계를 사용하십시오.

지표 필터를 만든 후 콘솔에서 지표를 볼 수 있습니다. CloudWatch 보려는 지표를 선택할 때 로그 그룹 이름과 일치하는 지표 네임스페이스를 선택합니다. 자세한 내용은 사용 가능한 지표 보기를 참조하세요.

CloudWatch 콘솔을 사용하여 지표 필터를 만들려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹의 이름을 선택합니다.
4. 작업, 지표 필터 생성을 선택합니다.
5. 필터 패턴에 **`Error`**를 입력합니다.

> **Note**
>
> 필터 패턴의 모든 항목들은 대소문자를 구분합니다.

6. (선택 사항) 필터 패턴을 테스트하려면 테스트 패턴에 패턴을 테스트하는 데 사용할 로그 이벤트를 하나 이상 입력합니다. 줄 바꿈은 로그 이벤트 메시지 상자에서 로그 이벤트를 구분할 때 사용하므로 각 로그 이벤트는 한 줄을 넘지 않아야 합니다.
7. 다음을 선택한 후 지표 할당 페이지에서 필터 이름에 **`MyAppErrorCount`**를 입력합니다.
8. 메트릭 세부 정보에서 메트릭 네임스페이스에 다음을 입력합니다. MyNameSpace
9. 지표 이름에 ErrorCount를 입력합니다.
10. 지표 값이 1인지 확인합니다. 이는 "Error"를 포함하는 모든 로그 이벤트에 대해 개수가 1씩 증가하도록 지정합니다.
11. 기본값에 0을 입력하고 다음을 선택합니다.
12. 지표 필터 생성을 선택합니다.

를 사용하여 메트릭 필터를 만들려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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
```

메시지에 "Error"라는 단어가 포함된 이벤트를 게재하여 이 새로운 정책을 테스트할 수 있습니다.

를 사용하여 이벤트를 게시하려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다. 패턴은 대소문자를 구분한다는 점을 유의하세요.

```
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"
```

예제: HTTP 404 코드 수 계산

CloudWatch 로그를 사용하면 Apache 서버가 HTTP 404 응답을 반환하는 횟수 (페이지를 찾을 수 없음) 에 대한 응답 코드인 횟수를 모니터링할 수 있습니다. 사이트 방문자들이 원하는 리소스를 찾지 못하는 빈도를 파악하기 위해 모니터링을 원할 수 있습니다. 로그 레코드는 각 로그 이벤트(사이트 방문)에 대해 다음 정보를 포함하도록 구성되었다고 가정합니다.

- 요청자 IP 주소
- RFC 1413 ID
- 사용자 이름
- Timestamp
- 요청된 리소스와 프로토콜이 포함된 요청 메서드
- 요청할 HTTP 응답 코드
- 요청 시 전송되는 바이트

예를 들어 다음과 같은 형태일 수 있습니다.

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 404 2326
```

다음 예제에서 알 수 있듯이, HTTP 404 오류에서 해당 구조의 이벤트와 매칭을 시도하는 규칙을 지정할 수 있습니다.

콘솔을 사용하여 지표 필터를 만들려면 CloudWatch

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 작업, 지표 필터 생성을 선택합니다.
4. 필터 패턴에 **[IP, UserInfo, User, Timestamp, RequestInfo, StatusCode=404, Bytes]**를 입력합니다.
5. (선택 사항) 필터 패턴을 테스트하려면 테스트 패턴에 패턴을 테스트하는 데 사용할 로그 이벤트를 하나 이상 입력합니다. 줄 바꿈은 로그 이벤트 메시지 상자에서 로그 이벤트를 구분할 때 사용하므로 각 로그 이벤트는 한 줄을 넘지 않아야 합니다.
6. 다음을 선택하고 필터 이름에 HTTP404Errors를 입력합니다.
7. 지표 세부 정보의 지표 네임스페이스에 **MyNameSpace**를 입력합니다.
8. 지표 이름에 **ApacheNotFoundErrorCount**를 입력합니다.
9. 지표 값이 1인지 확인합니다. 이는 모든 404 오류 이벤트에 대해 개수가 1씩 증가하도록 지정합니다.
10. 기본값에 0을 입력하고 다음을 선택합니다.
11. 지표 필터 생성을 선택합니다.

를 사용하여 메트릭 필터를 만들려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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
```

이 예제에서는 왼쪽/오른쪽 대괄호, 큰따옴표, 문자열 404 같은 리터럴 문자가 사용되었습니다. 패턴은 모니터링하려는 로그 이벤트의 전체 로그 이벤트 메시지와 일치해야 합니다.

describe-metric-filters 명령을 사용하여 지표 필터가 생성되었는지 확인할 수 있습니다. 다음과 유사한 출력 화면이 표시되어야 합니다.

```
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]"
        }
    ]
}
```

수동으로 몇 가지 이벤트를 게재할 수 있습니다.

```
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"
```

이러한 샘플 로그 이벤트를 업로드한 직후 CloudWatch 콘솔에서 이름이 인 메트릭을 검색할 수 ApacheNotFoundErrorCount 있습니다.

예제: HTTP 4xx 코드 수 계산

이전 예제에서와 마찬가지로 사용자는 웹 서비스 액세스 로그를 모니터링하고 HTTP 응답 코드 수준을 모니터링하려고 할 수 있습니다. 예를 들어 HTTP 400 수준 오류를 모두 모니터링하고 싶을 수 있습니다. 그러나 모든 반환 코드에 새로운 지표 필터를 지정하고 싶지 않을 수 있습니다.

다음 예제는 예제: HTTP 404 코드 수 계산 예제에서 Apache 액세스 로그 형식을 사용하여 액세스 로그로부터 400 수준의 모든 HTTP 코드 응답을 포함하는 지표를 생성하는 방법을 보여줍니다.

CloudWatch 콘솔을 사용하여 지표 필터를 만들려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. Apache 서버의 로그 그룹 이름을 선택합니다.
4. 작업, 지표 필터 생성을 선택합니다.
5. 필터 패턴에 **[ip, id, user, timestamp, request, status_code=4*, size]**를 입력합니다.
6. (선택 사항) 필터 패턴을 테스트하려면 테스트 패턴에 패턴을 테스트하는 데 사용할 로그 이벤트를 하나 이상 입력합니다. 줄 바꿈은 로그 이벤트 메시지 상자에서 로그 이벤트를 구분할 때 사용하므로 각 로그 이벤트는 한 줄을 넘지 않아야 합니다.
7. 다음을 선택한 후 필터 이름에 **HTTP4xxErrors**를 입력합니다.
8. 지표 세부 정보의 지표 네임스페이스에 **MyNameSpace**를 입력합니다.
9. 지표 이름에 HTTP4xxErrors를 입력합니다.
10. 지표 값에 1을 입력합니다. 이는 4xx 오류를 포함하는 모든 로그 이벤트에 대해 개수가 1씩 증가하도록 지정합니다.
11. 기본값에 0을 입력하고 다음을 선택합니다.
12. 지표 필터 생성을 선택합니다.

를 사용하여 메트릭 필터를 만들려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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
```

PUT 이벤트 호출에서 다음 데이터를 사용하여 이 규칙을 테스트할 수 있습니다. 이전 예제에서 모니터링 규칙을 제거하지 않았다면 서로 다른 두 개의 지표가 생성됩니다.

```
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
```

예제: Apache 로그에서 필드 추출 후 차원 할당

때로는 수를 계산하는 대신 지표 값으로 개별 로그 이벤트 내의 값을 사용하는 것이 도움이 됩니다. 이 예제는 Apache 웹 서버에서 전송된 바이트를 측정하는 지표를 생성하기 위해 추출 규칙을 만드는 방법을 보여줍니다.

이 예제에서는 생성 중인 지표에 차원을 할당하는 방법도 보여줍니다.

콘솔을 사용하여 지표 필터를 만들려면 CloudWatch

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. Apache 서버의 로그 그룹 이름을 선택합니다.
4. 작업, 지표 필터 생성을 선택합니다.
5. 필터 패턴에 **[ip, id, user, timestamp, request, status_code, size]**를 입력합니다.
6. (선택 사항) 필터 패턴을 테스트하려면 테스트 패턴에 패턴을 테스트하는 데 사용할 로그 이벤트를 하나 이상 입력합니다. 줄 바꿈은 로그 이벤트 메시지 상자에서 로그 이벤트를 구분할 때 사용하므로 각 로그 이벤트는 한 줄을 넘지 않아야 합니다.
7. 다음을 선택한 후 필터 이름에 **size**를 입력합니다.
8. 지표 세부 정보의 지표 네임스페이스에 **MyNameSpace**를 입력합니다. 새 네임스페이스이기 때문에 새로 생성이 선택되어 있습니다.
9. 지표 이름에 **BytesTransferred**를 입력합니다.
10. 지표 값에 **$size**를 입력합니다.
11. 단위로 바이트를 선택합니다.
12. 차원 이름(Dimension Name)에 **IP**를 입력합니다.
13. 차원 값(Dimension Value)에 **$ip**를 입력하고 다음(Next)을 선택합니다.
14. 지표 필터 생성을 선택합니다.

다음을 사용하여 이 메트릭 필터를 만들려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
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**
>
> 이 명령에서는 이 형식을 사용하여 여러 차원을 지정합니다.
>
> ```
> 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}'
> ```

put-log-event 호출에서 다음 데이터를 사용하여 이 규칙을 테스트할 수 있습니다. 이전 예제에서 모니터링 규칙을 제거하지 않았다면 서로 다른 두 개의 지표가 생성됩니다.

```
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
```

지표 필터 나열

로그 그룹에 속한 모든 지표 필터를 나열할 수 있습니다.

CloudWatch 콘솔을 사용하여 지표 필터를 나열하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 콘텐츠 창에서 로그 그룹 목록의 지표 필터 열에서 필터 수를 선택합니다.

   로그 그룹 > Filters for(필터 대상) 화면에 해당 로그 그룹과 연관된 모든 지표 필터들이 나열됩니다.

를 사용하여 메트릭 필터를 나열하려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
aws logs describe-metric-filters --log-group-name MyApp/access.log
```

다음은 예 출력입니다.

```
{
    "metricFilters": [
        {
            "filterName": "HTTP404Errors",
            "metricTransformations": [
                {
                    "metricValue": "1",
                    "metricNamespace": "MyNamespace",
                    "metricName": "ApacheNotFoundErrorCount"
                }
            ],
            "creationTime": 1399277571078,
            "filterPattern": "[ip, id, user, timestamp, request, status_code=404,
 size]"
        }
    ]
}
```

지표 필터 삭제

이름과 소속 로그 그룹으로 정책을 식별할 수 있습니다.

CloudWatch 콘솔을 사용하여 지표 필터를 삭제하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 콘텐츠 창의 지표 필터 열에서 로그 그룹에 대한 지표 필터 개수를 선택합니다.
4. 지표 필터 화면에서 삭제할 필터 이름 오른쪽에 있는 확인란을 선택합니다. 그런 다음 삭제를 선택합니다.
5. 확인 메시지가 나타나면 삭제를 선택합니다.

를 사용하여 메트릭 필터를 삭제하려면 AWS CLI

명령 프롬프트에서 다음 명령을 실행합니다.

```
aws logs delete-metric-filter --log-group-name MyApp/access.log \
 --filter-name MyFilterName
```

구독을 통한 로그 데이터 실시간 처리

구독을 사용하여 Logs에서 CloudWatch 로그 이벤트의 실시간 피드에 액세스하여 Amazon Kinesis 스트림, Amazon Data Firehose 스트림과 같은 다른 서비스로 전송하거나 사용자 지정 처리, 분석을 AWS Lambda 수행하거나 다른 시스템으로 로드할 수 있습니다. 로그 이벤트가 수신 서비스로 전송되면 base64로 인코딩되고 gzip 형식으로 압축됩니다.

로그 이벤트에 대한 구독을 시작하려면 이벤트가 제공되는 수신 소스(예: Kinesis Data Streams 스트림)를 생성합니다. 구독 필터는 AWS 리소스로 전달되는 로그 이벤트를 필터링하는 데 사용할 필터 패턴과 일치하는 로그 이벤트를 보낼 위치에 대한 정보를 정의합니다.

계정 수준 및 로그 그룹 수준에서 구독을 생성할 수 있습니다. 각 계정에는 하나의 계정 수준 구독 필터가 있을 수 있습니다. 각 로그 그룹에는 구독 필터를 최대 2개까지 연결할 수 있습니다.

> **Note**
>
> 대상 서비스가 제한 예외 또는 재시도 가능한 서비스 예외 (예: HTTP 5xx) 와 같은 재시도 가능한 오류를 반환하는 경우 CloudWatch Logs는 최대 24시간 동안 전송을 계속 재시도합니다. CloudWatch 오류가 또는 같이 다시 시도할 수 없는 오류인 경우 Logs는 재전송을 시도하지 않습니다. AccessDeniedException ResourceNotFoundException 이러한 경우 구독 필터가 최대 10분 동안 비활성화된 후 Logs는 목적지로 CloudWatch 로그를 다시 전송하려고 시도합니다. 이 비활성화된 기간 동안에는 로그를 건너뛰게 됩니다.

CloudWatch 로그는 또한 구독으로 로그 이벤트를 전달하는 데 대한 CloudWatch 지표를 생성합니다. 자세한 정보는 CloudWatch 메트릭을 사용한 모니터링을 참조하세요.

또한 CloudWatch Logs 구독을 사용하여 Amazon OpenSearch Service 클러스터로 로그 데이터를 거의 실시간으로 스트리밍할 수 있습니다. 자세한 내용은 Amazon OpenSearch Service로의 스트리밍 CloudWatch 로그 데이터를 참조하십시오.

구독은 Standard 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

> **Note**
>
> 구독 필터는 로그 이벤트를 일괄 처리하여 전송을 최적화하고 대상으로의 통화량을 줄일 수 있습니다. 일괄 처리가 보장되지는 않지만 가능하면 일괄 처리가 사용됩니다.

내용

개념

각 구독 필터는 다음과 같은 키 요소들로 이루어져 있습니다.

필터 패턴

대상 리소스로 전달되는 내용을 제한하는 필터링 표현식과 함께 로그가 각 로그 이벤트의 데이터를 해석하는 방식을 CloudWatch 상징적으로 설명합니다. AWS 필터 패턴 구문에 대한 자세한 내용은 지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail에 대한 필터 패턴 구문 섹션을 참조하세요.

destination arn

구독 피드의 대상으로 사용하려는 Kinesis 데이터 스트림, Firehose 스트림 또는 Lambda 함수의 Amazon 리소스 이름 (ARN)

역할 ARN

선택한 대상에 데이터를 저장하는 데 필요한 권한을 CloudWatch 로그에 부여하는 IAM 역할. Lambda 목적지에는 이 역할이 필요하지 않습니다. 로그는 Lambda 함수 자체의 액세스 제어 설정에서 필요한 권한을 얻을 수 CloudWatch 있기 때문입니다.

배포

대상이 Amazon Kinesis Data Streams의 스트림일 때 해당 대상으로 로그 데이터를 배포하는 데 사용되는 방법입니다. 기본적으로 로그 스트림에 따라 로그 데이터가 그룹화됩니다. 보다 균등한 배포를 위해 로그 데이터를 무작위로 그룹화할 수 있습니다.

로그 그룹 수준 구독의 경우 다음과 같은 주요 요소도 포함됩니다.

log_group_name

구독 필터에 연결되는 로그 그룹입니다. 이 로그 그룹에 업로드되는 모든 로그 이벤트는 구독 필터를 따르게 되며, 필터와 일치하는 로그 이벤트는 일치하는 로그 이벤트를 수신하는 대상 서비스로 전송됩니다.

계정 수준 구독의 경우 다음과 같은 주요 요소도 포함됩니다.

선택 기준

계정 수준 구독 필터가 적용된 로그 그룹을 선택하는 데 사용되는 기준입니다. 이를 지정하지 않으면 계정 수준의 구독 필터가 계정의 모든 로그 그룹에 적용됩니다. 이 필드는 무한 로그 루프를 방지하는 데 사용됩니다. 무한 로그 루프 문제에 대한 자세한 내용은 을 참조하십시오로그 재귀 방지.

선택 기준의 크기 제한은 25KB입니다.

로그 그룹 수준 구독 필터

Kinesis Data Streams, Lambda 또는 Firehose와 함께 구독 필터를 사용할 수 있습니다. 구독 필터를 통해 수신 서비스로 전송되는 로그는 base64로 인코딩되고 gzip 형식으로 압축됩니다.

필터 및 패턴 구문을 사용하여 로그 데이터를 검색할 수 있습니다.

예

- 예제 1: Kinesis Data Streams에 대한 구독 필터
- 예 2: 다음을 포함하는 구독 필터 AWS Lambda
- 예제 3: Amazon Data Firehose를 사용한 구독 필터

예제 1: Kinesis Data Streams에 대한 구독 필터

다음 예제에서는 구독 필터를 이벤트가 포함된 로그 그룹과 연결합니다. AWS CloudTrail 구독 필터는 “루트” AWS 자격 증명으로 기록된 모든 활동을 Kinesis Data Streams의 RootAccess "" 스트림으로 전달합니다. AWS CloudTrail 이벤트를 CloudWatch 로그로 보내는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudWatch 로그로 CloudTrail 이벤트 전송을 참조하십시오.

> **Note**
>
> 스트림을 생성하기 전에 생성할 로그 데이터의 볼륨을 계산합니다. 이 볼륨을 처리하기에 충분한 샤드로 스트림을 생성해야 합니다. 스트림에 샤드가 충분하지 않을 경우에는 로그 스트림에서 병목 현상이 발생합니다. 스트림 볼륨 제한에 대한 자세한 내용은 할당량 및 제한을 참조하세요.
> 제한된 결과물은 최대 24시간 동안 재시도됩니다. 24시간이 지나면 실패한 결과물이 삭제됩니다.
> 제한의 위험을 완화하려면 다음 단계를 수행하세요.
>
> - PutSubscriptionFilter또는 `random` 를 사용하여 구독 필터를 생성할 `distribution` 시기를 put-subscription-filter지정하십시오. 기본적으로 스트림 필터 배포는 로그 스트림별로 이루어지므로 병목 현상이 발생할 수 있습니다.
> - 측정치를 사용하여 CloudWatch 스트림을 모니터링하세요. 이렇게 하면 제한을 식별하고 그에 따라 구성을 조정할 수 있습니다. 예를 들어, `DeliveryThrottling` 지표를 사용하여 구독 대상으로 데이터를 전달할 때 로그가 병목 현상이 발생한 CloudWatch 로그 이벤트 수를 추적할 수 있습니다. 모니터링에 대한 자세한 내용은 CloudWatch 메트릭을 사용한 모니터링 섹션을 참조하세요.
> - Kinesis Data Streams의 스트림에 대해 온디맨드 용량 모드를 사용합니다. 온디맨드 모드는 워크로드가 확장 또는 축소될 때 즉시 워크로드를 수용합니다. 온디맨드 용량 모드에 대한 자세한 내용은 온디맨드 모드를 참조하세요.
> - Kinesis Data Streams의 스트림 용량과 일치하도록 CloudWatch 구독 필터 패턴을 제한하십시오. 스트림에 너무 많은 데이터를 전송하는 경우 필터 크기를 줄이거나 필터 기준을 조정해야 할 수 있습니다.

Kinesis Data Streams에 대한 구독 필터를 생성하려면

1. 다음 명령을 사용하여 대상 스트림을 생성합니다.

```
$ C:\>  aws kinesis create-stream --stream-name "RootAccess" --shard-count 1
```

2. 스트림이 활성 상태가 될 때까지 기다립니다(1~2분 정도 소요). 다음 Kinesis Data Streams 설명-스트림 명령을 사용하여 확인할 수 있습니다. StreamDescription StreamStatus속성. 또한 나중 단계에서 필요하므로 StreamDescription.streamArn 값을 기록해 두십시오.

```
aws kinesis describe-stream --stream-name "RootAccess"
```

다음은 예 출력입니다.

```
{
    "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. 스트림에 데이터를 넣을 수 있는 CloudWatch Logs 권한을 부여하는 IAM 역할을 생성하십시오. 먼저 신뢰 정책을 파일로 생성해야 합니다(예: ~/TrustPolicyForCWL-Kinesis.json). 텍스트 편집기를 사용하여 이 정책을 생성하세요. IAM 콘솔을 사용하여 정책을 생성하지 마세요.

   이 정책에는 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키가 포함되어 있습니다. 자세한 내용은 혼동된 대리자 방지 섹션을 참조하세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요할 수 있기 때문에 반환된 Role.Arn 값을 적어둡니다.

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

다음은 출력의 예제입니다.

```
{
    "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. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 권한 정책을 파일로 생성합니다(예: ~/PermissionsForCWL-Kinesis.json). 텍스트 편집기를 사용하여 이 정책을 생성하세요. IAM 콘솔을 사용하여 정책을 생성하지 마세요.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/RootAccess"
```

```
    }
  ]
}
```

6. 다음 put-role-policy명령을 사용하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. 스트림이 Active 상태가 되고 IAM 역할을 생성한 후 CloudWatch Logs 구독 필터를 생성할 수 있습니다. 그 즉시 구독 필터는 실시간으로 선택한 로그 그룹에서 스트림으로 로그 데이터를 이동시키기 시작합니다.

```
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. 구독 필터를 설정하면 CloudWatch Logs는 필터 패턴과 일치하는 모든 수신 로그 이벤트를 스트림에 전달합니다. Kinesis Data Streams 샤드 반복자를 확보하고 Kinesis Data Streams get-records 명령을 사용해 몇몇 Kinesis Data Streams 레코드를 가져와서 이러한 작업이 수행되고 있는지 확인할 수 있습니다.

```
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"
```

이 호출을 몇 차례 반복해야 Kinesis Data Streams가 데이터 반환을 시작할 수 있습니다.

레코드 어레이에서 응답을 확인할 수 있습니다. Kinesis Data Streams 레코드의 데이터 속성은 base64로 인코딩되고 gzip 형식으로 압축됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<Content of Data>" | base64 -d | zcat
```

디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

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

```
}
```

위의 데이터 구조에서 키 요소는 다음과 같습니다.

owner

최초 로그 데이터의 AWS 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

messageType

데이터 메시지는 "DATA_MESSAGE" 유형을 사용합니다. 가끔 CloudWatch 로그에서 “CONTROL_MESSAGE” 유형의 Kinesis Data Streams 레코드를 내보낼 수 있는데, 이는 주로 대상에 도달할 수 있는지 확인하기 위한 것입니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. "ID" 속성은 모든 로그 이벤트의 고유 식별자입니다.

예 2: 다음을 포함하는 구독 필터 AWS Lambda

이 예시에서는 AWS Lambda 함수에 CloudWatch 로그 데이터를 보내는 Logs 구독 필터를 만들어 보겠습니다.

> **Note**
>
> Lambda 함수를 생성하기 전에 생성할 로그 데이터의 볼륨을 계산합니다. 이 볼륨을 처리할 수 있는 함수를 생성해야 합니다. 함수에 볼륨이 충분하지 않을 경우에는 로그 스트림에서 병목 현상이 발생합니다. AWS Lambda 제한에 대한 자세한 내용은 Lambda 제한을 참조하세요.

Lambda에 대한 구독 필터를 생성하려면

1. AWS Lambda 함수를 생성하세요.

   Lambda 실행 역할이 설정되었는지 확인합니다. 자세한 내용은 AWS Lambda 개발자 안내서의 2.2단계: IAM 역할 생성(실행 역할)을 참조하세요.

2. 다음 콘텐츠로 텍스트 편집기를 열고 `helloWorld.js`라는 파일을 생성합니다.

```
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. helloWorld.js 파일을 압축하고 `helloWorld.zip`라는 이름으로 저장합니다.

4. 역할이 첫 단계에서 설정한 Lambda 실행 역할인 경우에는 다음 명령을 사용하세요.

```
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. CloudWatch Logs에 함수를 실행할 권한을 부여하십시오. 다음 명령을 사용하여 자리 표시자 계정을 자체 계정으로, 자리 표시자 그룹을 처리할 로그 그룹으로 바꿉니다.

```
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. 다음 명령을 사용하여 구독 필터를 생성하여 자리 표시자 계정을 자체 계정으로, 자리 표시자 그룹을 처리할 로그 그룹으로 바꿉니다.

```
aws logs put-subscription-filter \
    --log-group-name myLogGroup \
    --filter-name demo \
    --filter-pattern "" \
    --destination-arn arn:aws:lambda:region:123456789123:function:helloworld
```

7. (선택 사항) 샘플 로그 이벤트를 사용하여 테스트합니다. 명령 프롬프트에서 다음 명령을 실행하여 구독된 스트림으로 간단한 로그 메시지를 보냅니다.

   Lambda 함수의 출력을 보려면 /aws/lambda/helloworld에 출력이 표시되는 Lambda 함수를 탐색합니다.

```
aws logs put-log-events --log-group-name myLogGroup --log-stream-name stream1 --
log-events "[{\"timestamp\":<CURRENT TIMESTAMP MILLIS> , \"message\": \"Simple
 Lambda Test\"}]"
```

   Lambda 어레이에서 응답을 확인할 수 있습니다. Lambda 레코드의 데이터 속성은 base64로 인코딩되고 gzip 형식으로 압축됩니다. Lambda가 수신하는 실제 페이로드는 `{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }` 형식을 따릅니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

   디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

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

위의 데이터 구조에서 키 요소는 다음과 같습니다.

owner

원본 로그 데이터의 AWS 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

messageType

데이터 메시지는 "DATA_MESSAGE" 유형을 사용합니다. 때때로 CloudWatch Logs는 주로 대상에 도달할 수 있는지 확인하기 위해 “CONTROL_MESSAGE” 유형의 Lambda 레코드를 내보낼 수 있습니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. "ID" 속성은 모든 로그 이벤트의 고유 식별자입니다.

예제 3: Amazon Data Firehose를 사용한 구독 필터

이 예시에서는 정의된 필터와 일치하는 수신 로그 이벤트를 Amazon Data Firehose 전송 스트림으로 보내는 CloudWatch Logs 구독을 생성합니다. CloudWatch Logs에서 Amazon Data Firehose로 전송된 데이터는 이미 gzip 레벨 6 압축으로 압축되었으므로 Firehose 전송 스트림 내에서 압축을 사용할 필요가 없습니다. 그런 다음 Firehose의 압축 해제 기능을 사용하여 로그의 압축을 자동으로 해제할 수 있습니다. 자세한 내용은 로그를 사용하여 Kinesis Data CloudWatch Firehose에 쓰기를 참조하십시오.

> **Note**
>
> Firehose 스트림을 만들기 전에 생성될 로그 데이터의 양을 계산하세요. 이 볼륨을 처리할 수 있는 Firehose 스트림을 만들어야 합니다. 스트림이 볼륨을 처리할 수 없는 경우에는 로그 스트림에서 병목 현상이 발생합니다. Firehose 스트림 볼륨 제한에 대한 자세한 내용은 Amazon 데이터 Firehose 데이터 제한을 참조하십시오.

Firehose용 구독 필터를 만들려면

1. Amazon Simple Storage Service(S3) 버킷을 생성합니다. CloudWatch 로그용으로 특별히 만든 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

   다음 명령을 실행하여 자리 표시자 리전을 사용하고자 하는 리전으로 바꿉니다.

   ```
   aws s3api create-bucket --bucket my-bucket --create-bucket-configuration
    LocationConstraint=region
   ```

   다음은 예 출력입니다.

   ```
   {
       "Location": "/my-bucket"
   }
   ```

2. Amazon Data Firehose에 Amazon S3 버킷에 데이터를 넣을 수 있는 권한을 부여하는 IAM 역할을 생성합니다.

자세한 내용은 Amazon Data Firehose 개발자 안내서의 Amazon Data Firehose를 사용한 액세스 제어를 참조하십시오.

먼저 텍스트 편집기를 사용하여 신뢰 정책을 다음과 같은 ~/TrustPolicyForFirehose.json 파일로 생성합니다.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "firehose.amazonaws.com" },
    "Action": "sts:AssumeRole"
    }
}
```

3. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요할 수 있기 때문에 반환된 Role.Arn 값을 적어둡니다.

```
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. 권한 정책을 만들어 Firehose가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 텍스트 편집기를 사용하여 권한 정책을 ~/PermissionsForFirehose.json 파일로 생성합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "s3:AbortMultipartUpload",
          "s3:GetBucketLocation",
          "s3:GetObject",
          "s3:ListBucket",
          "s3:ListBucketMultipartUploads",
          "s3:PutObject" ],
      "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-bucket/*" ]
    }
  ]
}
```

5. 다음 put-role-policy 명령어를 사용하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-
Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. 다음과 같이 대상 Firehose 전송 스트림을 생성하여 RoLearn 및 BucketARN의 플레이스홀더 값을 생성한 역할 및 버킷 ARN으로 대체합니다.

```
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:::my-bucket"}'
```

참고로 Firehose는 전송된 Amazon S3 객체에 대해 YYYY/MM/DD/HH UTC 시간 형식의 접두사를 자동으로 사용합니다. 시간 형식 접두사 앞에 추가할 또 다른 접두사를 지정할 수 있습니다. 슬래시(/)로 끝난 접두사는 Amazon S3 버킷에서 자리 표시자로 표시됩니다.

7. 스트림이 활성 상태가 될 때까지 기다립니다(몇 분 소요). Firehose describe-delivery-stream명령을 사용하여 확인할 수 있습니다. DeliveryStreamDescription DeliveryStreamStatus속성. 또한 다음을 참고하십시오 DeliveryStreamDescription. DeliveryStreamARN 값 (이후 단계에서 필요함):

```
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:::my-bucket",
                    "BufferingHints": {
                        "IntervalInSeconds": 300,
                        "SizeInMBs": 5
                    }
                }
            }
        ]
    }
}
```

8. Firehose 전송 스트림에 데이터를 넣을 수 있는 CloudWatch Logs 권한을 부여하는 IAM 역할을 만드세요. 먼저 텍스트 편집기를 사용하여 신뢰 정책을 ~/TrustPolicyForCWL.json 파일로 생성합니다.

   이 정책에는 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키가 포함되어 있습니다. 자세한 내용은 혼동된 대리자 방지 섹션을 참조하세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
```

```
    "Condition": {
         "StringLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
     }
  }
}
```

9. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요할 수 있기 때문에 반환된 Role.Arn 값을 적어둡니다.

```
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. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 텍스트 편집기를 사용하여 권한 정책을 파일로 생성합니다(예: ~/PermissionsForCWL.json).

```
{
```

```
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
      }
    ]
}
```

11. 다음 put-role-policy 명령을 사용하여 권한 정책을 역할과 연결합니다.

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. Amazon Data Firehose 전송 스트림이 활성 상태가 되고 IAM 역할을 생성한 후 CloudWatch 로그 구독 필터를 생성할 수 있습니다. 구독 필터는 선택한 로그 그룹에서 Amazon Data Firehose 전송 스트림으로의 실시간 로그 데이터 흐름을 즉시 시작합니다.

```
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. 구독 필터를 설정하면 CloudWatch Logs는 필터 패턴과 일치하는 모든 수신 로그 이벤트를 Amazon Data Firehose 전송 스트림으로 전달합니다. Amazon Data Firehose 전송 스트림에 설정된 시간 버퍼 간격을 기준으로 데이터가 Amazon S3에 표시되기 시작합니다. 충분한 시간이 지나고 나면 Amazon S3 버킷을 확인하여 데이터를 확인할 수 있습니다.

```
aws s3api list-objects --bucket 'my-bucket' --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 'my-bucket' --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": {}
}
```

Amazon S3 객체의 데이터는 gzip 형식으로 압축됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
zcat testfile.gz
```

계정 수준 구독 필터

> **Important**
>
> 구독 필터를 사용하면 무한 반복 루프가 발생하여 이를 해결하지 않으면 수집 요금이 크게 증가할 위험이 있습니다. 이러한 위험을 줄이려면 계정 수준 구독 필터의 선택 기준을 사용하여 구독 전송 워크플로의 일부인 리소스에서 로그 데이터를 수집하는 로그 그룹을 제외하는 것이 좋습니다. 이 문제에 대한 자세한 내용과 제외할 로그 그룹 결정에 대한 자세한 내용은 을 참조하십시오. 로그 재귀 방지

계정의 로그 그룹 하위 집합을 포함하는 계정 수준 구독 정책을 설정할 수 있습니다. 계정 구독 정책은 Kinesis Data Streams, Lambda 또는 Firehose와 함께 사용할 수 있습니다. 계정 수준 구독 정책을 통해 수신 서비스로 전송되는 로그는 base64로 인코딩되고 gzip 형식으로 압축됩니다.

> **Note**
>
> 계정의 모든 구독 필터 정책 목록을 보려면 `describe-account-policies` 명령어를 매개변수 값과 함께 사용하십시오. `SUBSCRIPTION_FILTER_POLICY --policy-type` 자세한 내용은 describe-account-policies¶ 을 참조하십시오.

예

- 예제 1: Kinesis Data Streams에 대한 구독 필터
- 예 2: 다음을 포함하는 구독 필터 AWS Lambda
- 예제 3: Amazon Data Firehose를 사용한 구독 필터

예제 1: Kinesis Data Streams에 대한 구독 필터

계정 수준 구독 정책과 함께 사용할 Kinesis Data Streams 데이터 스트림을 생성하기 전에 생성될 로그 데이터의 양을 계산하십시오. 이 볼륨을 처리하기에 충분한 샤드로 스트림을 생성해야 합니다. 스트림에 샤드가 충분하지 않으면 스트림에 병목 현상이 발생합니다. 스트림 볼륨 제한에 대한 자세한 내용은 Kinesis Data Streams 설명서의 할당량 및 제한을 참조하십시오.

> **Warning**
>
> 여러 로그 그룹의 로그 이벤트가 대상으로 전달되기 때문에 병목 현상이 발생할 위험이 있습니다. 제한된 결과물은 최대 24시간 동안 재시도됩니다. 24시간이 지나면 실패한 결과물이 삭제됩니다.
> 제한의 위험을 완화하려면 다음 단계를 수행하세요.
>
> - 지표로 Kinesis Data Streams를 CloudWatch 모니터링할 수 있습니다. 이를 통해 스로틀링을 식별하고 그에 따라 구성을 조정할 수 있습니다. 예를 들어, `DeliveryThrottling` 지표는 구독 대상으로 데이터를 전달할 때 로그가 병목 현상이 발생한 CloudWatch 로그 이벤트 수를 추적합니다. 자세한 정보는 CloudWatch 메트릭을 사용한 모니터링을 참조하세요.
> - Kinesis Data Streams의 스트림에 대해 온디맨드 용량 모드를 사용합니다. 온디맨드 모드는 워크로드가 확장 또는 축소될 때 즉시 워크로드를 수용합니다. 자세한 내용은 온디맨드 모드를 참조하십시오.
> - Kinesis Data Streams의 스트림 용량과 일치하도록 CloudWatch 로그 구독 필터 패턴을 제한하십시오. 스트림에 너무 많은 데이터를 전송하는 경우 필터 크기를 줄이거나 필터 기준을 조정해야 할 수 있습니다.

다음 예에서는 계정 수준 구독 정책을 사용하여 Kinesis Data Streams의 스트림에 모든 로그 이벤트를 전달합니다. 필터 패턴은 모든 로그 이벤트를 텍스트와 `Test` 일치시키고 Kinesis Data Streams의 스트림으로 전달합니다.

Kinesis Data Streams에 대한 계정 수준 구독 정책을 생성하려면

1. 다음 명령을 사용하여 대상 스트림을 생성합니다.

```
$ C:\>  aws kinesis create-stream –stream-name "TestStream" –shard-count 1
```

2. 스트림이 활성화될 때까지 몇 분 정도 기다리십시오. describe-stream 명령을 사용하여 확인하면 스트림이 활성 상태인지 확인할 수 있습니다. StreamDescription StreamStatus속성.

```
aws kinesis describe-stream --stream-name "TestStream"
```

   다음은 예 출력입니다.

```
{
    "StreamDescription": {
```

```
        "StreamStatus": "ACTIVE",
        "StreamName": "TestStream",
        "StreamARN": "arn:aws:kinesis:region:123456789012:stream/TestStream",
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "EndingHashKey": "EXAMPLE84634633746074317682114555",
                    "StartingHashKey": "0"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber":
                    "EXAMPLE6888184566795038319814587845913527021815726l0"
                }
            }
        ]
    }
}
```

3. 스트림에 데이터를 넣을 수 있는 권한을 CloudWatch Logs (로그) 에 부여하는 IAM 역할을 생성하세요. 먼저 신뢰 정책을 파일로 생성해야 합니다(예: ~/TrustPolicyForCWL-Kinesis.json). 텍스트 편집기를 사용하여 이 정책을 생성하세요.

   이 정책에는 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키가 포함되어 있습니다. 자세한 내용은 혼동된 대리자 방지 섹션을 참조하세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringLike": { "aws:SourceArn": "arn:aws:logs:region:123456789012:*" }
     }
   }
}
```

4. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요할 수 있기 때문에 반환된 Role.Arn 값을 적어둡니다.

```
aws iam create-role --role-name CWLtoKinesisRole --assume-role-policy-document
 file://~/TrustPolicyForCWL-Kinesis.json
```

다음은 출력의 예제입니다.

```
{
    "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. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 권한 정책을 파일로 생성합니다(예: ~/PermissionsForCWL-Kinesis.json). 텍스트 편집기를 사용하여 이 정책을 생성하세요. IAM 콘솔을 사용하여 만들지 마세요.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:123456789012:stream/TestStream"
    }
  ]
}
```

6. 다음 put-role-policy명령을 사용하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy  --role-name CWLtoKinesisRole  --policy-name Permissions-
Policy-For-CWL  --policy-document file://~/PermissionsForCWL-Kinesis.json
```

7. 스트림이 Active 상태가 되고 IAM 역할을 생성한 후에는 CloudWatch 로그 구독 필터 정책을 생성할 수 있습니다. 정책은 스트림으로의 실시간 로그 데이터 흐름을 즉시 시작합니다. 이 예시에서는 이름이 LogGroupToExclude1 및 LogGroupToExclude2 인 로그 그룹의 이벤트를 제외하고 해당 문자열을 포함하는 모든 로그 이벤트가 ERROR 스트리밍됩니다.

```
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. 구독 필터를 설정하면 CloudWatch Logs는 필터 패턴 및 선택 기준과 일치하는 모든 수신 로그 이벤트를 스트림에 전달합니다.

   이 selection-criteria 필드는 선택 사항이지만 구독 필터에서 무한 로그 재귀를 일으킬 수 있는 로그 그룹을 제외하는 데 중요합니다. 이 문제와 제외할 로그 그룹 결정에 대한 자세한 내용은 을 참조하십시오로그 재귀 방지. 현재 지원되는 연산자는 NOT IN뿐입니다selection-criteria.

   Kinesis Data Streams 샤드 이터레이터를 사용하고 Kinesis Data Streams 명령을 사용하여 일부 Kinesis get-records 데이터 스트림 레코드를 가져오면 로그 이벤트의 흐름을 확인할 수 있습니다.

```
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"
```

Kinesis Data Streams가 데이터를 반환하기 시작하기 전에 이 명령을 몇 번 사용해야 할 수도 있습니다.

레코드 어레이에서 응답을 확인할 수 있습니다. Kinesis Data Streams 레코드의 데이터 속성은 base64로 인코딩되고 gzip 형식으로 압축됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<Content of Data>" | base64 -d | zcat
```

디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

```
{
    "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"
}
```

데이터 구조의 주요 요소는 다음과 같습니다.

messageType

데이터 메시지는 "DATA_MESSAGE" 유형을 사용합니다. 가끔 CloudWatch 로그에서 “CONTROL_MESSAGE” 유형의 Kinesis Data Streams 레코드를 내보낼 수 있는데, 이는 주로 대상에 도달할 수 있는지 확인하기 위한 것입니다.

owner

원본 로그 데이터의 AWS 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. "ID" 속성은 모든 로그 이벤트의 고유 식별자입니다.

정책 수준

정책이 시행된 수준. “ACCOUNT_LEVEL_POLICY”는 계정 수준의 구독 필터 정책을 `policyLevel` 위한 것입니다.

예 2: 다음을 포함하는 구독 필터 AWS Lambda

이 예시에서는 함수에 CloudWatch 로그 데이터를 보내는 Logs 계정 수준 구독 필터 정책을 생성합니다. AWS Lambda

> **Warning**
>
> Lambda 함수를 생성하기 전에 생성할 로그 데이터의 볼륨을 계산합니다. 이 볼륨을 처리할 수 있는 함수를 생성해야 합니다. 함수가 볼륨을 처리할 수 없는 경우 로그 스트림이 제한됩니다. 모든 로그 그룹 또는 계정 로그 그룹 일부의 로그 이벤트가 대상으로 전달되기 때문에 병목 현상이 발생할 위험이 있습니다. AWS Lambda 제한에 대한 자세한 내용은 Lambda 제한을 참조하세요.

Lambda에 대한 계정 수준 구독 필터 정책을 생성하려면

1. 함수를 생성하십시오. AWS Lambda

   Lambda 실행 역할이 설정되었는지 확인합니다. 자세한 내용은 AWS Lambda 개발자 안내서의 2.2단계: IAM 역할 생성(실행 역할)을 참조하세요.
2. 다음 콘텐츠로 텍스트 편집기를 열고 `helloWorld.js`라는 파일을 생성합니다.

```
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. helloWorld.js 파일을 압축하고 `helloWorld.zip`라는 이름으로 저장합니다.
4. 역할이 첫 단계에서 설정한 Lambda 실행 역할인 경우에는 다음 명령을 사용하세요.

```
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. CloudWatch Logs에 함수를 실행할 권한을 부여하십시오. 다음 명령어를 사용하여 플레이스홀더 계정을 자신의 계정으로 대체하십시오.

```
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. 다음 명령어를 사용하여 계정 수준의 구독 필터 정책을 생성하고 자리 표시자 계정을 자체 계정으로 대체합니다. 이 예시에서는 이름이 및 인 로그 그룹의 로그 이벤트를 제외하고 해당 문자열이 포함된 모든 로그 이벤트가 ERROR 스트리밍됩니다. LogGroupToExclude1 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"
```

   구독 필터를 설정하면 CloudWatch Logs는 필터 패턴 및 선택 기준과 일치하는 모든 수신 로그 이벤트를 스트림에 전달합니다.

   이 selection-criteria 필드는 선택 사항이지만 구독 필터에서 무한 로그 재귀를 일으킬 수 있는 로그 그룹을 제외하는 데 중요합니다. 이 문제와 제외할 로그 그룹 결정에 대한 자세한 내용은 을 참조하십시오로그 재귀 방지. 현재 지원되는 연산자는 NOT IN뿐입니다selection-criteria.

7. (선택 사항) 샘플 로그 이벤트를 사용하여 테스트합니다. 명령 프롬프트에서 다음 명령을 실행하여 구독된 스트림으로 간단한 로그 메시지를 보냅니다.

Lambda 함수의 출력을 보려면 /aws/lambda/helloworld에 출력이 표시되는 Lambda 함수를 탐색합니다.

```
aws logs put-log-events --log-group-name Example1 --log-stream-name logStream1 --
log-events "[{\"timestamp\":CURRENT TIMESTAMP MILLIS , \"message\": \"Simple Lambda
 Test\"}]"
```

Lambda 어레이에서 응답을 확인할 수 있습니다. Lambda 레코드의 데이터 속성은 base64로 인코딩되고 gzip 형식으로 압축됩니다. Lambda가 수신하는 실제 페이로드는 `{ "awslogs": {"data": "BASE64ENCODED_GZIP_COMPRESSED_DATA"} }` 형식을 따릅니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<BASE64ENCODED_GZIP_COMPRESSED_DATA>" | base64 -d | zcat
```

디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

```
{
    "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**
>
> 계정 수준 구독 필터는 대상 Lambda 함수의 로그 그룹에 적용되지 않습니다. 이는 무한 로그 재귀로 인해 수집 요금이 증가할 수 있는 것을 방지하기 위한 것입니다. 이 문제에 대한 자세한 내용은 을 참조하십시오. 로그 재귀 방지

데이터 구조의 주요 요소는 다음과 같습니다.

messageType

데이터 메시지는 "DATA_MESSAGE" 유형을 사용합니다. 가끔 CloudWatch 로그에서 “CONTROL_MESSAGE” 유형의 Kinesis Data Streams 레코드를 내보낼 수 있는데, 이는 주로 대상에 도달할 수 있는지 확인하기 위한 것입니다.

owner

원본 로그 데이터의 AWS 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. "ID" 속성은 모든 로그 이벤트의 고유 식별자입니다.

정책 수준

정책이 시행된 수준. “ACCOUNT_LEVEL_POLICY”는 계정 수준의 구독 필터 정책을 `policyLevel` 위한 것입니다.

예제 3: Amazon Data Firehose를 사용한 구독 필터

이 예시에서는 정의된 필터와 일치하는 수신 CloudWatch 로그 이벤트를 Amazon Data Firehose 전송 스트림으로 보내는 Logs 계정 수준의 구독 필터 정책을 생성합니다. CloudWatch Logs에서 Amazon Data Firehose로 전송된 데이터는 이미 gzip 레벨 6 압축으로 압축되었으므로 Firehose 전송 스트림 내에서 압축을 사용할 필요가 없습니다. 그런 다음 Firehose의 압축 해제 기능을 사용하여 로그의 압축을 자동으로 해제할 수 있습니다. 자세한 내용은 로그를 사용하여 Kinesis Data CloudWatch Firehose에 쓰기를 참조하십시오.

> **Warning**
>
> Firehose 스트림을 만들기 전에 생성될 로그 데이터의 양을 계산하세요. 이 볼륨을 처리할 수 있는 Firehose 스트림을 만들어야 합니다. 스트림이 볼륨을 처리할 수 없는 경우에는 로그 스트림에서 병목 현상이 발생합니다. Firehose 스트림 볼륨 제한에 대한 자세한 내용은 Amazon 데이터 Firehose 데이터 제한을 참조하십시오.

Firehose용 구독 필터를 만들려면

1. Amazon Simple Storage Service(S3) 버킷을 생성합니다. CloudWatch 로그용으로 특별히 만든 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

   다음 명령을 실행하여 자리 표시자 리전을 사용하고자 하는 리전으로 바꿉니다.

   ```
   aws s3api create-bucket --bucket my-bucket --create-bucket-configuration
    LocationConstraint=region
   ```

   다음은 예 출력입니다.

   ```
   {
       "Location": "/my-bucket"
   }
   ```

2. Amazon Data Firehose에 Amazon S3 버킷에 데이터를 넣을 수 있는 권한을 부여하는 IAM 역할을 생성합니다.

   자세한 내용은 Amazon Data Firehose 개발자 안내서의 Amazon Data Firehose를 사용한 액세스 제어를 참조하십시오.

   먼저 텍스트 편집기를 사용하여 신뢰 정책을 다음과 같은 ~/TrustPolicyForFirehose.json 파일로 생성합니다.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "firehose.amazonaws.com" },
    "Action": "sts:AssumeRole"
    }
}
```

3. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요하므로 반환된 Role.Arn 값을 기록해 두십시오.

```
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. 권한 정책을 만들어 Firehose가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 텍스트 편집기를 사용하여 권한 정책을 ~/PermissionsForFirehose.json 파일로 생성합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "s3:AbortMultipartUpload",
          "s3:GetBucketLocation",
          "s3:GetObject",
          "s3:ListBucket",
          "s3:ListBucketMultipartUploads",
          "s3:PutObject" ],
      "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-bucket/*" ]
    }
  ]
}
```

5. 다음 put-role-policy 명령어를 사용하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name Permissions-
Policy-For-Firehose --policy-document file://~/PermissionsForFirehose.json
```

6. 다음과 같이 대상 Firehose 전송 스트림을 생성하여 RoLearn 및 BucketARN의 플레이스홀더 값을 생성한 역할 및 버킷 ARN으로 대체합니다.

```
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:::my-bucket"}'
```

Firehose는 전송된 Amazon S3 객체에 대해 YYYY/MM/DD/HH UTC 시간 형식의 접두사를 자동으로 사용합니다. 시간 형식 접두사 앞에 추가할 또 다른 접두사를 지정할 수 있습니다. 슬래시(/)로 끝난 접두사는 Amazon S3 버킷에서 자리 표시자로 표시됩니다.

7. 스트림이 활성화될 때까지 몇 분 정도 기다리십시오. Firehose describe-delivery-stream명령을 사용하여 확인할 수 있습니다. DeliveryStreamDescription DeliveryStreamStatus속성. 또한 다음을 참고하십시오 DeliveryStreamDescription. DeliveryStreamARN 값 (이후 단계에서 필요함):

```
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:::my-bucket",
                    "BufferingHints": {
                        "IntervalInSeconds": 300,
                        "SizeInMBs": 5
                    }
                }
            }
        ]
    }
}
```

8. Firehose 전송 스트림에 데이터를 넣을 수 있는 CloudWatch Logs 권한을 부여하는 IAM 역할을 만드세요. 먼저 텍스트 편집기를 사용하여 신뢰 정책을 ~/TrustPolicyForCWL.json 파일로 생성합니다.

   이 정책에는 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키가 포함되어 있습니다. 자세한 내용은 혼동된 대리자 방지 섹션을 참조하세요.

```
{
```

```
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole",
    "Condition": {
         "StringLike": {
             "aws:SourceArn": "arn:aws:logs:region:123456789012:*"
         }
     }
  }
}
```

9. create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 이후 단계에서 필요하므로 반환된 Role.Arn 값을 기록해 둡니다.

```
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. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하십시오. 먼저 텍스트 편집기를 사용하여 권한 정책을 파일로 생성합니다(예: ~/PermissionsForCWL.json).

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:PutRecord"],
        "Resource":[
            "arn:aws:firehose:region:account-id:deliverystream/delivery-stream-
name"]
      }
    ]
}
```

11. 다음 put-role-policy 명령을 사용하여 권한 정책을 역할과 연결합니다.

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-
name Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

12. Amazon Data Firehose 전송 스트림이 활성 상태가 되고 IAM 역할을 생성한 후에는 CloudWatch Logs 계정 수준 구독 필터 정책을 생성할 수 있습니다. 이 정책은 선택한 로그 그룹에서 Amazon Data Firehose 전송 스트림으로의 실시간 로그 데이터 흐름을 즉시 시작합니다.

```
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. 구독 필터를 설정하면 CloudWatch Logs는 필터 패턴과 일치하는 수신 로그 이벤트를 Amazon Data Firehose 전송 스트림으로 전달합니다.

    이 selection-criteria 필드는 선택 사항이지만 구독 필터에서 무한 로그 재귀를 일으킬 수 있는 로그 그룹을 제외하는 데 중요합니다. 이 문제와 제외할 로그 그룹 결정에 대한 자세한 내용은 을 참조하십시오로그 재귀 방지. 현재 지원되는 연산자는 NOT IN뿐입니다selection-criteria.

Amazon Data Firehose 전송 스트림에 설정된 시간 버퍼 간격을 기준으로 데이터가 Amazon S3에 표시되기 시작합니다. 충분한 시간이 지나고 나면 Amazon S3 버킷을 확인하여 데이터를 확인할 수 있습니다.

```
aws s3api list-objects --bucket 'my-bucket' --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 'my-bucket' --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": {}
}
```

Amazon S3 객체의 데이터는 gzip 형식으로 압축됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
zcat testfile.gz
```

계정 간 지역 간 구독

다른 AWS 계정의 소유자와 협업하여 Amazon Kinesis 또는 Amazon Data Firehose 스트림 (계정 간 데이터 공유라고 함) 과 같은 AWS 리소스에서 소유자의 로그 이벤트를 수신할 수 있습니다. 예를 들어 중앙 집중식 Kinesis Data Streams 또는 Firehose 스트림에서 이 로그 이벤트 데이터를 읽어 사용자 지정 처리 및 분석을 수행할 수 있습니다. 사용자 지정 처리는 특히 다수의 계정에서 데이터를 공동 사용 및 분석할 때 유용합니다.

예를 들어 기업의 정보 보안 그룹은 실시간 침입 탐지나 이상 행동에 대한 데이터를 분석하고자 중앙 처리를 위해 연동된 프로덕션 로그를 수집하여 모든 부서의 계정에 대해 감사를 실시할 수 있습니다. 이러한 계정 간 이벤트 데이터의 실시간 스트림을 수집해서 정보 보안 그룹에 제공하면, 보안 그룹은 Kinesis Data Streams를 사용해 기존의 보안 분석 시스템에 데이터를 연결할 수 있습니다.

> **Note**
>
> 로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상이 가리키는 AWS 리소스가 다른 지역에 있을 수 있습니다. 다음 섹션의 예에서는 모든 지역별 리소스가 미국 동부 (버지니아 북부) 에서 생성됩니다.

주제

- Kinesis Data Streams를 사용한 계정 간 지역 간 로그 데이터 공유
- Firehose를 사용한 계정 간 지역 간 로그 데이터 공유
- Kinesis Data Streams를 사용한 계정 간 리전 계정 수준 구독
- Firehose를 사용한 교차 계정 지역 계정 수준 구독

Kinesis Data Streams를 사용한 계정 간 지역 간 로그 데이터 공유

교차 계정 구독을 생성할 때 단일 계정 또는 조직을 발신자로 지정할 수 있습니다. 조직을 지정하는 경우 이 절차를 통해 조직의 모든 계정에서 수신자 계정으로 로그를 보낼 수 있습니다.

계정에서 로그 데이터를 공유하려면 로그 데이터 발신자 및 수신자를 설정해야 합니다.

- 로그 데이터 발신자 - 수신자로부터 대상 정보를 가져와 지정된 대상으로 CloudWatch 로그 이벤트를 전송할 준비가 되었음을 Logs에 알립니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 발신자가 가상 AWS 계정 번호 1111111111로 표시됩니다.

  한 조직 내의 여러 계정에서 한 수신자 계정으로 로그를 보내려면 해당 조직의 모든 계정에서 수신자 계정으로 로그를 보낼 수 있는 권한을 부여하는 정책을 만들 수 있습니다. 각 발신자 계정에 대해 별도의 구독 필터를 설정해야 합니다.
- 로그 데이터 수신자 - Kinesis Data Streams 스트림을 캡슐화하는 대상을 설정하고 수신자가 로그 데이터를 받기를 원한다는 것을 CloudWatch Logs에 알립니다. 그런 다음 수신자는 이 대상에 대한 정보를 발신자와 공유합니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 수신자가 가상 AWS 계정 번호 9999999999로 표시됩니다.

계정 간 사용자로부터 로그 이벤트를 받기 시작하려면 로그 데이터 수신자가 먼저 로그 대상을 생성합니다. CloudWatch 각 대상은 다음과 같은 키 요소로 이루어져 있습니다.

대상 이름

생성하고자 하는 대상의 이름입니다.

대상 ARN

구독 피드의 대상으로 사용하려는 AWS 리소스의 Amazon 리소스 이름 (ARN)

역할 ARN

선택한 스트림에 데이터를 넣는 데 필요한 권한을 CloudWatch 로그에 부여하는 AWS Identity and Access Management (IAM) 역할.

액세스 정책

대상에 쓰기 권한이 허용된 사용자들에게 적용되는 IAM 정책 문서(JSON 형식, IAM 정책 문법을 사용해 작성)입니다.

> **Note**
>
> 로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상이 가리키는 AWS 리소스는 다른 리전에 위치할 수 있습니다. 다음 섹션의 예제에서는 모든 리전 관련 리소스가 미국 동부(버지니아 북주)에서 생성됩니다.

주제

새 교차 계정 구독 설정

이러한 섹션의 단계에 따라 새 교차 계정 로그 구독을 설정합니다.

주제

1단계: 대상 생성

> **Important**
>
> 이 절차의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.

이 예에서 로그 데이터 수신자 계정의 계정 ID는 999999999999인 반면, 로그 데이터 발신자 AWS 계정 ID는 111111111111입니다. AWS

이 예제에서는 RecipientStream 라는 Kinesis Data Streams 스트림을 사용하여 대상을 생성하고 CloudWatch Logs가 여기에 데이터를 쓸 수 있도록 하는 역할을 생성합니다.

대상이 생성되면 CloudWatch Logs는 수신자 계정을 대신하여 대상에 테스트 메시지를 보냅니다. 나중에 구독 필터가 활성화되면 CloudWatch Logs는 원본 계정을 대신하여 대상에 로그 이벤트를 보냅니다.

대상을 생성하려면

1. 수신자 계정에서 Kinesis Data Streams에 대상 스트림을 생성합니다. 명령 프롬프트에서 다음과 같이 입력합니다.

```
aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
```

2. 스트림이 활성 상태가 될 때까지 기다립니다. aws kinesis describe-stream 명령을 사용하여 확인할 수 있습니다. StreamDescription StreamStatus속성. 또한 StreamDescription.StreamArn 값은 나중에 로그에 전달되므로 기록해 CloudWatch 두십시오.

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

스트림이 활성 상태가 될 때까지 1~2분 정도 기다려야 할 수 있습니다.

3. CloudWatch Logs에 스트림에 데이터를 넣을 수 있는 권한을 부여하는 IAM 역할을 생성하세요. 먼저 ~/ TrustPolicyFor cwl.json 파일에 신뢰 정책을 생성해야 합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고 IAM 콘솔은 사용하지 마세요.

이 정책은 sourceAccountId를 지정하여 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키를 포함합니다. 첫 번째 호출에서 소스 계정 ID를 아직 모르는 경우 소스 ARN 필드에 대상 ARN을 넣는 것이 좋습니다. 후속 호출에서는 소스 ARN을 첫 번째 호출에서 수집한 실제 소스 ARN으로 설정해야 합니다. 자세한 정보는 혼동된 대리자 방지을 참조하세요.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
}
```

4. aws iam create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 반환된 Role.Arn 값은 나중에 Logs에도 전달되므로 기록해 두십시오. CloudWatch

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
```

```
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                },
                "Principal": {
                    "Service": "logs.amazonaws.com"
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2015-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 텍스트 편집기를 사용하여 ~/ PermissionsFor cwl.json 파일에 권한 정책을 생성합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. aws iam 명령을 사용하여 권한 정책을 역할에 연결합니다. put-role-policy

```
aws iam put-role-policy \
    --role-name CWLtoKinesisRole \
    --policy-name Permissions-Policy-For-CWL \
    --policy-document file://~/PermissionsForCWL.json
```

7. 스트림이 활성 상태가 되고 IAM 역할을 생성한 후 CloudWatch 로그 대상을 생성할 수 있습니다.

   a. 이 단계를 수행해도 액세스 정책이 대상에 연결되는 것은 아니며, 대상 생성을 완료하기 위한 두 단계 중 첫 번째 단계를 완료한 것일 뿐입니다. 페이로드에 DestinationArn반환되는 내용을 기록해 두십시오.

```
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. 7a 단계를 완료한 후 로그 데이터 수신자 계정에서 액세스 정책을 대상과 연결합니다. 이 정책은 로그: PutSubscriptionFilter 작업을 지정하고 발신자 계정에 대상에 액세스할 수 있는 권한을 부여해야 합니다.

정책은 로그를 보내는 AWS 계정에 권한을 부여합니다. 정책에서 이 계정 하나만 지정할 수 있으며, 또는 발신자 계정이 조직의 구성원인 경우 정책은 해당 조직의 조직 ID를 지정할 수 있습니다. 이렇게 하면 정책 하나만 생성하여 한 조직의 여러 계정이 이 대상 계정으로 로그를 보내도록 할 수 있습니다.

텍스트 편집기를 사용하여 이름이 ~/AccessPolicy.json이고 다음 정책 문 중 하나를 포함한 파일을 생성합니다.

이 첫 번째 예제 정책은 ID가 o-1234567890인 조직의 모든 계정이 수신자 계정으로 로그를 보내도록 허용합니다.

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Sid" : "",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : "logs:PutSubscriptionFilter",
            "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination",
            "Condition": {
               "StringEquals" : {
                   "aws:PrincipalOrgID" : ["o-1234567890"]
```

```
                }
            }
        }
    ]
}
```

다음 예에서는 로그 데이터 발신자 계정(111111111111)에서만 로그 데이터 수신자 계정으로 로그를 보내도록 허용합니다.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination"
    }
  ]
}
```

c. 이전 단계에서 생성한 정책을 대상에 연결합니다.

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

*# ### ### #### ID# 1111111111# AWS ### #### ARN arn:aws:logs: # #:999999999999:###:####### ### # **PutSubscriptionFilter**# ####.* 다른 사용자가 이 목적지를 상대로 전화를 걸려고 시도하면 거부됩니다. PutSubscriptionFilter

액세스 정책에 대한 사용자 권한의 유효성을 검사하려면 IAM 사용 설명서의 정책 검사기 사용을 참조하세요.

작업을 마쳤으면 교차 계정 권한을 사용하기 AWS Organizations 위해 사용하는 경우 다음 단계를 따르세요. 2단계: (조직을 사용하는 경우에만) IAM 역할 생성 Organizations를 사용하는 대신 다른 계정에 직접 권한을 부여하는 경우 해당 단계를 건너뛰고 4단계: 구독 필터 생성 단계로 넘어갑니다.

2단계: (조직을 사용하는 경우에만) IAM 역할 생성

이전 섹션에서 111111111111 계정이 속한 조직에 권한을 부여하는 액세스 정책을 사용하여 대상을 생성한 경우, 111111111111 계정에 직접 권한을 부여하는 대신 이 섹션의 단계를 따릅니다. 그렇지 않다면 4단계: 구독 필터 생성 단계로 건너뛰어도 됩니다.

이 섹션의 단계는 IAM 역할을 생성합니다. IAM 역할을 생성하면 발신자 계정에 수신자를 대상으로 구독 필터를 생성할 권한이 있는지 여부를 가정하고 검증할 CloudWatch 수 있습니다.

발신자 계정에서 이 섹션의 단계를 수행합니다. 역할은 발신자 계정에 있어야 하며 구독 필터에서 이 역할의 ARN을 지정합니다. 이 예제에서 발신자 계정은 111111111111입니다.

AWS Organizations를 사용하여 교차 계정 로그 구독에 필요한 IAM 역할 생성

1. /TrustPolicyForCWLSubscriptionFilter.json 파일에 다음 트러스트 정책을 생성합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고, IAM 콘솔은 사용하지 마세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 이 정책을 사용하는 IAM 역할을 생성합니다. 명령에 의해 반환되는 Arn 값은 이 절차의 뒷부분에 필요하므로 메모해 둡니다. 이 예제에서 생성하는 역할의 이름으로는 CWLtoSubscriptionFilterRole을 사용합니다.

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. 권한 정책을 생성하여 CloudWatch Logs가 계정에서 수행할 수 있는 작업을 정의하십시오.

   a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 ~/PermissionsForCWLSubscriptionFilter.json인 파일로 생성합니다.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

b. 다음 명령을 입력하여 방금 생성한 권한 정책을 2단계에서 생성한 역할에 연결합니다.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

작업을 마쳤으면 4단계: 구독 필터 생성 단계로 넘어갑니다.

3단계: 교차 계정 대상에 대한 IAM 권한 추가/검증

AWS 교차 계정 정책 평가 로직에 따르면 계정 간 리소스 (예: 구독 필터의 대상으로 사용되는 Kinesis 또는 Firehose 스트림) 에 액세스하려면 전송 계정에 계정 간 대상 리소스에 대한 명시적 액세스를 제공하는 ID 기반 정책이 있어야 합니다. 정책 평가 로직에 대한 자세한 내용은 교차 계정 정책 평가 로직을 참조하세요.

구독 필터를 생성하는 데 사용하는 IAM 역할 또는 IAM 사용자에게 자격 증명 기반 정책을 연결할 수 있습니다. 전송 계정에 이 정책이 있어야 합니다. 관리자 역할을 사용하여 구독 필터를 만드는 경우 이 단계를 건너뛰고 4단계: 구독 필터 생성(으)로 넘어갈 수 있습니다.

교차 계정에 필요한 IAM 권한을 추가 또는 검증하려면

1. 다음 명령을 입력하여 AWS 로그 명령을 실행하는 데 사용되는 IAM 역할 또는 IAM 사용자를 확인합니다.

   ```
   aws sts get-caller-identity
   ```

   이 명령은 다음과 비슷한 출력을 반환합니다.

```
{
"UserId": "User ID",
"Account": "sending account id",
"Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
}
```

또는 로 표시된 값을 기록해 두십시오. *RoleNameUserName*

2. 전송 계정에 로그인하고 AWS Management Console 1단계에서 입력한 명령의 출력에 반환된 IAM 역할 또는 IAM 사용자로 연결된 정책을 검색합니다.
3. 이 역할 또는 사용자에게 연결된 정책이 교차 계정 대상 리소스에서 `logs:PutSubscriptionFilter`를 호출할 수 있는 명시적 권한을 제공하는지 확인하세요. 다음 정책 예는 권장 권한을 보여 줍니다.

   다음 정책은 단일 AWS 계정 계정으로만 모든 대상 리소스에 구독 필터를 생성할 수 있는 권한을 제공합니다. 123456789012

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow subscription filters on any resource in one specific
 account",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:*"
            ]
        }
    ]
}
```

   다음 정책은 단일 AWS 계정, 계정으로 이름이 지정된 특정 대상 리소스에만 구독 필터를 만들 수 `sampleDestination` 있는 권한을 제공합니다123456789012.

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

```
            "Sid": "Allow subscription filters on one specific resource in one
 specific account",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:sampleDestination"
            ]
        }
    ]
}
```

4단계: 구독 필터 생성

대상을 생성하고 나면 로그 데이터 수신자 계정에서 다른 AWS 계정이 로그 이벤트를 동일한 대상으로 전송할 수 있도록 대상 ARN(arn:aws:logs:us-east-1:999999999999:destination:testDestination)을 이들과 공유할 수 있습니다. 그러면 이러한 다른 전송 계정 사용자는 이 대상에 해당되는 로그 그룹에 대한 구독 필터를 생성합니다. 그 즉시 구독 필터는 실시간으로 선택한 로그 그룹에서 지정된 스트림으로 로그 데이터를 이동시키기 시작합니다.

> **Note**
>
> 전체 조직에 구독 필터에 대한 권한을 부여하는 경우 2단계: (조직을 사용하는 경우에만) IAM 역할 생성에서 생성한 IAM 역할의 ARN을 사용해야 합니다.

다음 예시에서는 전송 계정에 구독 필터를 생성합니다. 이 필터는 AWS CloudTrail 이벤트가 포함된 로그 그룹과 연결되므로 “루트” AWS 자격 증명으로 기록된 모든 활동이 이전에 만든 대상으로 전달됩니다. 해당 대상은 ""이라는 스트림을 캡슐화합니다. RecipientStream

다음 섹션의 나머지 단계에서는 AWS CloudTrail 사용 설명서의 CloudWatch 로그로 CloudTrail 이벤트 보내기의 지침을 따르고 이벤트가 포함된 로그 그룹을 만들었다고 가정합니다. CloudTrail 이 단계에서는 이 로그 그룹의 이름이 `CloudTrail/logs`라고 가정합니다.

다음 명령을 입력할 때는3단계: 교차 계정 대상에 대한 IAM 권한 추가/검증에서 IAM 사용자로 로그인하거나 정책을 추가한 IAM 역할을 사용하여 로그인해야 합니다.

```
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"
```

로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상은 다른 지역에 있는 Kinesis Data Streams 스트림과 같은 AWS 리소스를 가리킬 수 있습니다.

로그 이벤트 흐름 검증

구독 필터를 만들면 CloudWatch Logs는 필터 패턴과 일치하는 모든 수신 로그 이벤트를 대상 스트림 내에 캡슐화된 ""의 스트림으로 전달합니다. RecipientStream 대상 소유자는 aws kinesis 명령을 사용하여 Kinesis Data Streams 샤드를 가져오고 aws kinesis get-records get-shard-iterator 명령을 사용하여 일부 Kinesis 데이터 스트림 레코드를 가져와서 이러한 문제가 발생하는지 확인할 수 있습니다.

```
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**
>
> get-records 명령어 반환을 몇 차례 반복해야 Kinesis Data Streams가 데이터 반환을 시작할 수 있습니다.

Kinesis Data Streams 레코드 어레이에서 응답을 확인할 수 있습니다. Kinesis Data Streams 레코드의 데이터 속성은 gzip 형식으로 압축된 다음 base64로 인코딩됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<Content of Data>" | base64 -d | zcat
```

디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

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

이 데이터 구조에서 키 요소는 다음과 같습니다.

owner

원본 AWS 로그 데이터의 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

messageType

데이터 메시지는 'DATA_MESSAGE' 유형을 사용합니다. 가끔 CloudWatch 로그에서 “CONTROL_MESSAGE” 유형의 Kinesis Data Streams 레코드를 내보낼 수 있는데, 이는 주로 대상에 도달할 수 있는지 확인하기 위한 것입니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. ID 속성은 모든 로그 이벤트의 고유 식별자입니다.

런타임 시 대상 멤버십 수정

소유한 대상에서 몇몇 사용자의 멤버십을 추가 또는 제거해야 하는 상황에 직면할 수 있습니다. 새 액세스 정책과 함께 대상에 `put-destination-policy` 명령을 사용할 수 있습니다. 다음 예제에서는 이전에 추가한 계정인 111111111111이 추가적인 로그 데이터 전송을 중단하고 계정 222222222222가 활성화됩니다.

1. 현재 대상 TestDestination 과 연결된 정책을 가져오고 다음 사항을 기록해 둡니다. 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. 계정 1111111111111이 중단되고 계정 2222222222222가 활성화되었음을 반영하도록 이 정책을 업데이트합니다. 이 정책을 ~/ NewAccessPolicy .json 파일에 넣으십시오.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "222222222222"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:region:999999999999:destination:testDestination"
    }
  ]
}
```

3. NewAccessPolicy.json 파일에 정의된 정책을 대상에 PutDestinationPolicy연결하려면 호출하십시오.

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

이렇게 하면 계정 ID 111111111111에서 로그 이벤트가 비활성화됩니다. 계정 ID 222222222222에서 나온 로그 이벤트는 계정 222222222222 의 소유자가 구독 필터를 생성하고 나면 그 즉시 대상으로 이동하기 시작합니다.

기존 교차 계정 구독 업데이트

현재 대상 계정이 특정 발신자 계정에만 권한을 부여하는 교차 계정 로그 구독이 있고 대상 계정이 조직의 모든 계정에 대한 액세스 권한을 부여하도록 이 구독을 업데이트하려는 경우 이 섹션의 단계를 따릅니다.

주제

1단계: 구독 필터 업데이트

> **Note**
>
> 이 단계는 AWS 서비스에서 로깅 활성화에 나열된 서비스에서 생성된 로그에 대한 교차 계정 구독에만 필요합니다. 해당 로그 그룹 중 하나에서 생성된 로그로 작업하지 않는 경우 2단계: 기존 대상 액세스 정책 업데이트 섹션으로 건너뛸 수 있습니다.

경우에 따라 대상 계정으로 로그를 보내는 모든 발신자 계정의 구독 필터를 업데이트해야 합니다. 이 업데이트에는 발신자 계정에 수신자 계정에 로그를 전송할 권한이 있다고 가정하고 CloudWatch 검증할 수 있는 IAM 역할이 추가되었습니다.

교차 계정 구독 권한에 대해 조직 ID를 사용하도록 업데이트하려는 모든 발신자 계정에 대해서는 이 섹션의 단계를 따릅니다.

이 섹션의 예제에서 두 개의 111111111111 계정 및 222222222222 계정은 999999999999 계정에 로그를 전송하기 위해 생성된 구독 필터를 이미 가지고 있습니다. 기존 구독 필터 값은 다음과 같습니다.

```
## 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"
```

현재 구독 필터 파라미터 값을 찾아야 하는 경우 다음 명령을 입력합니다.

```
aws logs describe-subscription-filters
```

```
    \ --log-group-name "my-log-group-name"
```

교차 계정 로그 권한에 대한 조직 ID 사용을 시작하도록 구독 필터 업데이트

1. ~/TrustPolicyForCWL.json 파일에 다음 트러스트 정책을 생성합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고, IAM 콘솔은 사용하지 마세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 이 정책을 사용하는 IAM 역할을 생성합니다. 명령에 의해 반환되는 Arn 값의 Arn 값은 이 절차의 뒷부분에 필요하므로 메모해 둡니다. 이 예제에서 생성하는 역할의 이름으로는 CWLtoSubscriptionFilterRole을 사용합니다.

```
aws iam create-role
    \ --role-name CWLtoSubscriptionFilterRole
    \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

3. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하십시오.

   a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 / PermissionsForCWLSubscriptionFilter.json인 파일로 생성합니다.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 다음 명령을 입력하여 방금 생성한 권한 정책을 2단계에서 생성한 역할에 연결합니다.

```
aws iam put-role-policy
```

```
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. 다음 명령을 입력하여 구독 필터를 업데이트합니다.

```
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"
```

2단계: 기존 대상 액세스 정책 업데이트

모든 발신자 계정에서 구독 필터를 업데이트한 후, 수신자 계정에서 대상 액세스 정책을 업데이트할 수 있습니다.

다음 예제에서 수신자 계정은 999999999999이며 대상의 이름은 testDestination입니다.

업데이트를 통해 ID가 o-1234567890인 조직의 모든 계정이 수신자 계정으로 로그를 보낼 수 있습니다. 구독 필터가 생성된 계정만이 실제로 수신자 계정에 로그를 보냅니다.

수신자 계정의 대상 액세스 정책을 업데이트하여 권한에 대한 조직 ID 사용

1. 수신자 계정에서 텍스트 편집기를 사용하여 다음 내용을 포함한 ~/AccessPolicy.json 파일을 생성합니다.

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Sid" : "",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : "logs:PutSubscriptionFilter",
            "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination",
            "Condition": {
               "StringEquals" : {
                   "aws:PrincipalOrgID" : ["o-1234567890"]
```

```
                }
            }
        }
    ]
}
```

2. 다음 명령을 입력하여 방금 생성한 정책을 기존 대상에 연결합니다. 특정 AWS 계정 ID을 나열하는 액세스 정책 대신 조직 ID로 액세스 정책을 사용하도록 대상을 업데이트하려면 `force` 파라미터를 포함시킵니다.

> **Warning**
>
> 에 나열된 AWS 서비스에서 전송한 로그로 작업하는 경우 이 단계를 수행하기 전에 먼저 에 설명된 대로 모든 발신자 계정의 구독 필터를 업데이트해야 합니다. AWS 서비스에서 로깅 활성화 1단계: 구독 필터 업데이트

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
    \ --force
```

Firehose를 사용한 계정 간 지역 간 로그 데이터 공유

계정에서 로그 데이터를 공유하려면 로그 데이터 발신자 및 수신자를 설정해야 합니다.

- 로그 데이터 발신자 - 수신자로부터 대상 정보를 받고 지정된 대상으로 CloudWatch 로그 이벤트를 전송할 준비가 되었음을 Logs에 알립니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 발신자가 가상 AWS 계정 번호 1111111111로 표시됩니다.
- 로그 데이터 수신자 - Kinesis Data Streams 스트림을 캡슐화하는 대상을 설정하고 수신자가 로그 데이터를 받기를 원한다는 것을 CloudWatch Logs에 알립니다. 그런 다음 수신자는 이 대상에 대한 정보를 발신자와 공유합니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 수신자가 가상 AWS 계정 번호 2222222222로 표시됩니다.

이 섹션의 예시에서는 Amazon S3 스토리지와 함께 Firehose 전송 스트림을 사용합니다. 다양한 설정으로 Firehose 전송 스트림을 설정할 수도 있습니다. 자세한 내용은 Firehose 전송 스트림 생성을 참조하십시오.

> **Note**
>
> 로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상이 가리키는 AWS 리소스는 다른 리전에 위치할 수 있습니다.

> **Note**
>
> 동일한 계정 및 지역 간 전송 스트림에 대한 Firehose 구독 필터가 지원됩니다.

주제

1단계: Firehose 전송 스트림 생성

> **Important**
>
> 다음 단계를 완료하기 전에 액세스 정책을 사용해야 Firehose가 Amazon S3 버킷에 액세스할 수 있습니다. 자세한 내용은 Amazon Data Firehose 개발자 안내서의 액세스 제어를 참조하십시오.
>
> 이 섹션(1단계)의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.
>
> 미국 동부(버지니아 북부)가 다음 샘플 명령에 사용됩니다. 이 리전을 배포에 적합한 리전으로 바꿉니다.

대상으로 사용할 Firehose 전송 스트림을 만들려면

1. Amazon S3 버킷 생성:

```
aws s3api create-bucket --bucket firehose-test-bucket1 --create-bucket-
configuration LocationConstraint=us-east-1
```

2. 데이터를 버킷에 넣을 수 있는 권한을 Firehose에 부여하는 IAM 역할을 생성합니다.

   a. 먼저 텍스트 편집기를 사용하여 신뢰 정책을 ~/TrustPolicyForFirehose.json 파일로 생성합니다.

   ```
   { "Statement": { "Effect": "Allow", "Principal": { "Service":
    "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
    { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
   ```

   b. IAM 역할을 생성하여 방금 생성한 신뢰 정책 파일을 지정합니다.

   ```
   aws iam create-role \
       --role-name FirehosetoS3Role \
       --assume-role-policy-document file://~/TrustPolicyForFirehose.json
   ```

   c. 이 명령의 출력은 다음과 비슷합니다. 역할 이름과 역할 ARN을 기록해 둡니다.

   ```
   {
       "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. 권한 정책을 만들어 Firehose가 계정에서 수행할 수 있는 작업을 정의하세요.

   a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 ~/PermissionsForFirehose.json인 파일로 생성합니다. 사용 사례에 따라 이 파일에 권한을 더 추가해야 할 수도 있습니다.

```
{
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::firehose-test-bucket1",
            "arn:aws:s3:::firehose-test-bucket1/*"
        ]
    }]
}
```

   b. 다음 명령을 입력하여 방금 생성한 권한 정책을 IAM 역할에 연결합니다.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. 다음 명령어를 입력하여 Firehose 전송 스트림을 생성합니다. *my-role-arn*및 를 *my-bucket-arn*배포에 맞는 올바른 값으로 바꾸십시오.

```
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:::firehose-test-bucket1"}'
```

   다음과 같이 출력됩니다

```
{
```

```
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

2단계: 대상 생성

> **Important**
>
> 이 절차의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.

대상이 생성되면 CloudWatch Logs는 수신자 계정을 대신하여 대상에 테스트 메시지를 보냅니다. 나중에 구독 필터가 활성화되면 CloudWatch Logs는 원본 계정을 대신하여 대상에 로그 이벤트를 보냅니다.

대상을 생성하려면

1. 에서 1단계: Firehose 전송 스트림 생성 만든 Firehose 스트림이 활성화될 때까지 기다리세요. 다음 명령어를 사용하여 확인할 수 있습니다. StreamDescription StreamStatus속성.

   ```
   aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
   ```

   또한, 기록해 두십시오 DeliveryStreamDescription. DeliveryStreamARN 값입니다. 이후 단계에서 사용해야 하기 때문입니다. 이 명령의 샘플 출력:

   ```
   {
       "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:::firehose-test-bucket1",
                    "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:::firehose-test-bucket1",
                    "BufferingHints": {
                        "SizeInMBs": 5,
                        "IntervalInSeconds": 300
                    },
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    },
                    "S3BackupMode": "Disabled"
                }
            }
        ],
        "HasMoreDestinations": false
    }
}
```

전송 스트림이 활성 상태가 될 때까지 1~2분 정도 기다려야 할 수 있습니다.

2. 전송 스트림이 활성화되면 Firehose 스트림에 데이터를 넣을 권한을 CloudWatch Logs에 부여하는 IAM 역할을 생성합니다. 먼저 TrustPolicyFor~/ cwl.json 파일에 신뢰 정책을 생성해야 합니다. 텍스트 편집기를 사용하여 이 정책을 생성하세요. CloudWatch 로그 엔드포인트에 대한 자세한 내용은 Amazon CloudWatch Logs 엔드포인트 및 할당량을 참조하십시오.

이 정책은 sourceAccountId를 지정하여 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 aws:SourceArn 글로벌 조건 컨텍스트 키를 포함합니다. 첫 번째 호출에서 소스 계정 ID를 아직 모르는 경우 소스 ARN 필드에 대상 ARN을 넣는 것이 좋습니다. 후속 호출에서는 소스 ARN을 첫 번째 호출에서 수집한 실제 소스 ARN으로 설정해야 합니다. 자세한 정보는 혼동된 대리자 방지을 참조하세요.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.region.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
     }
}
```

3. aws iam create-role 명령을 사용하여 IAM 역할을 생성해 방금 만든 신뢰 정책 파일을 지정합니다.

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

다음은 출력 샘플입니다. 이후 단계에서 사용해야 하므로 반환된 Role.Arn 값을 메모해 둡니다.

```
{
    "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": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하십시오. 먼저 텍스트 편집기를 사용하여 ~/ PermissionsFor cwl.json 파일에 권한 정책을 생성합니다.

```
{
    "Statement":[
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
}
```

5. 다음 명령을 입력하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Firehose 전송 스트림이 활성 상태가 되고 IAM 역할을 생성한 후 Logs 대상을 생성할 수 있습니다. CloudWatch

   a. 이 단계를 수행했다고 액세스 정책이 대상에 연결되는 것은 아니며, 대상 생성을 완료하기 위한 두 단계 중 첫 번째 단계일 뿐입니다. 페이로드에서 반환된 새 대상의 ARN은 이후 단계에서 `destination.arn`으로 사용할 것이므로 메모해 둡니다.

```
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. 이전 단계를 완료한 후 로그 데이터 수신자 계정(222222222222)에서 액세스 정책을 대상과 연결합니다.

이 정책을 사용하면 로그 데이터 발신자 계정(111111111111)이 로그 데이터 수신자 계정(222222222222)만의 대상에 액세스할 수 있습니다. 텍스트 편집기를 사용하여 ~/AccessPolicy .json 파일에 이 정책을 넣을 수 있습니다.

```
{
  "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. 이렇게 하면 대상에 대해 쓰기 액세스 권한을 가진 사람을 정의하는 정책이 생성됩니다. 이 정책은 대상에 액세스하기 위한 logs: PutSubscriptionFilter 작업을 지정해야 합니다. 교차 계정 사용자는 이 PutSubscriptionFilter작업을 사용하여 목적지로 로그 이벤트를 전송합니다.

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
    --access-policy file://~/AccessPolicy.json
```

3단계: 교차 계정 대상에 대한 IAM 권한 추가/검증

AWS 교차 계정 정책 평가 로직에 따르면 계정 간 리소스 (예: 구독 필터의 대상으로 사용되는 Kinesis 또는 Firehose 스트림) 에 액세스하려면 전송 계정에 계정 간 대상 리소스에 대한 명시적 액세스를 제공하는 ID 기반 정책이 있어야 합니다. 정책 평가 로직에 대한 자세한 내용은 교차 계정 정책 평가 로직을 참조하세요.

구독 필터를 생성하는 데 사용하는 IAM 역할 또는 IAM 사용자에게 자격 증명 기반 정책을 연결할 수 있습니다. 전송 계정에 이 정책이 있어야 합니다. 관리자 역할을 사용하여 구독 필터를 만드는 경우 이 단계를 건너뛰고 4단계: 구독 필터 생성(으)로 넘어갈 수 있습니다.

교차 계정에 필요한 IAM 권한을 추가 또는 검증하려면

1. 다음 명령을 입력하여 AWS 로그 명령을 실행하는 데 사용되는 IAM 역할 또는 IAM 사용자를 확인합니다.

   ```
   aws sts get-caller-identity
   ```

   이 명령은 다음과 비슷한 출력을 반환합니다.

   ```
   {
   "UserId": "User ID",
   "Account": "sending account id",
   "Arn": "arn:aws:sending account id:role/user:RoleName/UserName"
   }
   ```

   또는 로 표시된 값을 기록해 두십시오. *RoleNameUserName*

2. 전송 계정에 로그인하고 AWS Management Console 1단계에서 입력한 명령의 출력에 반환된 IAM 역할 또는 IAM 사용자로 연결된 정책을 검색합니다.

3. 이 역할 또는 사용자에게 연결된 정책이 교차 계정 대상 리소스에서 `logs:PutSubscriptionFilter`를 호출할 수 있는 명시적 권한을 제공하는지 확인하세요. 다음 정책 예는 권장 권한을 보여 줍니다.

   다음 정책은 단일 AWS 계정 계정으로만 모든 대상 리소스에 구독 필터를 생성할 수 있는 권한을 제공합니다. 123456789012

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow subscription filters on any resource in one specific
 account",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:*"
            ]
        }
    ]
}
```

   다음 정책은 단일 AWS 계정, 계정으로 이름이 지정된 특정 대상 리소스에만 구독 필터를 만들 수 `sampleDestination` 있는 권한을 제공합니다123456789012.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow subscription filters on one specific resource in one
 specific account",
            "Effect": "Allow",
            "Action": "logs:PutSubscriptionFilter",
            "Resource": [
                "arn:aws:logs:*:*:log-group:*",
                "arn:aws:logs:*:123456789012:destination:sampleDestination"
            ]
        }
    ]
}
```

4단계: 구독 필터 생성

이 예에서는 111111111111인 송신 계정으로 전환하세요. 이제 송신 계정에서 구독 필터를 생성합니다. 이 예시에서는 필터가 AWS CloudTrail 이벤트가 포함된 로그 그룹과 연결되어 '루트' AWS 자격 증명으로 기록된 모든 활동이 이전에 만든 대상으로 전달됩니다. AWS CloudTrail 이벤트를 CloudWatch 로그로 보내는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudWatch 로그에 CloudTrail 이벤트 전송을 참조하십시오.

다음 명령을 입력할 때는3단계: 교차 계정 대상에 대한 IAM 권한 추가/검증에서 IAM 사용자로 로그인하거나 정책을 추가한 IAM 역할을 사용하여 로그인해야 합니다.

```
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"
```

로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상은 다른 지역에 있는 Firehose 스트림과 같은 AWS 리소스를 가리킬 수 있습니다.

로그 이벤트 흐름 검증

구독 필터를 만들면 CloudWatch Logs는 필터 패턴과 일치하는 모든 수신 로그 이벤트를 Firehose 전송 스트림으로 전달합니다. Firehose 전송 스트림에 설정된 시간 버퍼 간격을 기반으로 Amazon S3 버킷에 데이터가 표시되기 시작합니다. 충분한 시간이 지나고 나면 Amazon S3 버킷을 확인하여 데이터를 확인할 수 있습니다. 버킷을 확인하려면 다음 명령을 입력합니다.

```
aws s3api list-objects --bucket 'firehose-test-bucket1'
```

해당 명령은 다음과 비슷하게 출력됩니다.

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

다음 명령을 입력하여 버킷에서 특정 객체를 검색할 수 있습니다. 이전 명령에서 찾은 값으로 key의 값을 바꿉니다.

```
aws s3api get-object --bucket 'firehose-test-bucket1' --key '2021/02/02/08/my-delivery-stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

Amazon S3 객체의 데이터는 gzip 형식으로 압축됩니다. 다음 명령 중 하나를 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

Linux:

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

런타임 시 대상 멤버십 수정

소유한 대상에서 로그 발신자를 추가 또는 제거해야 하는 상황에 직면할 수 있습니다. 새 액세스 정책을 적용하면 대상에서 PutDestinationPolicy작업을 사용할 수 있습니다. 다음 예제에서는 이전에 추가한 계정인 111111111111이 추가적인 로그 데이터 전송을 중단하고 계정 333333333333이 활성화됩니다.

1. 현재 대상 TestDestination 과 연결된 정책을 가져오고 다음 사항을 기록해 둡니다. 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. 계정 111111111111이 중단되고 계정 333333333333이 활성화되었음을 반영하도록 이 정책을 업데이트합니다. 이 정책을 ~/ NewAccessPolicy .json 파일에 넣으십시오.

```
{
  "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. 다음 명령을 NewAccessPolicy사용하여.json 파일에 정의된 정책을 대상에 연결합니다.

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \

    --access-policy file://~/NewAccessPolicy.json
```

이렇게 하면 계정 ID 111111111111에서 로그 이벤트가 비활성화됩니다. 계정 ID 333333333333에서 나온 로그 이벤트는 계정 333333333333의 소유자가 구독 필터를 생성하고 나면 그 즉시 대상으로 이동하기 시작합니다.

Kinesis Data Streams를 사용한 계정 간 리전 계정 수준 구독

교차 계정 구독을 생성할 때 단일 계정 또는 조직을 발신자로 지정할 수 있습니다. 조직을 지정하는 경우 이 절차를 통해 조직의 모든 계정에서 수신자 계정으로 로그를 보낼 수 있습니다.

계정에서 로그 데이터를 공유하려면 로그 데이터 발신자 및 수신자를 설정해야 합니다.

- 로그 데이터 발신자 - 수신자로부터 대상 정보를 가져와 지정된 대상으로 CloudWatch 로그 이벤트를 전송할 준비가 되었음을 Logs에 알립니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 발신자가 가상 AWS 계정 번호 111111111111로 표시됩니다.

  한 조직 내의 여러 계정에서 한 수신자 계정으로 로그를 보내려면 해당 조직의 모든 계정에서 수신자 계정으로 로그를 보낼 수 있는 권한을 부여하는 정책을 만들 수 있습니다. 각 발신자 계정에 대해 별도의 구독 필터를 설정해야 합니다.
- 로그 데이터 수신자 - Kinesis Data Streams 스트림을 캡슐화하는 대상을 설정하고 수신자가 로그 데이터를 받기를 원한다는 것을 CloudWatch Logs에 알립니다. 그런 다음 수신자는 이 대상에 대한 정보를 발신자와 공유합니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 수신자가 가상 AWS 계정 번호 999999999999로 표시됩니다.

계정 간 사용자로부터 로그 이벤트를 받기 시작하려면 로그 데이터 수신자가 먼저 로그 대상을 생성합니다. CloudWatch 각 대상은 다음과 같은 키 요소로 이루어져 있습니다.

대상 이름

생성하고자 하는 대상의 이름입니다.

대상 ARN

구독 피드의 대상으로 사용하려는 AWS 리소스의 Amazon 리소스 이름 (ARN)

역할 ARN

선택한 스트림에 데이터를 넣는 데 필요한 권한을 CloudWatch 로그에 부여하는 AWS Identity and Access Management (IAM) 역할.

액세스 정책

대상에 쓰기 권한이 허용된 사용자들에게 적용되는 IAM 정책 문서(JSON 형식, IAM 정책 문법을 사용해 작성)입니다.

> **Note**
>
> 로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상이 가리키는 AWS 리소스는 다른 리전에 위치할 수 있습니다. 다음 섹션의 예제에서는 모든 리전 관련 리소스가 미국 동부(버지니아 북주)에서 생성됩니다.

주제

새 교차 계정 구독 설정

이러한 섹션의 단계에 따라 새 교차 계정 로그 구독을 설정합니다.

주제

1단계: 대상 생성

> **Important**
>
> 이 절차의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.

이 예에서 로그 데이터 수신자 계정의 계정 ID는 999999999999인 반면, 로그 데이터 발신자 AWS 계정 ID는 111111111111입니다. AWS

이 예제에서는 RecipientStream 라는 Kinesis Data Streams 스트림을 사용하여 대상을 생성하고 CloudWatch Logs가 여기에 데이터를 쓸 수 있도록 하는 역할을 생성합니다.

대상이 생성되면 CloudWatch Logs는 수신자 계정을 대신하여 대상에 테스트 메시지를 보냅니다. 나중에 구독 필터가 활성화되면 CloudWatch Logs는 원본 계정을 대신하여 대상에 로그 이벤트를 보냅니다.

대상을 생성하려면

1. 수신자 계정에서 Kinesis Data Streams에 대상 스트림을 생성합니다. 명령 프롬프트에서 다음과 같이 입력합니다.

```
aws kinesis create-stream --stream-name "RecipientStream" --shard-count 1
```

2. 스트림이 활성 상태가 될 때까지 기다립니다. aws kinesis describe-stream 명령을 사용하여 확인할 수 있습니다. StreamDescription StreamStatus속성. 또한 StreamDescription.StreamArn 값은 나중에 로그에 전달되므로 기록해 CloudWatch 두십시오.

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

스트림이 활성 상태가 될 때까지 1~2분 정도 기다려야 할 수 있습니다.

3. CloudWatch Logs에 스트림에 데이터를 넣을 수 있는 권한을 부여하는 IAM 역할을 생성하세요. 먼저 ~/ TrustPolicyFor cwl.json 파일에 신뢰 정책을 생성해야 합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고 IAM 콘솔은 사용하지 마세요.

   이 정책은 `sourceAccountId`를 지정하여 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 `aws:SourceArn` 글로벌 조건 컨텍스트 키를 포함합니다. 첫 번째 호출에서 소스 계정 ID를 아직 모르는 경우 소스 ARN 필드에 대상 ARN을 넣는 것이 좋습니다. 후속 호출에서는 소스 ARN을 첫 번째 호출에서 수집한 실제 소스 ARN으로 설정해야 합니다. 자세한 정보는 혼동된 대리자 방지을 참조하세요.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        },
        "Action": "sts:AssumeRole"
    }
}
```

4. aws iam create-role 명령을 사용하여 신뢰 정책 파일을 지정하는 IAM 역할을 생성합니다. 반환된 Role.Arn 값은 나중에 Logs에도 전달되므로 기록해 두십시오. CloudWatch

```
aws iam create-role \
--role-name CWLtoKinesisRole \
--assume-role-policy-document file://~/TrustPolicyForCWL.json

{
    "Role": {
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Condition": {
```

```
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                },
                "Principal": {
                    "Service": "logs.amazonaws.com"
                }
            }
        },
        "RoleId": "AAOIIAH450GAB4HC5F431",
        "CreateDate": "2023-05-29T13:46:29.431Z",
        "RoleName": "CWLtoKinesisRole",
        "Path": "/",
        "Arn": "arn:aws:iam::999999999999:role/CWLtoKinesisRole"
    }
}
```

5. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하세요. 먼저 텍스트 편집기를 사용하여 ~/ PermissionsFor cwl.json 파일에 권한 정책을 생성합니다.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kinesis:PutRecord",
      "Resource": "arn:aws:kinesis:region:999999999999:stream/RecipientStream"
    }
  ]
}
```

6. aws iam 명령을 사용하여 권한 정책을 역할에 연결합니다. put-role-policy

```
aws iam put-role-policy \
    --role-name CWLtoKinesisRole \
    --policy-name Permissions-Policy-For-CWL \
    --policy-document file://~/PermissionsForCWL.json
```

7. 스트림이 활성 상태가 되고 IAM 역할을 생성한 후 CloudWatch 로그 대상을 생성할 수 있습니다.

a. 이 단계를 수행해도 액세스 정책이 대상에 연결되는 것은 아니며, 대상 생성을 완료하기 위한 두 단계 중 첫 번째 단계를 완료한 것일 뿐입니다. 페이로드에 DestinationArn반환되는 내용을 기록해 두십시오.

```
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. 7a 단계를 완료한 후 로그 데이터 수신자 계정에서 액세스 정책을 대상과 연결합니다. 이 정책은 로그: PutSubscriptionFilter 작업을 지정하고 발신자 계정에 대상에 액세스할 수 있는 권한을 부여해야 합니다.

정책은 로그를 보내는 AWS 계정에 권한을 부여합니다. 정책에서 이 계정 하나만 지정할 수 있으며, 또는 발신자 계정이 조직의 구성원인 경우 정책은 해당 조직의 조직 ID를 지정할 수 있습니다. 이렇게 하면 정책 하나만 생성하여 한 조직의 여러 계정이 이 대상 계정으로 로그를 보내도록 할 수 있습니다.

텍스트 편집기를 사용하여 이름이 ~/AccessPolicy.json이고 다음 정책 문 중 하나를 포함한 파일을 생성합니다.

이 첫 번째 예제 정책은 ID가 o-1234567890인 조직의 모든 계정이 수신자 계정으로 로그를 보내도록 허용합니다.

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Sid" : "",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
```

```
            "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination",
            "Condition": {
               "StringEquals" : {
                   "aws:PrincipalOrgID" : ["o-1234567890"]
                }
            }
        }
    ]
}
```

다음 예에서는 로그 데이터 발신자 계정(111111111111)에서만 로그 데이터 수신자 계정으로 로그를 보내도록 허용합니다.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111111111111"
      },
      "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
      "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination"
    }
  ]
}
```

c. 이전 단계에서 생성한 정책을 대상에 연결합니다.

```
aws logs put-destination-policy \
    --destination-name "testDestination" \
    --access-policy file://~/AccessPolicy.json
```

*# ### ### #### ID# 1111111111# AWS ### #### ARN arn:aws:logs: # #:999999999999:###:####### ### # **PutSubscriptionFilter**# ####.* 다른 사용자가 이 목적지를 상대로 전화를 걸려고 시도하면 거부됩니다. PutSubscriptionFilter

액세스 정책에 대한 사용자 권한의 유효성을 검사하려면 IAM 사용 설명서의 정책 검사기 사용을 참조하세요.

작업을 마쳤으면 교차 계정 권한을 사용하기 AWS Organizations 위해 사용하는 경우 다음 단계를 따르세요. 2단계: (조직을 사용하는 경우에만) IAM 역할 생성 Organizations를 사용하는 대신 다른 계정에 직접 권한을 부여하는 경우 해당 단계를 건너뛰고 3단계: 계정 수준의 구독 필터 정책 생성 단계로 넘어갑니다.

2단계: (조직을 사용하는 경우에만) IAM 역할 생성

이전 섹션에서 111111111111 계정이 속한 조직에 권한을 부여하는 액세스 정책을 사용하여 대상을 생성한 경우, 111111111111 계정에 직접 권한을 부여하는 대신 이 섹션의 단계를 따릅니다. 그렇지 않다면 3단계: 계정 수준의 구독 필터 정책 생성 단계로 건너뛰어도 됩니다.

이 섹션의 단계는 IAM 역할을 생성합니다. IAM 역할을 생성하면 발신자 계정에 수신자를 대상으로 구독 필터를 생성할 권한이 있는지 여부를 가정하고 검증할 CloudWatch 수 있습니다.

발신자 계정에서 이 섹션의 단계를 수행합니다. 역할은 발신자 계정에 있어야 하며 구독 필터에서 이 역할의 ARN을 지정합니다. 이 예제에서 발신자 계정은 111111111111입니다.

AWS Organizations를 사용하여 교차 계정 로그 구독에 필요한 IAM 역할 생성

1. /TrustPolicyForCWLSubscriptionFilter.json 파일에 다음 트러스트 정책을 생성합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고, IAM 콘솔은 사용하지 마세요.

```
{
  "Statement": {
    "Effect": "Allow",
    "Principal": { "Service": "logs.amazonaws.com" },
    "Action": "sts:AssumeRole"
  }
}
```

2. 이 정책을 사용하는 IAM 역할을 생성합니다. 명령에 의해 반환되는 Arn 값은 이 절차의 뒷부분에 필요하므로 메모해 둡니다. 이 예제에서 생성하는 역할의 이름으로는 CWLtoSubscriptionFilterRole을 사용합니다.

```
aws iam create-role \
     --role-name CWLtoSubscriptionFilterRole \
     --assume-role-policy-document file://~/
TrustPolicyForCWLSubscriptionFilter.json
```

3. 권한 정책을 생성하여 CloudWatch Logs가 계정에서 수행할 수 있는 작업을 정의하십시오.

a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 ~/PermissionsForCWLSubscriptionFilter.json인 파일로 생성합니다.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

b. 다음 명령을 입력하여 방금 생성한 권한 정책을 2단계에서 생성한 역할에 연결합니다.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

작업을 마쳤으면 3단계: 계정 수준의 구독 필터 정책 생성 단계로 넘어갑니다.

3단계: 계정 수준의 구독 필터 정책 생성

대상을 생성하고 나면 로그 데이터 수신자 계정에서 다른 AWS 계정이 로그 이벤트를 동일한 대상으로 전송할 수 있도록 대상 ARN(arn:aws:logs:us-east-1:999999999999:destination:testDestination)을 이들과 공유할 수 있습니다. 그러면 이러한 다른 전송 계정 사용자는 이 대상에 해당되는 로그 그룹에 대한 구독 필터를 생성합니다. 그 즉시 구독 필터는 실시간으로 선택한 로그 그룹에서 지정된 스트림으로 로그 데이터를 이동시키기 시작합니다.

> **Note**
>
> 전체 조직에 구독 필터에 대한 권한을 부여하는 경우 2단계: (조직을 사용하는 경우에만) IAM 역할 생성에서 생성한 IAM 역할의 ARN을 사용해야 합니다.

다음 예시에서는 보내는 계정에 계정 수준의 구독 필터 정책을 만듭니다. 필터는 발신자 111111111111 계정과 연결되어 필터 및 선택 기준과 일치하는 모든 로그 이벤트가 이전에 만든 대상으로 전달됩니다. 해당 대상은 "“라는 스트림을 캡슐화합니다. RecipientStream

이 selection-criteria 필드는 선택 사항이지만 구독 필터에서 무한 로그 재귀를 일으킬 수 있는 로그 그룹을 제외하는 데 중요합니다. 이 문제와 제외할 로그 그룹 결정에 대한 자세한 내용은 을 참조하십시오로그 재귀 방지. 현재 지원되는 연산자는 NOT IN뿐입니다selection-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"
```

발신자 계정의 로그 그룹과 대상이 같은 AWS 지역에 있어야 합니다. 하지만 대상은 다른 지역에 있는 Kinesis Data Streams 스트림과 같은 AWS 리소스를 가리킬 수 있습니다.

로그 이벤트 흐름 검증

계정 수준 구독 필터 정책을 만들면 CloudWatch Logs는 필터 패턴 및 선택 기준과 일치하는 모든 수신 로그 이벤트를 대상 스트림 내에 캡슐화된 스트림 "“에 전달합니다. RecipientStream 대상 소유자는 aws kinesis 명령을 사용하여 Kinesis Data Streams 샤드를 가져오고 aws kinesis get-records get-shard-iterator 명령을 사용하여 일부 Kinesis 데이터 스트림 레코드를 가져와서 이러한 문제가 발생하는지 확인할 수 있습니다.

```
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**
>
> Kinesis Data Streams가 데이터를 반환하기 시작하기 전에 `get-records` 명령을 몇 번 다시 실행해야 할 수 있습니다.

Kinesis Data Streams 레코드 어레이에서 응답을 확인할 수 있습니다. Kinesis Data Streams 레코드의 데이터 속성은 gzip 형식으로 압축된 다음 base64로 인코딩됩니다. 다음 Unix 명령을 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

```
echo -n "<Content of Data>" | base64 -d | zcat
```

디코딩 및 압축 해제된 base64 데이터는 다음 구조를 가진 JSON으로 포맷됩니다.

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

데이터 구조의 주요 요소는 다음과 같습니다.

messageType

데이터 메시지는 "DATA_MESSAGE" 유형을 사용합니다. 가끔 CloudWatch 로그에서 “CONTROL_MESSAGE” 유형의 Kinesis Data Streams 레코드를 내보낼 수 있는데, 이는 주로 대상에 도달할 수 있는지 확인하기 위한 것입니다.

owner

원본 로그 데이터의 AWS 계정 ID.

logGroup

원본 로그 데이터의 로그 그룹 이름입니다.

logStream

원본 로그 데이터의 로그 스트림 이름입니다.

subscriptionFilters

원본 로그 데이터과 일치한 구독 필터 이름 목록입니다.

logEvents

로그 이벤트 레코드 어레이 형태로 표현되는 실제 로그 데이터입니다. "ID" 속성은 모든 로그 이벤트의 고유 식별자입니다.

정책 수준

정책이 시행된 수준. “ACCOUNT_LEVEL_POLICY”는 계정 수준의 구독 필터 정책을 `policyLevel` 위한 것입니다.

런타임 시 대상 멤버십 수정

소유한 대상에서 몇몇 사용자의 멤버십을 추가 또는 제거해야 하는 상황에 직면할 수 있습니다. 새 액세스 정책과 함께 대상에 put-destination-policy 명령을 사용할 수 있습니다. 다음 예제에서는 이전에 추가한 계정인 111111111111이 추가적인 로그 데이터 전송을 중단하고 계정 222222222222가 활성화됩니다.

1. 현재 대상 TestDestination 과 연결된 정책을 가져오고 다음 사항을 기록해 둡니다. 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. 계정 111111111111이 중단되고 계정 222222222222가 활성화되었음을 반영하도록 이 정책을 업데이트합니다. 이 정책을 ~/ NewAccessPolicy .json 파일에 넣으십시오.

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "222222222222"
      },
      "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
      "Resource" : "arn:aws:logs:region:999999999999:destination:testDestination"
    }
```

```
  ]
}
```

3. NewAccessPolicy.json 파일에 정의된 정책을 대상에 PutDestinationPolicy연결하려면 호출하십시오.

```
aws logs put-destination-policy \
--destination-name "testDestination" \
--access-policy file://~/NewAccessPolicy.json
```

이렇게 하면 계정 ID 111111111111에서 로그 이벤트가 비활성화됩니다. 계정 ID 222222222222에서 나온 로그 이벤트는 계정 222222222222 의 소유자가 구독 필터를 생성하고 나면 그 즉시 대상으로 이동하기 시작합니다.

기존 교차 계정 구독 업데이트

현재 대상 계정이 특정 발신자 계정에만 권한을 부여하는 교차 계정 로그 구독이 있고 대상 계정이 조직의 모든 계정에 대한 액세스 권한을 부여하도록 이 구독을 업데이트하려는 경우 이 섹션의 단계를 따릅니다.

주제

- 1단계: 구독 필터 업데이트
- 2단계: 기존 대상 액세스 정책 업데이트

1단계: 구독 필터 업데이트

> **Note**
>
> 이 단계는 AWS 서비스에서 로깅 활성화에 나열된 서비스에서 생성된 로그에 대한 교차 계정 구독에만 필요합니다. 해당 로그 그룹 중 하나에서 생성된 로그로 작업하지 않는 경우 2단계: 기존 대상 액세스 정책 업데이트 섹션으로 건너뛸 수 있습니다.

경우에 따라 대상 계정으로 로그를 보내는 모든 발신자 계정의 구독 필터를 업데이트해야 합니다. 이 업데이트에는 발신자 계정에 수신자 계정에 로그를 전송할 권한이 있다고 가정하고 CloudWatch 검증할 수 있는 IAM 역할이 추가되었습니다.

교차 계정 구독 권한에 대해 조직 ID를 사용하도록 업데이트하려는 모든 발신자 계정에 대해서는 이 섹션의 단계를 따릅니다.

이 섹션의 예제에서 두 개의 111111111111 계정 및 222222222222 계정은 999999999999 계정에 로그를 전송하기 위해 생성된 구독 필터를 이미 가지고 있습니다. 기존 구독 필터 값은 다음과 같습니다.

```
## Existing Subscription Filter parameter values
{
    "DestinationArn": "arn:aws:logs:region:999999999999:destination:testDestination",
    "FilterPattern": "{$.userIdentity.type = Root}",
    "Distribution": "Random"
}
```

현재 구독 필터 파라미터 값을 찾아야 하는 경우 다음 명령을 입력합니다.

```
aws logs describe-account-policies \
--policy-type "SUBSCRIPTION_FILTER_POLICY" \
--policy-name "CrossAccountStreamsExamplePolicy"
```

교차 계정 로그 권한에 대한 조직 ID 사용을 시작하도록 구독 필터 업데이트

1. ~/TrustPolicyForCWL.json 파일에 다음 트러스트 정책을 생성합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성하고, IAM 콘솔은 사용하지 마세요.

   ```
   {
     "Statement": {
       "Effect": "Allow",
       "Principal": { "Service": "logs.amazonaws.com" },
       "Action": "sts:AssumeRole"
     }
   }
   ```

2. 이 정책을 사용하는 IAM 역할을 생성합니다. 명령에 의해 반환되는 Arn 값의 Arn 값은 이 절차의 뒷부분에 필요하므로 메모해 둡니다. 이 예제에서 생성하는 역할의 이름으로는 CWLtoSubscriptionFilterRole을 사용합니다.

   ```
   aws iam create-role
       \ --role-name CWLtoSubscriptionFilterRole
       \ --assume-role-policy-document file://~/TrustPolicyForCWL.json
   ```

3. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하십시오.

   a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 `/PermissionsForCWLSubscriptionFilter.json`인 파일로 생성합니다.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:region:111111111111:log-
group:LogGroupOnWhichSubscriptionFilterIsCreated:*"
        }
    ]
}
```

   b. 다음 명령을 입력하여 방금 생성한 권한 정책을 2단계에서 생성한 역할에 연결합니다.

```
aws iam put-role-policy
    --role-name CWLtoSubscriptionFilterRole
    --policy-name Permissions-Policy-For-CWL-Subscription-filter
    --policy-document file://~/PermissionsForCWLSubscriptionFilter.json
```

4. 다음 명령을 입력하여 구독 필터 정책을 업데이트합니다.

```
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"
```

2단계: 기존 대상 액세스 정책 업데이트

모든 발신자 계정에서 구독 필터를 업데이트한 후, 수신자 계정에서 대상 액세스 정책을 업데이트할 수 있습니다.

다음 예제에서 수신자 계정은 `999999999999`이며 대상의 이름은 `testDestination`입니다.

업데이트를 통해 ID가 o-1234567890인 조직의 모든 계정이 수신자 계정으로 로그를 보낼 수 있습니다. 구독 필터가 생성된 계정만이 실제로 수신자 계정에 로그를 보냅니다.

수신자 계정의 대상 액세스 정책을 업데이트하여 권한에 대한 조직 ID 사용

1. 수신자 계정에서 텍스트 편집기를 사용하여 다음 내용을 포함한 ~/AccessPolicy.json 파일을 생성합니다.

```
{
"Version" : "2012-10-17",
    "Statement" : [
        {
"Sid" : "",
            "Effect" : "Allow",
            "Principal" : "*",
            "Action" : ["logs:PutSubscriptionFilter","logs:PutAccountPolicy"],
            "Resource" :
 "arn:aws:logs:region:999999999999:destination:testDestination",
            "Condition": {
"StringEquals" : {
"aws:PrincipalOrgID" : ["o-1234567890"]
                }
            }
        }
    ]
}
```

2. 다음 명령을 입력하여 방금 생성한 정책을 기존 대상에 연결합니다. 특정 AWS 계정 ID을 나열하는 액세스 정책 대신 조직 ID로 액세스 정책을 사용하도록 대상을 업데이트하려면 force 파라미터를 포함시킵니다.

> **Warning**
>
> 에 나열된 AWS 서비스에서 전송한 로그로 작업하는 경우 이 단계를 수행하기 전에 먼저 에 설명된 대로 모든 발신자 계정의 구독 필터를 업데이트해야 합니다. AWS 서비스에서 로깅 활성화 1단계: 구독 필터 업데이트

```
aws logs put-destination-policy
    \ --destination-name "testDestination"
    \ --access-policy file://~/AccessPolicy.json
```

```
    \ --force
```

Firehose를 사용한 교차 계정 지역 계정 수준 구독

계정에서 로그 데이터를 공유하려면 로그 데이터 발신자 및 수신자를 설정해야 합니다.

- 로그 데이터 발신자 - 수신자로부터 대상 정보를 받고 지정된 대상으로 CloudWatch 로그 이벤트를 전송할 준비가 되었음을 Logs에 알립니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 발신자가 가상 AWS 계정 번호 1111111111로 표시됩니다.
- 로그 데이터 수신자 - Kinesis Data Streams 스트림을 캡슐화하는 대상을 설정하고 수신자가 로그 데이터를 받기를 원한다는 것을 CloudWatch Logs에 알립니다. 그런 다음 수신자는 이 대상에 대한 정보를 발신자와 공유합니다. 이 섹션의 나머지 부분에서의 절차에서는 로그 데이터 수신자가 가상 AWS 계정 번호 2222222222로 표시됩니다.

이 섹션의 예시에서는 Amazon S3 스토리지와 함께 Firehose 전송 스트림을 사용합니다. 다양한 설정으로 Firehose 전송 스트림을 설정할 수도 있습니다. 자세한 내용은 Firehose 전송 스트림 생성을 참조하십시오.

> **Note**
>
> 로그 그룹과 대상은 같은 AWS 지역에 있어야 합니다. 하지만 대상이 가리키는 AWS 리소스는 다른 리전에 위치할 수 있습니다.

> **Note**
>
> 동일한 계정 및 지역 간 전송 스트림에 대한 Firehose 구독 필터가 지원됩니다.

주제

1단계: Firehose 전송 스트림 생성

> **Important**
>
> 다음 단계를 완료하기 전에 액세스 정책을 사용해야 Firehose가 Amazon S3 버킷에 액세스할 수 있습니다. 자세한 내용은 Amazon Data Firehose 개발자 안내서의 액세스 제어를 참조하십시오.
> 이 섹션(1단계)의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.
> 미국 동부(버지니아 북부)가 다음 샘플 명령에 사용됩니다. 이 리전을 배포에 적합한 리전으로 바꿉니다.

대상으로 사용할 Firehose 전송 스트림을 만들려면

1. Amazon S3 버킷 생성:

```
aws s3api create-bucket --bucket firehose-test-bucket1 --create-bucket-
configuration LocationConstraint=us-east-1
```

2. 데이터를 버킷에 넣을 수 있는 권한을 Firehose에 부여하는 IAM 역할을 생성합니다.

   a. 먼저 텍스트 편집기를 사용하여 신뢰 정책을 ~/TrustPolicyForFirehose.json 파일로 생성합니다.

```
{ "Statement": { "Effect": "Allow", "Principal": { "Service":
 "firehose.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition":
 { "StringEquals": { "sts:ExternalId":"222222222222" } } } }
```

   b. IAM 역할을 생성하여 방금 생성한 신뢰 정책 파일을 지정합니다.

```
aws iam create-role \
    --role-name FirehosetoS3Role \
    --assume-role-policy-document file://~/TrustPolicyForFirehose.json
```

   c. 이 명령의 출력은 다음과 비슷합니다. 역할 이름과 역할 ARN을 기록해 둡니다.

```
{
    "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. 권한 정책을 만들어 Firehose가 계정에서 수행할 수 있는 작업을 정의하세요.

   a. 먼저 텍스트 편집기를 사용하여 다음 권한 정책을 이름이 ~/PermissionsForFirehose.json인 파일로 생성합니다. 사용 사례에 따라 이 파일에 권한을 더 추가해야 할 수도 있습니다.

```
{
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::firehose-test-bucket1",
            "arn:aws:s3:::firehose-test-bucket1/*"
        ]
    }]
}
```

   b. 다음 명령을 입력하여 방금 생성한 권한 정책을 IAM 역할에 연결합니다.

```
aws iam put-role-policy --role-name FirehosetoS3Role --policy-name
 Permissions-Policy-For-Firehose-To-S3 --policy-document file://~/
PermissionsForFirehose.json
```

4. 다음 명령어를 입력하여 Firehose 전송 스트림을 생성합니다. *my-role-arn*및 를 *my-bucket-arn*배포에 맞는 올바른 값으로 바꾸십시오.

```
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:::firehose-test-bucket1"}'
```

다음과 같이 출력됩니다

```
{
    "DeliveryStreamARN": "arn:aws:firehose:us-east-1:222222222222:deliverystream/
my-delivery-stream"
}
```

2단계: 대상 생성

> **Important**
>
> 이 절차의 모든 단계는 로그 데이터 수신자 계정에서 수행해야 합니다.

대상이 생성되면 CloudWatch Logs는 수신자 계정을 대신하여 대상에 테스트 메시지를 보냅니다. 나중에 구독 필터가 활성화되면 CloudWatch Logs는 원본 계정을 대신하여 대상에 로그 이벤트를 보냅니다.

대상을 생성하려면

1. 에서 1단계: Firehose 전송 스트림 생성 만든 Firehose 스트림이 활성화될 때까지 기다리세요. 다음 명령어를 사용하여 확인할 수 있습니다. StreamDescription StreamStatus속성.

```
aws firehose describe-delivery-stream --delivery-stream-name "my-delivery-stream"
```

또한, 기록해 두십시오 DeliveryStreamDescription. DeliveryStreamARN 값입니다. 이후 단계에서 사용해야 하기 때문입니다. 이 명령의 샘플 출력:

```
{
    "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:::firehose-test-bucket1",
                    "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:::firehose-test-bucket1",
                    "BufferingHints": {
                        "SizeInMBs": 5,
                        "IntervalInSeconds": 300
                    },
                    "CompressionFormat": "UNCOMPRESSED",
                    "EncryptionConfiguration": {
```

```
                        "NoEncryptionConfig": "NoEncryption"
                    },
                    "CloudWatchLoggingOptions": {
                        "Enabled": false
                    },
                    "S3BackupMode": "Disabled"
                }
            }
        ],
        "HasMoreDestinations": false
    }
}
```

전송 스트림이 활성 상태가 될 때까지 1~2분 정도 기다려야 할 수 있습니다.

2. 전송 스트림이 활성화되면 Firehose 스트림에 데이터를 넣을 권한을 CloudWatch Logs에 부여하는 IAM 역할을 생성합니다. 먼저 TrustPolicyFor~/ cwl.json 파일에 신뢰 정책을 생성해야 합니다. 텍스트 편집기를 사용하여 이 정책을 생성하세요. CloudWatch 로그 엔드포인트에 대한 자세한 내용은 Amazon CloudWatch Logs 엔드포인트 및 할당량을 참조하십시오.

   이 정책은 `sourceAccountId`를 지정하여 혼동된 대리자 보안 문제를 방지하는 데 도움이 되는 `aws:SourceArn` 글로벌 조건 컨텍스트 키를 포함합니다. 첫 번째 호출에서 소스 계정 ID를 아직 모르는 경우 소스 ARN 필드에 대상 ARN을 넣는 것이 좋습니다. 후속 호출에서는 소스 ARN을 첫 번째 호출에서 수집한 실제 소스 ARN으로 설정해야 합니다. 자세한 정보는 혼동된 대리자 방지을 참조하세요.

```
{
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "logs.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringLike": {
                "aws:SourceArn": [
                    "arn:aws:logs:region:sourceAccountId:*",
                    "arn:aws:logs:region:recipientAccountId:*"
                ]
            }
        }
     }
```

```
}
```

3. aws iam create-role 명령을 사용하여 IAM 역할을 생성해 방금 만든 신뢰 정책 파일을 지정합니다.

```
aws iam create-role \
      --role-name CWLtoKinesisFirehoseRole \
      --assume-role-policy-document file://~/TrustPolicyForCWL.json
```

다음은 출력 샘플입니다. 이후 단계에서 사용해야 하므로 반환된 Role.Arn 값을 메모해 둡니다.

```
{
    "Role": {
        "Path": "/",
        "RoleName": "CWLtoKinesisFirehoseRole",
        "RoleId": "AROAR3BXASEKYJYWF243H",
        "Arn": "arn:aws:iam::222222222222:role/CWLtoKinesisFirehoseRole",
        "CreateDate": "2023-02-02T08:10:43+00:00",
        "AssumeRolePolicyDocument": {
            "Statement": {
                "Effect": "Allow",
                "Principal": {
                    "Service": "logs.amazonaws.com"
                },
                "Action": "sts:AssumeRole",
                "Condition": {
                    "StringLike": {
                        "aws:SourceArn": [
                            "arn:aws:logs:region:sourceAccountId:*",
                            "arn:aws:logs:region:recipientAccountId:*"
                        ]
                    }
                }
            }
        }
    }
}
```

4. 권한 정책을 생성하여 CloudWatch 로그가 계정에서 수행할 수 있는 작업을 정의하십시오. 먼저 텍스트 편집기를 사용하여 ~/ PermissionsFor cwl.json 파일에 권한 정책을 생성합니다.

```
{
    "Statement":[
```

```
      {
        "Effect":"Allow",
        "Action":["firehose:*"],
        "Resource":["arn:aws:firehose:region:222222222222:*"]
      }
    ]
}
```

5. 다음 명령을 입력하여 권한 정책을 역할에 연결합니다.

```
aws iam put-role-policy --role-name CWLtoKinesisFirehoseRole --policy-name
 Permissions-Policy-For-CWL --policy-document file://~/PermissionsForCWL.json
```

6. Firehose 전송 스트림이 활성 상태가 되고 IAM 역할을 생성한 후 Logs 대상을 생성할 수 있습니다. CloudWatch

   a. 이 단계를 수행했다고 액세스 정책이 대상에 연결되는 것은 아니며, 대상 생성을 완료하기 위한 두 단계 중 첫 번째 단계일 뿐입니다. 페이로드에서 반환된 새 대상의 ARN은 이후 단계에서 `destination.arn`으로 사용할 것이므로 메모해 둡니다.

```
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. 이전 단계를 완료한 후 로그 데이터 수신자 계정(222222222222)에서 액세스 정책을 대상과 연결합니다. 이 정책을 사용하면 로그 데이터 발신자 계정(111111111111)이 로그 데이터 수신자 계정(222222222222)만의 대상에 액세스할 수 있습니다. 텍스트 편집기를 사용하여 이 정책을 파일에 넣을 수 있습니다. `~/AccessPolicy.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. 이렇게 하면 대상에 대해 쓰기 액세스 권한을 가진 사람을 정의하는 정책이 생성됩니다. 이 정책은 대상에 액세스하기 위한 `logs:PutSubscriptionFilter` 및 `logs:PutAccountPolicy` 작업을 지정해야 합니다. 교차 계정 사용자는 `PutSubscriptionFilter` 및 `PutAccountPolicy` 작업을 사용하여 목적지로 로그 이벤트를 전송합니다.

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \
    --access-policy file://~/AccessPolicy.json
```

3단계: 계정 수준의 구독 필터 정책 생성

이 예에서는 111111111111인 송신 계정으로 전환하세요. 이제 보내는 계정에 계정 수준의 구독 필터 정책을 생성해 보겠습니다. 이 예제에서 필터는 두 개의 로그 그룹을 제외한 모든 로그 그룹의 문자열을 `ERROR` 포함하는 모든 로그 이벤트가 이전에 만든 대상으로 전달되도록 합니다.

```
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"
```

보내는 계정의 로그 그룹과 대상이 같은 AWS 지역에 있어야 합니다. 하지만 대상은 다른 지역에 있는 Firehose 스트림과 같은 AWS 리소스를 가리킬 수 있습니다.

로그 이벤트 흐름 검증

구독 필터를 만들면 CloudWatch Logs는 필터 패턴 및 선택 기준과 일치하는 모든 수신 로그 이벤트를 Firehose 전송 스트림으로 전달합니다. Firehose 전송 스트림에 설정된 시간 버퍼 간격을 기반으로 Amazon S3 버킷에 데이터가 표시되기 시작합니다. 충분한 시간이 지나고 나면 Amazon S3 버킷을 확인하여 데이터를 확인할 수 있습니다. 버킷을 확인하려면 다음 명령을 입력합니다.

```
aws s3api list-objects --bucket 'firehose-test-bucket1'
```

해당 명령은 다음과 비슷하게 출력됩니다.

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

다음 명령을 입력하여 버킷에서 특정 객체를 검색할 수 있습니다. 이전 명령에서 찾은 값으로 key의 값을 바꿉니다.

```
aws s3api get-object --bucket 'firehose-test-bucket1' --key '2021/02/02/08/my-delivery-
stream-1-2021-02-02-08-55-24-5e6dc317-071b-45ba-a9d3-4805ba39c2ba' testfile.gz
```

Amazon S3 객체의 데이터는 gzip 형식으로 압축됩니다. 다음 명령 중 하나를 사용하여 명령줄에서 원시 데이터를 검토할 수 있습니다.

Linux:

```
zcat testfile.gz
```

macOS:

```
zcat <testfile.gz
```

런타임 시 대상 멤버십 수정

소유한 대상에서 로그 발신자를 추가 또는 제거해야 하는 상황에 직면할 수 있습니다. 새 액세스 정책과 함께 목적지의 PutDestinationPolicy및 PutAccountPolicy 작업을 사용할 수 있습니다. 다음 예제에서는 이전에 추가한 계정인 111111111111이 추가적인 로그 데이터 전송을 중단하고 계정 333333333333이 활성화됩니다.

1. 현재 대상 TestDestination 과 연결된 정책을 가져오고 다음 사항을 기록해 둡니다. AccessPolicy

   ```
   aws logs describe-destinations \
       --destination-name-prefix "testFirehoseDestination"
   ```

   반환된 데이터는 다음과 같을 수 있습니다.

   ```
   {
       "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. 계정 111111111111이 중단되고 계정 333333333333이 활성화되었음을 반영하도록 이 정책을 업데이트합니다. ~/ NewAccessPolicy .json 파일에 다음 정책을 입력합니다.

```
{
  "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. 다음 명령을 NewAccessPolicy사용하여.json 파일에 정의된 정책을 대상에 연결합니다.

```
aws logs put-destination-policy \
    --destination-name "testFirehoseDestination" \

    --access-policy file://~/NewAccessPolicy.json
```

이렇게 하면 계정 ID 111111111111에서 로그 이벤트가 비활성화됩니다. 계정 ID 333333333333에서 나온 로그 이벤트는 계정 333333333333의 소유자가 구독 필터를 생성하고 나면 그 즉시 대상으로 이동하기 시작합니다.

혼동된 대리자 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS크로스 서비스 사칭으로 인해 대리인 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(호출하는 서비스)가 다른 서비스(호출되는 서비스)를 호출할 때 발생할 수 있습니다. 직접적으로 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 계정 내 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 통해 모든 서비스의 데이터를 보호하는 데 도움이 되는 도구를 AWS 제공합니다.

리소스 정책에서 aws:SourceArn, aws:SourceAccountaws:SourceOrgID, 및 aws:SourceOrgPaths글로벌 조건 컨텍스트 키를 사용하여 다른 서비스에 리소스에 부여하는 권한을 제한하는 것이 좋습니다. aws:SourceArn을 사용하면 하나의 리소스만 교차 서비스 액세스 권한과 연결됩니다. aws:SourceAccount를 사용하면 해당 계정의 모든 리소스가 교차 서비스 사용 권한과 연결됩니다. aws:SourceOrgID를 사용하면 조직 내 모든 계정의 모든 리소스가 교차 서비스 사용 권한과 연결될 수 있습니다. aws:SourceOrgPaths를 사용하면 AWS Organizations 경로 내 계정의 모든 리소스가 교차 서비스 사용 권한과 연결됩니다. 경로 사용 및 이해에 대한 자세한 내용은 AWS Organizations 엔티티 경로 이해를 참조하십시오.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 arn:aws:*servicename*:*:*123456789012*:*입니다.

만약 aws:SourceArn 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우, 권한을 제한하려면 두 aws:SourceAccount 및 aws:SourceArn를 모두 사용해야 합니다.

혼동된 대리자 문제로부터 보호하려면 리소스 기반 정책에서 리소스의 조직 ID 또는 조직 경로와 함께 aws:SourceOrgID 또는 aws:SourceOrgPaths 전역 조건 컨텍스트 키를 사용하세요. aws:SourceOrgID 또는 aws:SourceOrgPaths 키가 포함된 정책에는 올바른 계정이 자동으로 포함되며 조직에서 계정을 추가, 제거 또는 이동할 때 정책을 수동으로 업데이트할 필요가 없습니다.

Kinesis Data Streams 및 Firehose에 데이터를 쓸 수 있도록 CloudWatch 로그에 대한 액세스 권한을 부여하는 정책을 설명하고 aws: SourceArn 글로벌 조건 컨텍스트 키를 사용하여 혼동되는 부정 문제를 방지하는 방법을 2단계: 대상 생성 보여줍니다. 1단계: 대상 생성

로그 재귀 방지

구독 필터를 사용하면 무한 로그 재귀가 발생하여 방지하지 않으면 CloudWatch 로그와 대상 모두에서 수집 요금이 크게 증가할 위험이 있습니다. 구독 필터가 구독 전송 워크플로의 결과로 로그 이벤트를 수신하는 로그 그룹과 연결된 경우 이 문제가 발생할 수 있습니다. 로그 그룹에 수집된 로그는 대상으로 전달되므로 로그 그룹에서 더 많은 로그를 수집하게 되고 이 로그가 다시 대상으로 전달되어 재귀 루프가 발생합니다.

예를 들어 대상을 Firehose로 지정하여 Amazon S3에 로그 이벤트를 전송하는 구독 필터를 생각해 보십시오. 또한 Amazon S3로 전송된 새 이벤트를 처리하고 일부 로그를 자체적으로 생성하는 Lambda 함수도 있습니다. 구독 필터가 Lambda 함수의 로그 그룹에 적용되면 함수에서 생성된 로그 이벤트가

대상에 있는 Firehose와 Amazon S3로 전달되고, 그러면 함수가 다시 호출되어 더 많은 로그가 생성되어 Firehose와 Amazon S3로 전달되어 함수가 다시 호출되는 등의 문제가 발생합니다. 이는 무한 루프로 발생하며, 이로 인해 로그 수집, Firehose 및 Amazon S3에 대한 요금이 예기치 않게 증가합니다.

Lambda 함수가 Logs에 대해 흐름 로그가 활성화된 VPC에 연결된 경우 VPC의 CloudWatch 로그 그룹도 로그 재귀를 일으킬 수 있습니다.

구독 전송 워크플로의 일부인 로그 그룹에는 구독 필터를 적용하지 않는 것이 좋습니다. 계정 수준 구독 필터의 경우 `PutAccountPolicy` API의 `selectionCriteria` 매개변수를 사용하여 정책에서 이러한 로그 그룹을 제외할 수 있습니다.

로그 그룹을 제외할 때는 로그를 생성하고 구독 전송 워크플로의 일부일 수 있는 다음 AWS 서비스를 고려해 보세요.

- 아마존 EC2 (파게이트 탑재)
- Lambda
- AWS Step Functions
- 로그에 사용할 수 있는 Amazon VPC 흐름 로그 CloudWatch

> **Note**
>
> Lambda 대상의 로그 그룹에서 생성된 로그 이벤트는 계정 수준 구독 필터 정책을 위해 Lambda 함수로 다시 전달되지 않습니다. 이 경우 계정 구독 정책에 대상 Lambda 함수의 로그 그룹을 `selectionCriteria` 사용하지 않아도 됩니다.

지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail에 대한 필터 패턴 구문

> **Note**
>
> Amazon CloudWatch Logs Insights 쿼리 언어를 사용하여 로그 그룹을 쿼리하는 방법에 대한 자세한 내용은 을 참조하십시오CloudWatch 로그 인사이트 쿼리 구문.

CloudWatch 로그를 사용하면 지표 필터를 사용하여 로그 데이터를 실행 가능한 지표로 변환하고, 구독 필터를 사용하여 로그 이벤트를 다른 AWS 서비스로 라우팅하고, 로그 이벤트를 필터링하여 로그 이벤트를 검색하며, Live Tail을 사용하여 로그를 수집할 때 실시간으로 대화식으로 확인할 수 있습니다.

필터 패턴은 지표 필터, 구독 필터, 필터 로그 이벤트 및 Live Tail을 사용하여 로그 이벤트에서 일치하는 용어를 검색하는 구문을 구성합니다. 용어는 단어, 정확한 문구 또는 숫자 값일 수 있습니다. 정규식을 사용하여 독립 실행형 필터 패턴을 생성하거나 JSON 및 공백으로 구분된 필터 패턴에 정규식을 통합할 수 있습니다.

일치시키려는 용어를 사용하여 필터 패턴을 생성합니다. 필터 패턴은 정의한 용어가 포함된 로그 이벤트만 반환합니다. 콘솔에서 필터 패턴을 테스트할 수 있습니다. CloudWatch

주제

지원되는 정규식 구문

지원되는 정규식 구문

정규식을 사용하여 로그 데이터를 검색하고 필터링하는 경우 표현식을 %로 묶어야 합니다.

정규식을 포함하는 필터 패턴에는 다음만 포함할 수 있습니다.

- 영숫자 - 영숫자는 문자(A~Z 또는 a~z) 또는 숫자(0~9)에 해당하는 문자입니다.
- 지원되는 기호 문자 – 여기에는 '_', '#', '=', '@','/', ';', ',' 및 '-'가 포함됩니다. 예를 들어, `%something!%`는 '!'가 지원되지 않으므로 거부됩니다.
- 지원되는 연산자 - 여기에는 '^', '$', '?', '[', ']', '{', '}', '|', '\', '*', '+' 및 '.'가 포함됩니다.

(및) 연산자는 지원되지 않습니다. 하위 패턴을 정의할 때 괄호를 사용할 수 없습니다.

멀티바이트 문자는 지원되지 않습니다.

> **Note**
>
> 할당량
> 지표 필터 또는 구독 필터를 생성하는 경우 각 로그 그룹에 대한 정규식이 포함된 최대 5개의 필터 패턴이 있습니다.
> 지표 필터 및 구독 필터에 대해 구분된 필터 패턴 또는 JSON 필터 패턴을 생성하거나 로그 이벤트 또는 Live Tail을 필터링하는 경우 각 필터 패턴에 대해 2개의 정규식으로 제한됩니다.

지원되는 연산자 사용

- ^: 문자열의 시작 위치로 일치 항목을 고정합니다. 예를 들어, `%^[hc]at%`는 문자열의 시작 위치만 일치시켜 'hat' 및 'cat'과 일치합니다.
- $: 문자열의 끝 위치로 일치 항목을 고정합니다. 예를 들어, `%[hc]at$%`는 문자열의 끝 위치만 일치시켜 'hat' 및 'cat'과 일치합니다.
- ?: 이전 용어의 0개 이상의 인스턴스와 일치합니다. 예를 들어, `%colou?r%`는 'color' 및 'colour' 모두와 일치할 수 있습니다.
- []: 문자 클래스를 정의합니다. 대괄호 안에 포함된 문자 목록 또는 문자 범위와 일치합니다. 예를 들어, `%[abc]%`는 'a', 'b' 또는 'c'와 일치하고, `%[a-z]%`는 'a'에서 'z'까지의 모든 소문자와 일치하며, `%[abcx-z]%`는 'a', 'b', 'c', 'x', 'y' 또는 'z'와 일치합니다.
- {m, n}: 앞에 오는 용어와 m~n번만큼 일치합니다. 예를 들어, `%a{3,5}%` 는 'aaa', 'aaaa' 및 'aaaaa'와만 일치합니다.

> **Note**
>
> 최솟값 또는 최댓값을 정의하지 않으려는 경우 m 또는 n을 생략할 수 있습니다.

- `|`: 부울 'Or' 연산자로, 세로 막대의 양쪽에 있는 용어 중 하나와 일치합니다. 예를 들어, `%gra|ey%`는 'gray' 또는 'grey'와 일치할 수 있습니다.

  > **Note**
  >
  > 용어는 단일 문자이거나 `?`, `*`, `+` 또는 `{n,m}` 연산자 중 하나를 사용하는 반복되는 문자 클래스입니다.

- `\`: 이스케이프 문자로, 이 문자를 사용하면 연산자의 특수한 의미 대신, 문자 그대로의 의미를 사용할 수 있습니다. 예를 들어, `%\[.\]%`는 대괄호가 이스케이프 처리되었으므로 '[a]', '[b]', '[7]', '[@]', '[]]' 및 '[ ]'와 같이 '[' 및 ']'로 묶인 모든 단일 문자와 일치합니다.

  > **Note**
  >
  > `%10\.10\.0\.1%`는 IP 주소 10.10.0.1과 일치하는 정규식을 생성하는 올바른 방법입니다.

- `*`: 이전 용어의 0개 이상의 인스턴스와 일치합니다. 예를 들어, `%ab*c%` 는 'ac', 'abc' 및 'abbbc'와 일치할 수 있으며, `%ab[0-9]*%`는 'ab', 'ab0' 및 'ab129'와 일치할 수 있습니다.
- `+`: 이전 용어의 하나 이상의 인스턴스와 일치합니다. 예를 들어, `%ab+c%`는 'abc', 'abbc' 및 'abbbc'와 일치할 수 있지만, 'ac'와는 일치하지 않습니다.
- `.`: 모든 문자와 일치합니다. 예를 들어, `%.at%`는 'hat', 'cat', 'bat', '4at', '#at' 및 ' at'(공백으로 시작)을 포함하여 'at'으로 끝나는 3자리 문자열과 일치합니다.

  > **Note**
  >
  > IP 주소와 일치하는 정규식을 생성하는 경우 `.` 연산자를 이스케이프 처리하지 않아야 합니다. 예를 들어, `%10.10.0.1%`는 '10010,051'과 일치할 수 있으며, 이는 식의 의도된 실제 용도가 아닐 수 있습니다.

- `\d`, `\D`: 숫자 및 숫자 외 문자와 일치합니다. 예를 들어, `%\d%`는 `%[0-9]%`와 같고, `%\D%`는 `%[^0-9]%`와 같습니다.

  > **Note**
  >
  > 대문자인 연산자는 소문자인 연산자와 의미가 반대입니다.

- `\s`, `\S`: 공백 문자 및 공백 외 문자와 일치합니다.

> **Note**
>
> 대문자인 연산자는 소문자인 연산자와 의미가 반대입니다. 공백 문자에는 탭(\t), 공백( ), 줄 바꿈(\n) 문자가 포함됩니다.

- \w, \W: 영숫자 및 영숫자 외 문자와 일치합니다. 예를 들어, %\w%는 %[a-zA-Z_0-9]%와 같고, %\W%는 %[^a-zA-Z_0-9]%와 같습니다.

> **Note**
>
> 대문자인 연산자는 소문자인 연산자와 의미가 반대입니다.

- \xhh: 2자리 16진수 문자의 ASCII 매핑을 일치합니다. \x는 다음 문자가 ASCII의 16진수 값임을 나타내는 이스케이프 시퀀스입니다. hh는 ASCII 테이블의 문자를 가리키는 2자리 16진수(0~9 및 A~F)를 지정합니다.

> **Note**
>
> \xhh를 사용하여 필터 패턴에서 지원하지 않는 기호 문자를 일치시킬 수 있습니다. 예를 들어, %\x3A%는 :과 일치하며, %\x28%는 (와 일치합니다.

필터 패턴을 사용하여 정규식과 일치하는 용어 검색

정규식을 사용하여 일치하는 용어 검색

%(정규식 패턴 앞뒤의 백분율 기호)로 묶인 정규식 패턴을 사용하여 로그 이벤트에서 일치하는 용어를 검색할 수 있습니다. 다음 코드 조각은 AUTHORIZED 키워드로 구성된 모든 로그 이벤트를 반환하는 필터 패턴의 예시를 보여줍니다.

지원되는 정규식 목록은 지원되는 정규식을 참조하세요.

```
%AUTHORIZED%
```

이 필터 패턴은 다음과 같은 로그 이벤트 메시지를 반환합니다.

- [ERROR 401] UNAUTHORIZED REQUEST
- [SUCCESS 200] AUTHORIZED REQUEST

필터 패턴을 사용하여 비정형 로그 이벤트에서 일치하는 용어 검색

비정형 로그 이벤트에서 일치하는 용어 검색

다음 예제에는 필터 패턴을 사용하여 비정형 로그 이벤트에서 일치하는 용어를 검색하는 방법을 보여 주는 코드 조각이 포함되어 있습니다.

> **Note**
>
> 필터 패턴은 대/소문자를 구분합니다. 영숫자가 아닌 문자가 포함된 정확한 문구 및 용어를 큰 따옴표("")로 묶습니다.

Example: Match a single term

다음 코드 조각은 메시지에 ERROR라는 단어가 포함된 모든 로그 이벤트를 반환하는 단일 용어 필터 패턴의 예를 보여 줍니다.

```
ERROR
```

이 필터 패턴은 다음과 같이 로그 이벤트 메시지와 일치합니다.

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST
- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

Example: Match multiple terms

다음 코드 조각은 메시지에 ERROR 및 ARGUMENTS라는 단어가 포함된 모든 로그 이벤트를 반환하는 다중 용어 필터 패턴의 예를 보여 줍니다.

```
ERROR ARGUMENTS
```

필터는 다음과 같은 로그 이벤트 메시지를 반환합니다.

- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

이 필터 패턴은 필터 패턴에 지정된 두 용어를 모두 포함하지 않으므로 다음의 로그 이벤트 메시지를 반환하지 않습니다.

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`

Example: Match optional terms

패턴 일치를 사용하여 선택적 용어가 포함된 로그 이벤트를 반환하는 필터 패턴을 생성할 수 있습니다. 일치시키려는 용어 앞에 물음표('?')를 입력합니다. 다음 코드 조각은 메시지에 ERROR라는 단어 또는 ARGUMENTS라는 단어가 포함된 모든 로그 이벤트를 반환하는 필터 패턴의 예를 보여줍니다.

```
?ERROR ?ARGUMENTS
```

이 필터 패턴은 다음과 같이 로그 이벤트 메시지와 일치합니다.

- `[ERROR 400] BAD REQUEST`
- `[ERROR 401] UNAUTHORIZED REQUEST`
- `[ERROR 419] MISSING ARGUMENTS`
- `[ERROR 420] INVALID ARGUMENTS`

> **Note**
>
> 물음표('?')를 포함 및 제외 항 등의 다른 필터 패턴과 결합할 수 없습니다. '?'를 다른 필터 패턴과 결합하면 물음표('?')는 무시됩니다.

예를 들어, 다음 필터 패턴은 REQUEST라는 단어가 포함된 모든 이벤트와 일치하지만 물음표('?') 필터는 무시되고 효과가 없습니다.

```
?ERROR ?ARGUMENTS REQUEST
```

로그 이벤트 일치

- [INFO] REQUEST FAILED
- [WARN] UNAUTHORIZED REQUEST
- [ERROR] 400 BAD REQUEST

Example: Match exact phrases

다음 코드 조각은 메시지에 INTERNAL SERVER ERROR라는 정확한 구가 포함된 모든 로그 이벤트를 반환하는 필터 패턴의 예를 보여 줍니다.

```
"INTERNAL SERVER ERROR"
```

이 필터는 다음과 같은 로그 이벤트 메시지를 반환합니다.

- [ERROR 500] INTERNAL SERVER ERROR

Example: Include and exclude terms

메시지에 일부 용어가 포함되고 다른 용어가 제외된 로그 이벤트를 반환하는 필터 패턴을 만들 수 있습니다. 제외하려는 용어 앞에 빼기 기호('-')를 입력합니다. 다음 코드 조각은 메시지에 ERROR라는 용어를 포함하고 ARGUMENTS라는 용어를 제외한 로그 이벤트를 반환하는 필터 패턴의 예를 보여 줍니다.

```
ERROR -ARGUMENTS
```

이 필터 패턴은 다음과 같은 로그 이벤트 메시지를 반환합니다.

- [ERROR 400] BAD REQUEST
- [ERROR 401] UNAUTHORIZED REQUEST

이 필터 패턴은 단어 ARGUMENTS를 포함하므로 다음 로그 이벤트 메시지를 반환하지 않습니다.

- [ERROR 419] MISSING ARGUMENTS
- [ERROR 420] INVALID ARGUMENTS

Example: Match everything

로그 이벤트의 모든 항목을 큰따옴표로 일치시킬 수 있습니다. 다음 코드 조각은 모든 로그 이벤트를 반환하는 필터 패턴의 예를 보여 줍니다.

```
" "
```

필터 패턴을 사용하여 JSON 로그 이벤트에서 일치하는 용어 검색

JSON 로그 이벤트에 대한 필터 패턴 작성

다음 예제에서는 문자열 및 숫자 값을 포함하는 JSON 용어와 일치하는 필터 패턴에 대한 구문을 작성하는 방법을 설명합니다.

Writing filter patterns that match strings

JSON 로그 이벤트에서 일치하는 문자열을 검색하도록 지표 필터를 생성할 수 있습니다. 다음 코드 조각은 문자열 기반 필터 패턴 구문 예제를 보여줍니다.

```
{ PropertySelector EqualityOperator String }
```

필터 패턴은 중괄호('{}')로 묶습니다. 문자열 기반 필터 패턴은 다음 부분을 포함해야 합니다.

- 속성 선택기

뒤에 마침표가 있는 달러 기호('$.')로 속성 선택기를 설정합니다. 속성 선택기는 하이픈('-') 및 밑줄('_') 문자도 지원하는 영숫자 문자열입니다. 문자열은 과학적 표기법을 지원하지 않습니다. 속성 선택기는 JSON 로그 이벤트의 값 노드를 가리킵니다. 값 노드는 문자열이나 숫자일 수 있습니다. 속성 선택기 뒤에 배열을 배치합니다. 배열의 요소는 0부터 시작하는 번호 매기기 체계를 따릅니다. 즉, 배열의 첫 번째 요소는 요소 0이고, 두 번째 요소는 요소 1이 되는 식입니다. 요소를 대괄호('[]')로 묶습니다. 속성 선택기가 배열이나 객체를 가리키는 경우 필터 패턴은 로그 형식과 일치하지 않습니다. JSON 속성에 마침표("."")가 포함된 경우 대괄호 표기법을 사용하여 해당 속성을 선택할 수 있습니다.

> **Note**
>
> 와일드카드 선택기
> JSON 와일드카드를 사용하여 배열 요소 또는 JSON 객체 필드를 선택할 수 있습니다.
> 할당량
> 속성 선택기에서는 와일드카드 선택기를 최대 1개만 사용할 수 있습니다.

- 같음 연산자

  같음('=') 또는 같지 않음('!=') 기호 중 하나를 사용하여 같음 연산자를 설정합니다. 같음 연산자는 부울 값(true 또는 false)을 반환합니다.

- 문자열

  문자열을 큰따옴표("")로 묶을 수 있습니다. 영숫자 이외의 형식과 밑줄 기호가 포함된 문자열은 큰따옴표로 묶어야 합니다. 별표('*')를 와일드카드로 사용하여 텍스트와 일치시킵니다.

> **Note**
>
> JSON 로그 이벤트에서 일치하는 용어를 검색하기 위해 필터 패턴을 생성하는 경우 모든 조건부 정규식을 사용할 수 있습니다. 지원되는 정규식 목록은 지원되는 정규식을 참조하세요.

다음 코드 조각에는 문자열과 JSON 용어를 일치시키도록 필터 패턴의 서식을 지정하는 방법을 보여주는 필터 패턴 예제가 포함되어 있습니다.

```
{ $.eventType = "UpdateTrail" }
```

Writing filter patterns that match numeric values

JSON 로그 이벤트에서 일치하는 숫자 값을 검색하도록 필터 패턴을 생성할 수 있습니다. 다음 코드 조각은 숫자 값과 일치하는 필터 패턴 구문 예제를 보여줍니다.

```
{ PropertySelector NumericOperator Number }
```

필터 패턴은 중괄호('{}')로 묶습니다. 숫자 값과 일치하는 필터 패턴은 다음 부분을 포함해야 합니다.

- 속성 선택기

  뒤에 마침표가 있는 달러 기호('$.')로 속성 선택기를 설정합니다. 속성 선택기는 하이픈('-') 및 밑줄('_') 문자도 지원하는 영숫자 문자열입니다. 문자열은 과학적 표기법을 지원하지 않습니다. 속성 선택기는 JSON 로그 이벤트의 값 노드를 가리킵니다. 값 노드는 문자열이나 숫자일 수 있습니다. 속성 선택기 뒤에 배열을 배치합니다. 배열의 요소는 0부터 시작하는 번호 매기기 체계를 따릅니다. 즉, 배열의 첫 번째 요소는 요소 0이고, 두 번째 요소는 요소 1이 되는 식입니다. 요소를 대괄호('[]')로 묶습니다. 속성 선택기가 배열이나 객체를 가리키는 경우 필터 패턴은 로그 형식과 일치하지 않습니다. JSON 속성에 마침표('.')가 포함된 경우 대괄호 표기법을 사용하여 해당 속성을 선택할 수 있습니다.

  > **Note**
  >
  > 와일드카드 선택기
  > JSON 와일드카드를 사용하여 배열 요소 또는 JSON 객체 필드를 선택할 수 있습니다.
  > 할당량
  > 속성 선택기에서는 와일드카드 선택기를 최대 1개만 사용할 수 있습니다.

- 숫자 연산자

  다음 기호 중 하나를 사용하여 숫자 연산자를 설정합니다. 보다 큼('>'), 보다 작음('<'), 같음('='), 같지 않음('!='), 이상('>=') 또는 이하('<=') 등.

- 숫자

더하기('+') 또는 빼기('-') 기호를 포함하는 정수를 사용하고 과학적 표기법을 따를 수 있습니다. 별표('*')를 와일드카드로 사용하여 숫자와 일치시킵니다.

다음 코드 조각에는 숫자 값과 JSON 용어를 일치시키도록 필터 패턴의 서식을 지정하는 방법을 보여주는 예제가 포함되어 있습니다.

```
// 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 }
```

간단한 표현식을 사용하여 JSON 로그 이벤트에서 일치하는 용어 검색

다음 예제에는 필터 패턴을 통해 JSON 로그 이벤트에서 일치하는 용어를 검색하는 방법을 보여주는 코드 조각이 포함되어 있습니다.

> **Note**
>
> JSON 로그 이벤트 예제에서 필터 패턴 예제를 테스트하는 경우 JSON 로그 예제를 한 줄에 입력해야 합니다.

JSON 로그 이벤트

```
{
      "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

이 필터 패턴은 "eventType" 속성에서 "UpdateTrail" 문자열과 일치합니다.

```
{ $.eventType = "UpdateTrail" }
```

Example: Filter pattern that matches string values (IP address)

이 필터 패턴은 와일드카드를 포함하며 "123.123." 접두사가 있는 숫자는 포함하지 않으므로 "sourceIPAddress" 속성과 일치합니다.

```
{ $.sourceIPAddress != 123.123.* }
```

Example: Filter pattern that matches a specific array element with a string value

이 필터 패턴은 "arrayKey" 배열에서 "value" 요소와 일치합니다.

```
{ $.arrayKey[0] = "value" }
```

Example: Filter pattern that matches a string using regex

이 필터 패턴은 "eventType" 속성에서 "Trail" 문자열과 일치합니다.

```
{ $.eventType = %Trail% }
```

Example: Filter pattern that uses a wildcard to match values of any element in the array using regex

필터 패턴에는 "arrayKey" 배열에서 "value" 요소와 일치하는 정규식이 포함되어 있습니다.

```
{ $.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)

이 필터 패턴에는 "sourceIPAddress" 속성에서 "111.111.111.111" 요소와 일치하는 정규식이 포함되어 있습니다.

```
{ $.* = %111\.111\.111\.1[0-9]{1,2}% }
```

> **Note**
>
> 할당량
> 속성 선택기에서는 와일드카드 선택기를 최대 1개만 사용할 수 있습니다.

Example: Filter pattern that matches a JSON property with a period (.) in the key

```
{ $.['cluster.name'] = "c" }
```

Example: Filter pattern that matches JSON logs using IS

IS 변수와 JSON 로그의 필드를 일치시키는 필터 패턴을 생성할 수 있습니다. IS 변수는 NULL, TRUE 또는 FALSE 값을 포함하는 필드와 일치할 수 있습니다. 다음 필터 패턴은 SomeObject 값이 NULL인 JSON 로그를 반환합니다.

```
{ $.SomeObject IS NULL }
```

Example: Filter pattern that matches JSON logs using not_exists

NOT EXISTS변수를 사용하여 필터 패턴을 생성하여 로그 데이터에 특정 필드가 포함되지 않은 JSON 로그를 반환할 수 있습니다. 다음 필터 패턴은 SomeOtherObject 필드를 포함하지 않는 JSON 로그를 반환하는 NOT EXISTS를 사용합니다.

```
{ $.SomeOtherObject NOT EXISTS }
```

> **Note**
>
> 변수 IS NOT 및 EXISTS는 현재 지원되지 않습니다.

복합 표현식을 사용하여 JSON 객체에서 일치하는 용어 검색

필터 패턴에서 논리 연산자 AND('&&') 및 OR('||')을 사용하여 둘 이상의 조건이 참인 로그 이벤트와 일치하는 복합 표현식을 생성할 수 있습니다. 복합 표현식은 괄호('()') 사용과 () > && > || 표준 연산 순서를 지원합니다. 다음 예제에는 JSON 객체에서 일치하는 용어를 검색하도록 복합 표현식에서 필터 패턴을 사용하는 방법을 보여주는 코드 조각이 포함되어 있습니다.

JSON 객체

```
{
    "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 (&&)

이 필터 패턴에는 숫자 값이 1인 "user"의 "id"와 일치하고 문자열이 "John.Doe@example.com"인 "users" 배열 첫 번째 요소의 "email"과 일치하는 복합 표현식이 포함되어 있습니다.

```
{ ($.user.id = 1) && ($.users[0].email = "John.Doe@example.com") }
```

Example: Expression that matches using OR (||)

이 필터 패턴에는 문자열이 "John.Stiles@example.com"인 "user"의 "email"과 일치하는 복합 표현식이 포함되어 있습니다.

```
{ $.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch" &&
 $.actions[2] = "nonmatch" }
```

Example: Expression that doesn't match using AND (&&)

이 필터 패턴에는 표현식이 "actions"의 세 번째 작업과 일치하지 않아서 일치 항목을 찾지 못하는 복합 표현식이 포함되어 있습니다.

```
{ ($.user.email = "John.Stiles@example.com" || $.coordinates[0][1] = "nonmatch") &&
 $.actions[2] = "nonmatch" }
```

> **Note**
>
> 할당량
> 속성 선택기에서 와일드카드 선택기를 최대 1개만 사용할 수 있으며, 복합 표현식이 포함된 필터 패턴에서는 와일드카드 선택기를 3개까지 사용할 수 있습니다.

Example: Expression that doesn't match using OR (||)

이 필터 패턴에는 표현식이 "users"의 첫 번째 속성 또는 "actions"의 세 번째 작업과 일치하지 않아서 일치 항목을 찾지 못하는 복합 표현식이 포함되어 있습니다.

```
{ ($.user.id = 2 && $.users[0].email = "nonmatch") || $.actions[2] = "GET" }
```

필터 패턴을 사용하여 공백으로 구분된 로그 이벤트에서 일치하는 용어 검색

공백으로 구분된 로그 이벤트에 대한 필터 패턴 작성

공백으로 구분된 로그 이벤트에서 일치하는 용어를 검색하도록 필터 패턴을 생성할 수 있습니다. 다음에서는 공백으로 구분된 로그 이벤트의 예제를 제공하고, 공백으로 구분된 로그 이벤트에서 일치하는 용어를 검색하는 필터 패턴 구문을 작성하는 방법을 설명합니다.

> **Note**
>
> 공백으로 구분된 로그 이벤트에서 일치하는 용어를 검색하기 위해 필터 패턴을 생성하는 경우 모든 조건부 정규식을 사용할 수 있습니다. 지원되는 정규식 목록은 지원되는 정규식을 참조하세요.

Example: Space-delimited log event

다음 코드 조각은 7개의 필드(`ip`, `user`, `username`, `timestamp`, `request`, `status_code`, `bytes`)를 포함하는 공백으로 구분된 로그 이벤트를 보여 줍니다.

```
127.0.0.1 Prod frank [10/Oct/2000:13:25:15 -0700] "GET /index.html HTTP/1.0" 404
 1534
```

> **Note**
>
> 대괄호('[]') 와 큰따옴표("") 사이의 문자는 단일 필드로 간주됩니다.

Writing filter patterns that match terms in a space-delimited log event

공백으로 구분된 로그 이벤트에서 일치하는 용어를 검색하는 필터 패턴을 생성하려면 필터 패턴을 대괄호('[]')로 묶고 쉼표(',')로 이름이 구분된 필드를 지정합니다. 다음 필터 패턴은 7개의 필드를 구문 분석합니다.

```
[ip=%127\.0\.0\.[1-9]%, user, username, timestamp, request =*.html*, status_code =
 4*, bytes]
```

숫자 연산자(>, <, =, !=, >= 또는 <=) 및 별표(*)를 와일드카드 또는 정규식으로 사용하여 필터 패턴에 조건을 제공할 수 있습니다. 필터 패턴 예제에서 `ip`는 IP 주소 범위 127.0.0.1~127.0.0.9와 일치

하는 정규식을 사용하고, request에는 .html을 포함하는 값을 추출해야 함을 나타내는 와일드카드가 포함되어 있으며, status_code에는 4로 시작하는 값을 추출해야 함을 나타내는 와일드카드가 포함되어 있습니다.

공백으로 구분된 로그 이벤트에서 구문 분석하는 필드 수를 모르는 경우 줄임표(...)를 사용하여 이름이 지정되지 않은 필드를 참조할 수 있습니다. 줄임표는 필요 시 많은 필드를 참조할 수 있습니다. 다음 예제에서는 이전 필터 패턴 예제에 나온 이름이 없는 처음 4개의 필드를 나타내는 줄임표를 포함하는 필터 패턴을 보여줍니다.

```
[..., request =*.html*, status_code = 4*, bytes]
```

논리 연산자 AND(&&) 및 OR(||)를 사용하여 복합 표현식을 만들 수도 있습니다. 다음 필터 패턴에는 status_code 값은 404 또는 410이어야 함을 나타내는 복합 표현식이 포함되어 있습니다.

```
[ip, user, username, timestamp, request =*.html*, status_code = 404 || status_code =
 410, bytes]
```

패턴 일치를 사용하여 공백으로 구분된 로그 이벤트에서 일치하는 용어 검색

패턴 일치를 사용하여 특정 순서로 용어를 일치시키는 공백으로 구분된 필터 패턴을 생성할 수 있습니다. 표시기를 사용하여 용어의 순서를 지정합니다. 첫 번째 용어를 나타낼 때는 w1을 사용하고 후속 용어의 순서를 나타낼 때는 w2 등을 사용합니다. 용어 사이에 쉼표(',')를 입력합니다. 다음 예제에는 공백으로 구분된 필터 패턴으로 패턴 일치를 사용하는 방법을 보여주는 코드 조각이 포함되어 있습니다.

> **Note**
>
> 공백으로 구분된 로그 이벤트에서 일치하는 용어를 검색하기 위해 필터 패턴을 생성하는 경우 모든 조건부 정규식을 사용할 수 있습니다. 지원되는 정규식 목록은 지원되는 정규식을 참조하세요.

공백으로 구분된 로그 이벤트

```
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

다음과 같은 공백으로 구분된 필터 패턴은 로그 이벤트의 첫 번째 단어가 ERROR인 로그 이벤트를 반환합니다.

```
[w1=ERROR, w2]
```

> **Note**
>
> 패턴 일치를 사용하는 공백으로 구분된 지표 필터를 생성하는 경우 용어 순서를 지정한 후 빈 표시기를 포함해야 합니다. 예를 들어, 첫 번째 단어가 ERROR인 로그 이벤트를 반환하는 필터 패턴을 생성하는 경우 w1 용어 뒤에 빈 w2 표시기를 포함합니다.

Example: Match terms with AND (&&) and OR (||)

논리 연산자 AND('&&') 및 OR('||')을 사용하여 조건을 포함하는 공백으로 구분된 필터 패턴을 생성할 수 있습니다. 다음 필터 패턴은 이벤트의 첫 번째 단어가 ERROR 또는 WARNING인 로그 이벤트를 반환합니다.

```
[w1=ERROR || w1=WARNING, w2]
```

Example: Exclude terms from matches

하나 이상의 용어를 제외한 로그 이벤트를 반환하는 공백으로 구분된 필터 패턴을 생성할 수 있습니다. 제외하려는 용어 앞에 같지 않음 기호('!=')를 입력합니다. 다음 코드 조각은 첫 번째 단어가 ERROR 및 WARNING이 아닌 로그 이벤트를 반환하는 필터 패턴 예제를 보여줍니다.

```
[w1!=ERROR && w1!=WARNING, w2]
```

Example: Match the top level item in a resource URI

다음 코드 조각은 정규식을 사용하여 리소스 URI의 상위 항목과 일치하는 필터 패턴 예제를 보여줍니다.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+$%, response_time]
```

Example: Match the child level item in a resource URI

다음 코드 조각은 정규식을 사용하여 리소스 URI의 하위 항목과 일치하는 필터 패턴 예제를 보여줍니다.

```
[logLevel, date, time, method, url=%/service/resource/[0-9]+/part/[0-9]+$%,
 response_time]
```

AWS 서비스에서 로깅 활성화

많은 서비스가 로그에만 로그를 게시하지만 일부 AWS 서비스는 Amazon Simple Storage Service 또는 Amazon Data Firehose에 직접 로그를 게시할 수 있습니다. CloudWatch 로그의 주요 요구 사항이 이러한 서비스 중 하나에서의 저장 또는 처리인 경우, 로그를 생성하는 서비스에서 추가 설정 없이 손쉽게 로그를 Amazon S3 또는 Firehose로 직접 전송하도록 할 수 있습니다.

로그가 Amazon S3 또는 Firehose에 직접 게시되는 경우에도 요금이 부과됩니다. 자세한 내용은 Amazon CloudWatch Pricing의 로그 탭에 있는 벤디드 로그를 참조하십시오.

일부 AWS 서비스는 공통 인프라를 사용하여 로그를 전송합니다. 서비스에서 로깅을 활성화하려면 특정 권한을 가진 사용자로 로그인해야 합니다. 또한 로그를 전송할 수 있는 AWS 권한을 부여해야 합니다.

이러한 권한이 필요한 서비스의 경우 두 가지 버전의 권한이 필요합니다. 이러한 추가 권한이 필요한 서비스는 표에 지원되는 [V1 권한]과 지원되는 [V2 권한]으로 표시되어 있습니다. 이러한 필수 권한에 대한 자세한 내용은 표 다음 섹션을 참조하세요.

로그 유형	CloudWatch Logs	Amazon S3	Firehose
Amazon API Gateway 액세스 로그	지원되는 [V1 권한]		
AWS AppSync 로그	지원		
Amazon Aurora MySQL 로그	지원		
Amazon Bedrock 지식 기반, 로깅	지원되는 [V2 권한]	지원되는 [V2 권한]	지원되는 [V2 권한]
Amazon Chime 미디어 품질 지표 로그 및 SIP 메시지 로그	지원되는 [V1 권한]		
CloudFront: 액세스 로그		지원되는 [V1 권한]	
AWS CloudHSM 감사 로그	지원		

로그 유형	CloudWatch Logs	Amazon S3	Firehose
CloudWatch 분명히 평가 이벤트 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	
CloudWatch 인터넷 모니터 로그		지원되는 [V1 권한]	
CloudTrail 로그	지원		
AWS CodeBuild 로그	지원		
Amazon CodeWhisperer 이벤트 로그	지원되는 [V2 권한]	지원되는 [V2 권한]	지원되는 [V2 권한]
Amazon Cognito 로그	지원되는 [V1 권한]		
Amazon Connect 로그	지원		
AWS DataSync 로그	지원		
아마존 포 ElastiCache 레디스용 로그	지원되는 [V1 권한]		지원되는 [V1 권한]
AWS Elastic Beanstalk 로그	지원		
Amazon Elastic Container Service 로그	지원		
Amazon Elastic Kubernetes Service 컨트롤 플레인 로그	지원		
Amazon EventBridge 파이프 로깅	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS Fargate 로그	지원		
AWS Fault Injection Service 실험 로그		지원되는 [V1 권한]	

로그 유형	CloudWatch Logs	Amazon S3	Firehose
Amazon FinSpace	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS Global Accelerator 플로우 로그		지원되는 [V1 권한]	
AWS Glue 작업 로그	지원		
IAM ID 센터 오류 로그	지원되는 [V2 권한]	지원되는 [V2 권한]	지원되는 [V2 권한]
Amazon Interactive Video Service 채팅 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS IoT 로그	지원		
AWS IoT FleetWise 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS Lambda 로그	지원		
Amazon Macie 로그	지원		
AWS Mainframe Modernization	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon Managed Service for Prometheus 로그	지원되는 [V1 권한]		
Amazon MSK 브로커 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon MSK Connect 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon MQ 일반 및 감사 로그	지원		

로그 유형	CloudWatch Logs	Amazon S3	Firehose
AWS Network Firewall 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Network Load Balancer 액세스 로그		지원되는 [V1 권한]	
OpenSearch 로그	지원		
Amazon OpenSearch 서비스 통합 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS OpsWorks 로그	지원		
Amazon 관계형 데이터베이스 ServicePostgreSQL 로그	지원		
AWS RoboMaker 로그	지원		
Amazon Route 53 퍼블릭 DNS 쿼리 로그	지원		
Amazon Route 53 Resolver 쿼리 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	
아마존 SageMaker 이벤트	지원되는 [V1 권한]		
아마존 SageMaker 워커 이벤트	지원되는 [V1 권한]		
AWS 사이트 간 VPN 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon Simple Notification Service 로그	지원		
Amazon Simple Notification Service 데이터 보호 정책 로그	지원		

로그 유형	CloudWatch Logs	Amazon S3	Firehose
EC2 스팟 인스턴스 데이터 피드 파일		지원되는 [V1 권한]	
AWS Step Functions 익스프레스 워크플로우 및 표준 워크플로우 로그	지원되는 [V1 권한]		
Storage Gateway 감사 로그 및 상태 로그	지원되는 [V1 권한]		
AWS Transfer Family 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS Verified Access 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon Virtual Private Cloud 흐름 로그	지원	지원되는 [V1 권한]	지원되는 [V1 권한]
Amazon VPC Lattice 액세스 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원되는 [V1 권한]
AWS WAF 로그	지원되는 [V1 권한]	지원되는 [V1 권한]	지원
Amazon WorkMail 로그	지원되는 [V2 권한]	지원되는 [V2 권한]	지원되는 [V2 권한]

추가 권한 [V1]이 필요한 로깅

일부 AWS 서비스는 공통 인프라를 사용하여 로그를 로그, Amazon S3 또는 Firehose로 CloudWatch 전송합니다. 다음 표에 나열된 AWS 서비스에서 로그를 대상으로 전송하도록 하려면 특정 권한이 있는 사용자로 로그인해야 합니다.

또한 로그를 전송할 수 있는 권한을 AWS 부여해야 합니다. AWS 로그를 설정할 때 해당 권한을 자동으로 생성할 수도 있고, 로깅을 설정하기 전에 먼저 권한을 직접 만들 수도 있습니다. 계정 간 전송의 경우 권한 정책을 직접 수동으로 만들어야 합니다.

자신이나 조직의 누군가가 로그 전송을 처음 설정할 때 필요한 권한 및 리소스 정책을 AWS 자동으로 설정하도록 선택한 경우, 로그 전송을 설정하는 사용자는 이 섹션의 뒷부분에서 설명하는 것처럼 특정 권한을 가지고 있어야 합니다. 또는 리소스 정책을 직접 만들면 로그 전송을 설정한 사용자에게 많은 권한이 필요하지 않습니다.

다음 표에서는 이 섹션의 정보가 적용되는 로그 유형과 로그 대상을 요약합니다.

다음 섹션에서는 이러한 각 대상에 대한 자세한 내용을 제공합니다.

로그로 전송된 CloudWatch 로그

> **Important**
>
> 다음 목록의 로그 유형을 로그로 전송하도록 설정하면 필요한 경우 CloudWatch 로그를 수신하는 로그 그룹과 관련된 리소스 정책을 AWS 만들거나 변경합니다. 자세한 내용을 살펴보려면 이 섹션을 계속 읽으세요.

이 섹션은 이전 섹션의 표에 나열된 로그 유형이 로그로 전송될 CloudWatch 때 적용됩니다.

사용자 권한

이러한 유형의 로그를 처음으로 Logs로 전송하도록 CloudWatch 설정하려면 다음 권한을 가진 계정으로 로그인해야 합니다.

- `logs:CreateLogDelivery`
- `logs:PutResourcePolicy`
- `logs:DescribeResourcePolicies`
- `logs:DescribeLogGroups`

> **Note**
>
> `logs:DescribeLogGroupslogs:DescribeResourcePolicies`, 또는 `logs:PutResourcePolicy` 권한을 지정할 때는 단일 로그 그룹 이름만 지정하는 대신 해당 `Resource` 행의 ARN이 `*` 와일드카드를 사용하도록 설정해야 합니다. 예제: `"Resource": "arn:aws:logs:us-east-1:111122223333:log-group:*"`

이러한 유형의 로그가 이미 Logs의 로그 그룹으로 전송되고 있는 경우 이러한 유형의 다른 CloudWatch 로그를 동일한 로그 그룹에 전송하도록 설정하려면 해당 권한만 있으면 됩니다. `logs:CreateLogDelivery`

로그 그룹 리소스 정책

로그를 보내는 로그 그룹에는 특정 권한이 포함된 리소스 정책이 있어야 합니다. 로그 그룹에 현재 리소스 정책이 없고 로깅을 설정하는 사용자에게 로그 그룹에 대한 `logs:PutResourcePolicy`, `logs:DescribeResourcePolicies`, 및 `logs:DescribeLogGroups` 권한이 있는 경우 로그를 Logs로 CloudWatch 보내기 시작할 때 다음 정책이 AWS 자동으로 생성됩니다.

```
{
  "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-1:0123456789:log-group:my-log-group:log-stream:*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": ["0123456789"]
        },
        "ArnLike": {
          "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:*"]
        }
      }
    }
  ]
}
```

로그 그룹에 리소스 정책이 있지만 해당 정책에 이전 정책에 표시된 문이 포함되어 있지 않고 로깅을 설정하는 사용자에게 로그 그룹에 대한 `logs:PutResourcePolicy`, `logs:DescribeResourcePolicies` 및 `logs:DescribeLogGroups` 권한이 있는 경우 해당 문이 로그 그룹의 리소스 정책에 추가됩니다.

로그 그룹 리소스 정책 크기 제한 고려 사항

이러한 서비스는 로그를 보내는 각 로그 그룹을 리소스 정책에 나열해야 하며 CloudWatch 로그 리소스 정책은 5120자로 제한됩니다. 다수의 로그 그룹에 로그를 보내는 서비스는 이 한도에 도달할 수 있습니다.

이를 완화하기 위해 Logs는 CloudWatch 로그를 보내는 서비스에서 사용하는 리소스 정책의 크기를 모니터링하여 정책의 크기 제한인 5120자에 근접하는 것이 감지되면 CloudWatch Logs는 해당 서비스의 리소스 정책을 자동으로 `/aws/vendedlogs/*` 활성화합니다. 그런 다음 `/aws/vendedlogs/`로 시작하는 이름을 가진 로그 그룹을 이러한 서비스의 로그 대상으로 사용하기 시작할 수 있습니다.

Amazon S3 로 보낸 로그

Amazon S3로 전송할 로그를 설정하면 필요한 경우 로그를 수신하는 S3 버킷과 관련된 리소스 정책을 AWS 생성하거나 변경합니다.

Amazon S3에 게시되는 로그는 사용자가 지정한 기존 버킷에 게시됩니다. 지정된 버킷에 5분마다 하나 이상의 로그 파일이 생성됩니다.

Amazon S3 버킷에 처음으로 로그를 전송할 때 로그를 전송하는 서비스는 버킷의 소유자를 기록하여 로그가 이 계정에 속한 버킷에만 전달되도록 합니다. 따라서 Amazon S3 버킷 소유자를 변경하려면 원본 서비스에서 로그 구독을 다시 생성하거나 업데이트해야 합니다.

> **Note**
>
> CloudFront 벤드 로그를 S3에 보내는 다른 서비스와는 다른 권한 모델을 사용합니다. 자세한 내용은 표준 로깅 구성 및 로그 파일에 액세스에 필요한 권한을 참조하세요.
> 또한 CloudFront 액세스 로그에 동일한 S3 버킷과 다른 로그 소스를 사용하는 경우, 버킷에서 ACL을 활성화하면 이 버킷을 사용하는 다른 모든 로그 CloudFront 소스에도 권한이 부여됩니다.

사용자 권한

처음으로, 이러한 유형의 로그를 Amazon S3로 전송하도록 설정하려면 다음 권한이 있는 계정에 로그인해야 합니다.

- `logs:CreateLogDelivery`
- `S3:GetBucketPolicy`
- `S3:PutBucketPolicy`

이러한 유형의 로그가 이미 Amazon S3 버킷으로 전송되고 있는 경우 해당 유형 로그 중 다른 하나를 동일한 버킷으로 보내도록 설정하려면 `logs:CreateLogDelivery` 권한만 있으면 됩니다.

S3 버킷 리소스 정책

로그가 전송되는 S3 버킷에는 특정 권한을 포함하는 리소스 정책이 있어야 합니다. 현재 버킷에 리소스 정책이 없고 로깅을 설정하는 사용자에게 버킷에 대한 `S3:GetBucketPolicy` 및 `S3:PutBucketPolicy` 권한이 있는 경우 Amazon S3로 로그를 보내기 시작하면 해당 버킷에 대한 다음 정책을 AWS 자동으로 생성합니다.

```
{
    "Version": "2012-10-17",
    "Id": "AWSLogDeliveryWrite20150319",
    "Statement": [
        {
            "Sid": "AWSLogDeliveryAclCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
                },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::my-bucket",
            "Condition": {
                "StringEquals": {
                "aws:SourceAccount": ["0123456789"]
                },
                "ArnLike": {
                "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:*"]
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-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-1:0123456789:*"]
                }
            }
        }
    ]
}
```

이전 정책에서 aws:SourceAccount에 대해 이 버킷으로 로그를 전달할 계정 IDS의 목록을 지정합니다. aws:SourceArn에 대해 로그를 생성하는 리소스의 ARN 목록을 arn:aws:logs:*source-region*:*source-account-id*:* 형식으로 지정합니다.

버킷에 리소스 정책이 있지만 해당 정책에 이전 정책에 표시된 문이 포함되어 있지 않고 로깅을 설정하는 사용자에게 버킷에 대한 S3:GetBucketPolicy 및 S3:PutBucketPolicy 권한이 있는 경우 해당 문이 로그 그룹의 리소스 정책에 추가됩니다.

> **Note**
>
> s3:ListBucket권한이 부여되지 않은 AWS CloudTrail 경우 경우에 따라 AccessDenied 오류가 발생할 수 delivery.logs.amazonaws.com 있습니다. CloudTrail 로그에서 이러한 오류를 방지하려면 s3:ListBucket 권한을 delivery.logs.amazonaws.com 부여하고 이전 버킷 정책에 설정된 s3:GetBucketAcl 권한과 함께 표시된 Condition 파라미터를 포함해야 합니다. 이 작업을 더 간단하게 하려면 새로운 Statement을(를) 만드는 대신 AWSLogDeliveryAclCheck을(를) “Action”: [“s3:GetBucketAcl”, “s3:ListBucket”](으)로 직접 업데이트할 수 있습니다.

Amazon S3 버킷 서버 측 암호화

Amazon S3 관리 키를 사용한 서버 측 암호화 (SSE-S3) 또는 키가 저장된 서버 측 암호화 (SSE-KMS)를 활성화하여 Amazon S3 버킷의 데이터를 보호할 수 있습니다. AWS KMS AWS Key Management Service 자세한 내용은 서버 측 암호화를 사용하여 데이터 보호를 참조하세요.

SSE-S3를 선택하면 추가 구성이 필요하지 않습니다. Amazon S3는 암호화 키를 처리합니다.

> **Warning**
>
> SSE-KMS를 선택하는 경우 고객 관리 키를 사용해야 합니다. 이 시나리오에서는 관리형 키 사용이 지원되지 않기 때문입니다. AWS AWS 관리 키를 사용하여 암호화를 설정하는 경우 로그는 읽을 수 없는 형식으로 전달됩니다.

고객 관리 AWS KMS 키를 사용하는 경우 버킷 암호화를 활성화할 때 고객 관리 키의 Amazon 리소스 이름 (ARN) 을 지정할 수 있습니다. 로그 전달 계정이 S3 버킷에 쓸 수 있으려면 다음 사항을 S3 버킷의 버킷 정책이 아니라 고객 관리형 키의 키 정책에 추가해야 합니다.

SSE-KMS를 선택하는 경우 고객 관리 키를 사용해야 합니다. 이 시나리오에서는 AWS 관리 키 사용이 지원되지 않기 때문입니다. 고객 관리 AWS KMS 키를 사용하는 경우 버킷 암호화를 활성화할 때 고객 관리 키의 Amazon 리소스 이름 (ARN) 을 지정할 수 있습니다. 로그 전달 계정이 S3 버킷에 쓸 수 있으려면 다음 사항을 S3 버킷의 버킷 정책이 아니라 고객 관리형 키의 키 정책에 추가해야 합니다.

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

aws:SourceAccount에 대해 이 버킷으로 로그를 전달할 계정 IDS의 목록을 지정합니다. aws:SourceArn에 대해 로그를 생성하는 리소스의 ARN 목록을 arn:aws:logs:*source-region*:*source-account-id*:* 형식으로 지정합니다.

Firehose로 전송된 로그

이 섹션은 이전 섹션의 표에 나열된 로그 유형이 Firehose로 전송될 때 적용됩니다.

사용자 권한

이러한 유형의 로그를 Firehose로 처음으로 전송하도록 설정하려면 다음 권한을 가진 계정으로 로그인해야 합니다.

- logs:CreateLogDelivery
- firehose:TagDeliveryStream
- iam:CreateServiceLinkedRole

이러한 유형의 로그가 이미 Firehose로 전송되고 있는 경우 이러한 유형의 다른 로그를 Firehose로 보내도록 설정하려면 및 권한만 있으면 됩니다. logs:CreateLogDelivery firehose:TagDeliveryStream

권한에 사용되는 IAM 역할

Firehose는 리소스 정책을 사용하지 않으므로 이러한 로그를 Firehose로 전송하도록 설정할 때 IAM 역할을 AWS 사용합니다. AWS 라는 서비스 연결 역할을 생성합니다. AWSServiceRoleForLogDelivery 이 서비스 연결 역할에는 다음 권한이 포함됩니다.

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

```
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

이 서비스 연결 역할은 태그가 로 설정된 모든 Firehose 전송 스트림에 권한을 부여합니다.LogDeliveryEnabled. true AWS 로깅을 설정할 때 대상 전송 스트림에 이 태그를 제공합니다.

또한 이 서비스 연결 역할에는 delivery.logs.amazonaws.com 서비스 보안 주체가 필요한 서비스 연결 역할을 맡도록 허용하는 신뢰 정책이 있습니다. 이러한 신뢰 정책은 다음과 같습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

추가 권한 [V2]가 필요한 로깅

일부 AWS 서비스는 새로운 방법을 사용하여 로그를 전송합니다. 이는 이러한 서비스에서 CloudWatch Logs, Amazon S3 또는 Firehose와 같은 하나 이상의 대상으로 로그를 전송하도록 설정할 수 있는 유연한 방법입니다.

작업 로그 전송은 세 가지 요소로 구성됩니다.

- A는 `DeliverySource` 실제로 로그를 보내는 리소스를 나타내는 논리적 객체입니다.
- A는 `DeliveryDestination` 실제 전송 목적지를 나타내는 논리적 객체입니다.
- A`Delivery`: 배송 소스를 배송지에 연결합니다.

지원되는 AWS 서비스와 대상 간의 로그 전달을 구성하려면 다음을 수행해야 합니다.

- 를 사용하여 전송 소스를 생성합니다 PutDeliverySource.
- 를 사용하여 배송지를 생성합니다 PutDeliveryDestination.
- 계정 간에 로그를 전송하는 경우 대상 PutDeliveryDestinationPolicy계정에서 사용하여 대상에 IAM 정책을 할당해야 합니다. 이 정책은 계정 A의 전송 소스에서 계정 B의 전송 목적지까지 전송을 생성할 수 있는 권한을 부여합니다. 계정 간 전송의 경우 권한 정책을 직접 수동으로 만들어야 합니다.
- 를 사용하여 정확히 하나의 배송 소스와 하나의 배송지를 페어링하여 배송을 생성하십시오. CreateDelivery

다음 섹션에서는 V2 프로세스를 사용하여 각 유형의 대상에 대한 로그 전달을 설정하기 위해 로그인할 때 필요한 권한의 세부 정보를 제공합니다. 이러한 권한은 로그인한 IAM 역할에 부여할 수 있습니다.

> **Important**
>
> 로그 생성 리소스를 삭제한 후 로그 전송 리소스를 제거하는 것은 사용자의 책임입니다. 이렇게 하려면 다음 단계를 따르세요.
>
> 1. DeleteDelivery작업을 `Delivery` 사용하여 삭제합니다.
> 2. DeleteDeliverySource작업을 `DeliverySource` 사용하여 삭제합니다.
> 3. 방금 삭제한 `DeliveryDestination` `DeliverySource` 항목과 관련된 항목이 이 특정 `DeliverySource` 항목에만 사용되는 경우 DeleteDeliveryDestinations작업을 사용하여 제거할 수 있습니다.

목차

- 콘솔별 권한

로그로 전송된 CloudWatch 로그

사용자 권한

로그로 로그를 보낼 수 있게 하려면 다음 권한으로 로그인해야 합니다. CloudWatch

```
{
    "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"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:delivery:*",
                "arn:aws:logs:region:account-id:delivery-source:*",
                "arn:aws:logs:region:account-id:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries"
            ],
            "Resource": "*"
```

```
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyCWL",
            "Effect": "Allow",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:*"
            ]
        }
    ]
}
```

로그 그룹 리소스 정책

로그를 보내는 로그 그룹에는 특정 권한이 포함된 리소스 정책이 있어야 합니다. 로그 그룹에 현재 리소스 정책이 없고 로깅을 설정하는 사용자에게 로그 그룹에 대한`logs:PutResourcePolicy`,`logs:DescribeResourcePolicies`, 및 `logs:DescribeLogGroups` 권한이 있는 경우 로그를 Logs로 CloudWatch 보내기 시작하면 다음과 같은 정책이 AWS 자동으로 생성됩니다.

```
{
  "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-1:0123456789:log-group:my-log-group:log-stream:*"
      ],
      "Condition": {
```

```
        "StringEquals": {
          "aws:SourceAccount": ["0123456789"]
        },
        "ArnLike": {
          "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:*"]
        }
      }
    }
  ]
}
```

로그 그룹 리소스 정책 크기 제한 고려 사항

이러한 서비스는 로그를 보내는 각 로그 그룹을 리소스 정책에 나열해야 하며 CloudWatch 로그 리소스 정책은 5120자로 제한됩니다. 많은 수의 로그 그룹에 로그를 전송하는 서비스가 이 제한에 도달할 수 있습니다.

이를 완화하기 위해 Logs는 CloudWatch 로그를 보내는 서비스에서 사용하는 리소스 정책의 크기를 모니터링하여 정책이 크기 제한인 5120자에 근접하는 것으로 감지되면 CloudWatch Logs는 해당 서비스의 리소스 `/aws/vendedlogs/*` 정책에서 자동으로 활성화합니다. 그런 다음 `/aws/vendedlogs/`로 시작하는 이름을 가진 로그 그룹을 이러한 서비스의 로그 대상으로 사용하기 시작할 수 있습니다.

Amazon S3 로 보낸 로그

사용자 권한

Amazon S3로 로그를 전송하려면 다음 권한으로 로그인해야 합니다.

```
{
    "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"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:delivery:*",
                "arn:aws:logs:region:account-id:delivery-source:*",
                "arn:aws:logs:region:account-id:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyS3",
            "Effect": "Allow",
            "Action": [
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
            "Resource": "arn:aws:s3:::bucket-name"
        }
    ]
}
```

로그가 전송되는 S3 버킷에는 특정 권한을 포함하는 리소스 정책이 있어야 합니다. 현재 버킷에 리소스 정책이 없고 로깅을 설정하는 사용자에게 버킷에 대한 `S3:GetBucketPolicy` 및 `S3:PutBucketPolicy` 권한이 있는 경우 Amazon S3로 로그를 보내기 시작하면 해당 버킷에 대한 다음 정책을 AWS 자동으로 생성합니다.

```
{
    "Version": "2012-10-17",
    "Id": "AWSLogDeliveryWrite20150319",
```

```
    "Statement": [
        {
            "Sid": "AWSLogDeliveryAclCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
                },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::my-bucket",
            "Condition": {
                "StringEquals": {
                "aws:SourceAccount": ["0123456789"]
                },
                "ArnLike": {
                "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:delivery-source*"]
                }
            }
        },
        {
            "Sid": "AWSLogDeliveryWrite",
            "Effect": "Allow",
            "Principal": {
                "Service": "delivery.logs.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-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-1:0123456789:delivery-
source:*"]
                }
            }
        }
    ]
}
```

이전 정책에서 aws:SourceAccount에 대해 이 버킷으로 로그를 전달할 계정 IDS의 목록을 지정합니다. aws:SourceArn에 대해 로그를 생성하는 리소스의 ARN 목록을 arn:aws:logs:*source-region*:*source-account-id*:* 형식으로 지정합니다.

버킷에 리소스 정책이 있지만 해당 정책에 이전 정책에 표시된 문이 포함되어 있지 않고 로깅을 설정하는 사용자에게 버킷에 대한 `S3:GetBucketPolicy` 및 `S3:PutBucketPolicy` 권한이 있는 경우 해당 문이 로그 그룹의 리소스 정책에 추가됩니다.

> **Note**
>
> `s3:ListBucket`권한이 부여되지 않은 AWS CloudTrail 경우 경우에 따라 `AccessDenied` 오류가 발생할 수 `delivery.logs.amazonaws.com` 있습니다. CloudTrail 로그에서 이러한 오류를 방지하려면 `s3:ListBucket` 권한을 `delivery.logs.amazonaws.com` 부여하고 이전 버킷 정책에 설정된 `s3:GetBucketAcl` 권한과 함께 표시된 `Condition` 파라미터를 포함해야 합니다. 이 작업을 더 간단하게 하려면 새로운 `Statement`을(를) 만드는 대신 `AWSLogDeliveryAclCheck`을(를) `“Action”: [“s3:GetBucketAcl”, “s3:ListBucket”]`(으)로 직접 업데이트할 수 있습니다.

Amazon S3 버킷 서버 측 암호화

Amazon S3 관리 키를 사용한 서버 측 암호화 (SSE-S3) 또는 키가 저장된 서버 측 암호화 (SSE-KMS) 를 활성화하여 Amazon S3 버킷의 데이터를 보호할 수 있습니다. AWS KMS AWS Key Management Service 자세한 내용은 서버 측 암호화를 사용하여 데이터 보호를 참조하세요.

SSE-S3를 선택하면 추가 구성이 필요하지 않습니다. Amazon S3는 암호화 키를 처리합니다.

> **Warning**
>
> SSE-KMS를 선택하는 경우 고객 관리 키를 사용해야 합니다. 이 시나리오에서는 관리형 키 사용이 지원되지 않기 때문입니다. AWS AWS 관리 키를 사용하여 암호화를 설정하는 경우 로그는 읽을 수 없는 형식으로 전달됩니다.

고객 관리 AWS KMS 키를 사용하는 경우 버킷 암호화를 활성화할 때 고객 관리 키의 Amazon 리소스 이름 (ARN) 을 지정할 수 있습니다. 로그 전달 계정이 S3 버킷에 쓸 수 있으려면 다음 사항을 S3 버킷의 버킷 정책이 아니라 고객 관리형 키의 키 정책에 추가해야 합니다.

SSE-KMS를 선택하는 경우 고객 관리 키를 사용해야 합니다. 이 시나리오에서는 AWS 관리 키 사용이 지원되지 않기 때문입니다. 고객 관리 AWS KMS 키를 사용하는 경우 버킷 암호화를 활성화할 때 고객 관리 키의 Amazon 리소스 이름 (ARN) 을 지정할 수 있습니다. 로그 전달 계정이 S3 버킷에 쓸 수 있으려면 다음 사항을 S3 버킷의 버킷 정책이 아니라 고객 관리형 키의 키 정책에 추가해야 합니다.

```
{
    "Sid": "Allow Logs Delivery to use the key",
    "Effect": "Allow",
    "Principal": {
        "Service": [ "delivery.logs.amazonaws.com" ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": ["0123456789"]
        },
        "ArnLike": {
            "aws:SourceArn": ["arn:aws:logs:us-east-1:0123456789:delivery-source:*"]
        }
        }
}
```

aws:SourceAccount에 대해 이 버킷으로 로그를 전달할 계정 IDS의 목록을 지정합니다. aws:SourceArn에 대해 로그를 생성하는 리소스의 ARN 목록을 arn:aws:logs:*source-region*:*source-account-id*:* 형식으로 지정합니다.

Firehose로 전송된 로그

사용자 권한

Firehose에 로그를 보낼 수 있게 하려면 다음 권한으로 로그인해야 합니다.

```
{
    "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"
            ],
            "Resource": [
                "arn:aws:logs:region:account-id:delivery:*",
                "arn:aws:logs:region:account-id:delivery-source:*",
                "arn:aws:logs:region:account-id:delivery-destination:*"
            ]
        },
        {
            "Sid": "ListAccessForLogDeliveryActions",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeDeliveryDestinations",
                "logs:DescribeDeliverySources",
                "logs:DescribeDeliveries"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUpdatesToResourcePolicyFH",
            "Effect": "Allow",
            "Action": [
                "firehose:TagDeliveryStream"
            ],
            "Resource": [
                "arn:aws:firehose:region:account-id:deliverystream/*"
            ]
        },
        {
            "Sid": "CreateServiceLinkedRole",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
```

```
            "Resource": "arn:aws:iam::account-id:role/aws-service-role/
delivery.logs.amazonaws.com/AWSServiceRoleForLogDelivery"
        }
    ]
}
```

리소스 권한에 사용되는 IAM 역할

Firehose는 리소스 정책을 사용하지 않으므로 이러한 로그를 Firehose로 전송하도록 설정할 때 IAM 역할을 AWS 사용합니다. AWS 라는 서비스 연결 역할을 생성합니다. AWSServiceRoleForLogDelivery 이 서비스 연결 역할에는 다음 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch",
                "firehose:ListTagsForDeliveryStream"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/LogDeliveryEnabled": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

이 서비스 연결 역할은 태그가 로 설정된 모든 Firehose 전송 스트림에 권한을 부여합니다`LogDeliveryEnabled`. `true` AWS 로깅을 설정할 때 대상 전송 스트림에 이 태그를 제공합니다.

또한 이 서비스 연결 역할에는 `delivery.logs.amazonaws.com` 서비스 보안 주체가 필요한 서비스 연결 역할을 맡도록 허용하는 신뢰 정책이 있습니다. 이러한 신뢰 정책은 다음과 같습니다.

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

```
      "Effect": "Allow",
      "Principal": {
        "Service": "delivery.logs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

서비스별 권한

이전 섹션에 나열된 대상별 권한 외에도 일부 서비스에서는 추가 보안 계층으로 고객이 리소스에서 로그를 전송할 수 있도록 허용하는 명시적 승인이 필요합니다. 해당 서비스 내에서 로그를 판매하는 리소스에 대한 AllowVendedLogDeliveryForResource 작업을 승인합니다. 이러한 서비스의 경우 다음 정책을 사용하고 *service* 및 *resource-type#* 적절한 값으로 대체하십시오. 이러한 필드의 서비스별 값은 해당 서비스의 공급업체 로그 설명서 페이지를 참조하십시오.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceLevelAccessForLogDelivery",
            "Effect": "Allow",
            "Action": [
                "service:AllowVendedLogDeliveryForResource"
            ],
            "Resource": "arn:aws:service:region:account-id:resource-type/*"
        }
    ]
}
```

콘솔별 권한

API 대신 콘솔을 사용하여 로그 전달을 설정하는 경우 이전 섹션에 나열된 권한 외에도 다음과 같은 추가 권한이 필요합니다.

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

```
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "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": [
                "*"
            ]
        }
    ]
}
```

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS크로스 서비스 사칭으로 인해 대리인 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(호출하는 서비스)가 다른 서비스(호출되는 서비스)를 호출할 때 발생할 수 있습니다. 직접적으로 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계

정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

CloudWatch Logs가 리소스에 다른 서비스에 부여하는 권한을 제한하려면 리소스 정책에 aws:SourceArnaws:SourceAccountaws:SourceOrgID,,, aws:SourceOrgPaths글로벌 조건 컨텍스트 키를 사용하는 것이 좋습니다. aws:SourceArn을 사용하면 하나의 리소스만 교차 서비스 액세스 권한과 연결됩니다. aws:SourceAccount를 사용하면 해당 계정의 모든 리소스가 교차 서비스 사용 권한과 연결됩니다. aws:SourceOrgID를 사용하면 조직 내 모든 계정의 모든 리소스가 교차 서비스 사용 권한과 연결될 수 있습니다. aws:SourceOrgPaths를 사용하면 AWS Organizations 경로 내 계정의 모든 리소스가 교차 서비스 사용 권한과 연결됩니다. 경로 사용 및 이해에 대한 자세한 내용은 AWS Organizations 엔티티 경로 이해를 참조하십시오.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 arn:aws:*servicename*:*:*123456789012*:*입니다.

만약 aws:SourceArn 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우, 권한을 제한하려면 두 aws:SourceAccount 및 aws:SourceArn를 모두 사용해야 합니다.

혼동된 대리자 문제로부터 보호하려면 리소스 기반 정책에서 리소스의 조직 ID 또는 조직 경로와 함께 aws:SourceOrgID 또는 aws:SourceOrgPaths 전역 조건 컨텍스트 키를 사용하세요. aws:SourceOrgID 또는 aws:SourceOrgPaths 키가 포함된 정책에는 올바른 계정이 자동으로 포함되며 조직에서 계정을 추가, 제거 또는 이동할 때 정책을 수동으로 업데이트할 필요가 없습니다.

이 페이지의 이전 섹션에 있는 정책은 aws:SourceArn 및 aws:SourceAccount 글로벌 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

CloudWatch AWS 관리형 정책에 대한 업데이트를 기록합니다.

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후 CloudWatch Logs의 AWS 관리형 정책 업데이트에 대한 세부 정보를 볼 수 있습니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 CloudWatch 로그 문서 기록 페이지에서 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
AWSServiceRoleForLogDelivery 서비스 연결 역할 정책 — 기존 정책 업데이트	CloudWatch 로그로 인해 서비스 연결 역할과 관련된 IAM 정책의 권한이 변경되었습니다. AWSServiceRoleForLogDelivery 변경 내용은 다음과 같습니다. • `firehose:ResourceTag/LogDeliveryEnabled": "true"` 조건 키가 `aws:ResourceTag/LogDeliveryEnabled": "true"`로 변경되었습니다.	2021년 7월 15일
CloudWatch 로그에서 변경 사항을 추적하기 시작했습니다.	CloudWatch 로그에서 AWS 관리형 정책의 변경 사항을 추적하기 시작했습니다.	2021년 6월 10일

Amazon S3로 로그 데이터 내보내기

로그 그룹에서 Amazon S3 버킷으로 로그 데이터를 내보내고 이 데이터를 사용자 지정 처리 및 분석에 사용하거나 다른 시스템에 로드합니다. 동일한 계정 또는 다른 계정에 있는 버킷으로 내보낼 수 있습니다.

다음을 수행할 수 있습니다.

- () 에서 SSE-KMS로 암호화된 S3 버킷으로 로그 데이터를 내보냅니다. AWS Key Management Service AWS KMS
- 보존 기간이 있고 S3 객체 잠금이 사용 설정된 S3 버킷으로 로그 데이터 내보내기

> **Note**
>
> Amazon S3로 내보내기는 표준 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

내보내기 프로세스를 시작하려면 S3 버킷을 생성해서 내보낸 로그 데이터를 저장해야 합니다. S3 버킷에 내보낸 파일을 저장하고 Amazon S3 수명 주기 규칙을 정의하여 내보낸 파일을 자동으로 보관하거나 삭제할 수 있습니다.

AES-256 또는 SSE-KMS로 암호화된 S3 버킷으로 내보낼 수 있습니다. DSSE-KMS로 암호화된 버킷으로의 내보내기는 지원되지 않습니다.

여러 로그 그룹이나 시간 범위에서 내보낸 로그를 동일한 S3 버킷으로 내보낼 수 있습니다. 각 내보내기 작업에 대해 로그 데이터를 분리하려면 내보낸 모든 객체에서 Amazon S3 키 접두사로 사용될 접두사를 지정할 수 있습니다.

> **Note**
>
> 내보낸 파일 내의 로그 데이터 청크에 대한 시간 기반 정렬은 보장되지 않습니다. Linux 유틸리티를 사용하여 내보낸 로그 필드 데이터를 정렬할 수 있습니다. 예를 들어, 다음 유틸리티 명령은 단일 폴더에 있는 모든 `.gz` 파일의 이벤트를 정렬합니다.
>
> ```
> find . -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
> ```
>
> 다음 유틸리티 명령은 여러 하위 폴더에서.gz 파일을 정렬합니다.

```
find ./*/ -type f -exec zcat {} + | sed -r 's/^[0-9]+/\x0&/' | sort -z
```

또한 다른 stdout 명령을 사용하여 정렬된 출력을 다른 파일로 파이프하여 저장할 수 있습니다.

로그 데이터가 내보내기가 가능한 상태가 되는 데는 최대 12시간이 걸릴 수 있습니다. 내보내기 작업은 24시간 이후 시간 초과됩니다. 내보내기 작업의 시간이 초과될 경우 내보내기 작업을 생성할 때 시간 범위를 줄이세요.

로그 데이터를 실시간에 가깝게 분석하려면 대신에 로그 인사이트를 통한 CloudWatch 로그 데이터 분석 또는 구독을 통한 로그 데이터 실시간 처리 섹션을 참조하세요.

내용

개념

내보내기를 시작하기 전에 다음 개념을 익힙니다.

log_group_name

내보내기 작업과 연관된 로그 그룹의 이름. 이 로그 그룹의 로그 데이터는 지정된 S3 버킷으로 내보내집니다.

(타임스탬프)부터

1970년 1월 1일 00:00:00 UTC 이후 경과된 시간(밀리초)로 표현되는 필수 타임스탬프. 이 기간 또는 이후에 수집된 로그 그룹의 모든 로그 이벤트를 내보냅니다.

(타임스탬프)까지

1970년 1월 1일 1:1970 00:00:00 UTC 이후 경과된 시간(밀리초)로 표현되는 필수 타임스탬프. 이 시간이 내보내기 되기 전에 수집된 로그 그룹의 모든 로그 이벤트.

대상 버킷

내보내기 작업과 연관된 S3 버킷의 이름. 이 버킷은 지정된 로그 그룹에서 로그 데이터를 내보내는 데 사용됩니다.

대상 접두사

내보낸 모든 객체에 대한 Amazon S3 키 접두사로 사용되는 선택적인 속성. 이 옵션은 버킷에 폴더 같은 조직을 생성하는 데 도움이 됩니다.

콘솔을 사용하여 Amazon S3로 로그 데이터 내보내기

다음 예시에서는 Amazon CloudWatch 콘솔을 사용하여 이름이 지정된 Amazon CloudWatch Logs 로그 그룹에서 이름이 지정된 Amazon S3 `my-log-group` 버킷으로 모든 데이터를 내보냅니다`my-exported-logs`.

SSE-KMS로 암호화된 S3 버킷으로 로그 데이터 내보내기는 지원됩니다. DSSE-KMS로 암호화된 버킷으로의 내보내기는 지원되지 않습니다.

내보내기 설정 방법에 대한 세부 정보는 내보내려는 Amazon S3 버킷이 내보내는 로그와 동일한 계정에 있는지, 또는 다른 계정에 있는지에 따라 다릅니다.

주제

- 동일한 계정에 내보내기
- 다른 계정에 내보내기

동일한 계정에 내보내기

Amazon S3 버킷이 내보내는 로그와 동일한 계정에 있는 경우 이 섹션의 지침을 사용합니다.

주제

- 1단계: Amazon S3 버킷 생성
- 2단계: 액세스 권한 설정
- 3단계: S3 버킷에 대한 권한 설정
- (선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기
- 5단계: 내보내기 작업 생성

1단계: Amazon S3 버킷 생성

CloudWatch 로그용으로 특별히 생성된 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

> **Note**
>
> S3 버킷은 내보낼 로그 데이터와 동일한 지역에 있어야 합니다. CloudWatch 로그는 다른 지역의 S3 버킷으로 데이터를 내보내는 것을 지원하지 않습니다.

S3 버킷을 생성하려면,

1. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 필요한 경우, 지역을 변경합니다. 탐색 표시줄에서 CloudWatch 로그가 있는 지역을 선택합니다.
3. Create Bucket(버킷 생성)을 선택합니다.
4. Bucket Name(버킷 이름)에 버킷의 이름을 입력합니다.
5. 지역의 경우 CloudWatch 로그 데이터가 있는 지역을 선택합니다.
6. 생성을 선택합니다.

2단계: 액세스 권한 설정

5단계에서 내보내기 작업을 생성하려면 `AmazonS3ReadOnlyAccess` IAM 역할과 다음 권한으로 로그인해야 합니다.

- `logs:CreateExportTask`
- `logs:CancelExportTask`
- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- 다음 지역의 AWS IAM Identity Center사용자 및 그룹:

권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.

- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

3단계: S3 버킷에 대한 권한 설정

기본적으로 모든 S3 버킷 및 객체는 비공개입니다. 리소스 소유자(버킷을 생성한 AWS 계정)만 버킷과 해당 버킷이 포함하는 객체에 액세스할 수 있습니다. 그러나 리소스 소유자는 액세스 정책을 작성하여 다른 리소스 및 사용자에게 액세스 권한을 부여할 수 있습니다.

정책을 설정할 때 의도한 로그 스트림만 버킷으로 내보내도록 임의로 생성되는 문자열을 버킷의 접두사로 포함할 것을 권장합니다.

> **Important**
>
> S3 버킷을 더 안전하게 내보내려면 이제 로그 데이터를 S3 버킷으로 내보낼 수 있는 소스 계정 목록을 지정해야 합니다.
> 다음 예제에서 `aws:SourceAccount` 키의 계정 ID 목록은 사용자가 로그 데이터를 S3 버킷으로 내보낼 수 있는 계정입니다. `aws:SourceArn` 키는 작업이 수행되는 리소스가 됩니다. 이를 특정 로그 그룹으로 제한하거나 이 예와 같이 와일드카드를 사용할 수 있습니다.
> 동일한 계정 내에서 내보내기를 허용하려면 S3 버킷이 생성된 계정의 계정 ID도 포함하는 것이 좋습니다.

Amazon S3 버킷에 대한 권한을 설정하려면

1. Amazon S3 콘솔에서 1단계에서 생성한 버킷을 선택합니다.
2. 권한, 버킷 정책을 선택합니다.

3. Bucket Policy Editor(버킷 정책 편집기)에 다음 정책을 추가합니다. `my-exported-logs`을(를) S3 버킷의 이름으로 변경합니다. 보안 주체에 올바른 리전 엔드포인트(예: `us-west-1`)를 지정해야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs",
          "Principal": { "Service": "logs.Region.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "aws:SourceAccount": [
                    "AccountId1",
                    "AccountId2",
                    ...
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                     ]
            }
          }
      },
      {
          "Action": "s3:PutObject" ,
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs/*",
          "Principal": { "Service": "logs.Region.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                    "AccountId1",
                    "AccountId2",
                    ...
                ]
            },
```

```
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                    ]
            }
          }
      }
    ]
}
```

4. 저장을 선택하여 버킷에서 액세스 정책으로 방금 추가한 정책을 설정합니다. 이 정책을 사용하면 CloudWatch 로그가 로그 데이터를 S3 버킷으로 내보낼 수 있습니다. 버킷 소유자는 내보낸 모든 개체에 대해 모든 권한을 가집니다.

> **Warning**
>
> 기존 버킷에 이미 하나 이상의 정책이 연결되어 있는 경우 해당 정책 또는 정책에 CloudWatch 로그 액세스에 대한 설명을 추가하십시오. 발생한 권한 집합이 버킷에 액세스하는 사용자에게 적절한지를 여부를 평가하는 것이 좋습니다.

(선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기

이 단계는 서버 측 암호화를 사용하는 S3 버킷으로 내보내는 경우에만 필요합니다. AWS KMS keys이 암호화를 SSE-KMS라고 합니다.

SSE-KMS로 암호화된 버킷으로 내보내려면

1. https://console.aws.amazon.com/kms 에서 AWS KMS 콘솔을 엽니다.
2. 를 변경하려면 페이지 오른쪽 상단의 지역 선택기를 사용하십시오. AWS 리전
3. 탐색 창에서 Customer managed keys(고객 관리형 키)를 선택합니다.

   Create key(키 생성)를 선택합니다.
4. 키 유형에 대해 대칭을 선택합니다.
5. Key usage(키 사용)에서 Encrypt and decrypt(암호화 및 해독)를 선택한 다음 Next(다음)를 선택합니다.

6. Add labels(레이블 추가)에서 키의 별칭을 입력하고 필요에 따라 설명이나 태그를 추가합니다. 그런 다음 Next(다음)를 선택합니다.
7. Key administrators(키 관리자)에서 이 키를 관리할 수 있는 사용자를 선택한 다음 Next(다음)를 선택합니다.
8. Define key usage permissions(키 사용 권한 정의)에서 변경하지 않고 Next(다음)를 선택합니다.
9. 설정을 검토한 다음 Finish(완료)를 선택합니다.
10. Customer managed keys(고객 관리형 키) 페이지로 돌아가서 방금 생성한 키의 이름을 선택합니다.
11. Key policy(키 정책) 탭에서 Switch to policy view(정책 보기로 전환)를 선택합니다.
12. Key policy(키 정책) 섹션에서 Edit(편집)를 선택합니다.
13. 키 정책 문 목록에 다음 문을 추가합니다. 그런 다음 *##*을 로그의 리전으로 바꾸고 *account-ARN*을 KMS 키를 소유한 계정의 ARN으로 바꿉니다.

```
{
    "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. 변경 사항 저장를 선택합니다.
15. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
16. 1단계: S3 버킷 생성에서 생성한 버킷을 찾아 버킷 이름을 선택합니다.
17. 속성(Properties) 탭을 선택합니다. Default Encryption(기본 암호화)에서 Edit(편집)를 선택합니다.
18. Server-side encryption(서버 측 암호화)에서 Enable(활성화)을 선택합니다.
19. 암호화 유형에서 AWS Key Management Service 키(SSE-KMS)를 선택합니다.
20. 키에서 선택을 선택하고 생성한 AWS KMS 키를 찾으세요.
21. Bucket Key(버킷 키)에서 Enable(활성화)을 선택합니다.
22. 변경 사항 저장를 선택합니다.

5단계: 내보내기 작업 생성

이 단계에서는 로그 그룹에서 로그를 내보낼 수 있도록 내보내기 태스크를 생성합니다.

CloudWatch 콘솔을 사용하여 Amazon S3로 데이터를 내보내려면

1. 2단계: 액세스 권한 설정에 설명된 대로 충분한 권한으로 로그인합니다.
2. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
3. 탐색 창에서 로그 그룹을 선택합니다.
4. 로그 그룹 화면에서 로그 그룹의 이름을 선택합니다.
5. Actions(작업)에서 Export data to Amazon S3(Amazon S3로 데이터 내보내기)를 선택합니다.
6. Export data to Amazon S3(Amazon S3로 데이터 내보내기) 화면의 Define data export(데이터 내보내기 정의)에서 From(부터) 및 To(까지)를 사용하여 내보낼 데이터에 대한 시간 범위를 설정합니다.
7. 로그 그룹에 여러 개의 로그 스트림이 있는 경우에는 로그 스트림 접두사를 제공하여 로그 그룹 데이터를 특정 스트림으로 제한할 수 있습니다. 고급을 선택하고 스트림 접두사에 로그 스트림 접두사를 입력합니다.
8. Choose S3 bucket(S3 버킷 선택)에서 S3 버킷과 연관된 계정을 선택합니다.

9. S3 bucket name(S3 버킷 이름)에서 S3 버킷을 선택합니다.
10. S3 Bucket prefix(S3 버킷 접두사)에 버킷 정책에서 지정한 임의로 생성된 문자열을 입력합니다.
11. Export(내보내기)를 선택하여 Amazon S3로 로그 데이터를 내보냅니다.
12. Amazon S3로 내보낸 로그 데이터의 상태를 보려면 Actions(작업)을 선택한 다음 View all exports to Amazon S3(Amazon S3에 대한 모든 내보내기 보기)를 선택합니다.

다른 계정에 내보내기

Amazon S3 버킷이 내보내는 로그와 다른 계정에 있는 경우 이 섹션의 지침을 사용합니다.

주제

1단계: Amazon S3 버킷 생성

CloudWatch 로그용으로 특별히 생성된 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

> **Note**
>
> S3 버킷은 내보낼 로그 데이터와 동일한 지역에 있어야 합니다. CloudWatch 로그는 다른 지역의 S3 버킷으로 데이터를 내보내는 것을 지원하지 않습니다.

S3 버킷을 생성하려면,

1. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
2. 필요한 경우, 지역을 변경합니다. 탐색 표시줄에서 CloudWatch 로그가 있는 지역을 선택합니다.
3. Create Bucket(버킷 생성)을 선택합니다.
4. Bucket Name(버킷 이름)에 버킷의 이름을 입력합니다.
5. 지역의 경우 CloudWatch 로그 데이터가 있는 지역을 선택합니다.

6. 생성을 선택합니다.

2단계: 액세스 권한 설정

먼저 새 IAM 정책을 생성하여 CloudWatch Logs가 대상 계정의 대상 Amazon S3 버킷에 대한 `s3:PutObject` 권한을 갖도록 해야 합니다.

생성하는 정책은 대상 버킷이 AWS KMS 암호화를 사용하는지 여부에 따라 달라집니다.

Amazon S3 버킷으로 로그를 내보내는 IAM 정책 생성:

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성(Create policy)을 선택합니다.
4. 정책 편집기 섹션에서 JSON을 선택합니다.
5. 대상 버킷이 AWS KMS 암호화를 사용하지 않는 경우 다음 정책을 편집기에 붙여넣습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-exported-logs/*"
        }
    ]
}
```

대상 버킷이 AWS KMS 암호화를 사용하는 경우 다음 정책을 편집기에 붙여넣습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::my-exported-logs/*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "ARN_OF_KMS_KEY"
    }
  ]
}
```

6. 다음을 선택합니다.
7. 정책 이름을 입력합니다. 이 이름을 사용하여 정책을 IAM 역할에 연결합니다.
8. 정책 생성을 선택하여 새 정책을 저장합니다.

5단계에서 내보내기 작업을 생성하려면 AmazonS3ReadOnlyAccess IAM 역할로 로그인해야 합니다. 또한 방금 생성한 IAM 정책과 다음 권한으로 로그인해야 합니다.

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- 내 사용자 및 그룹 AWS IAM Identity Center:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

3단계: S3 버킷에 대한 권한 설정

기본적으로 모든 S3 버킷 및 객체는 비공개입니다. 리소스 소유자(버킷을 생성한 AWS 계정)만 버킷과 해당 버킷이 포함하는 객체에 액세스할 수 있습니다. 그러나 리소스 소유자는 액세스 정책을 작성하여 다른 리소스 및 사용자에게 액세스 권한을 부여할 수 있습니다.

정책을 설정할 때 의도한 로그 스트림만 버킷으로 내보내도록 임의로 생성되는 문자열을 버킷의 접두사로 포함할 것을 권장합니다.

> **Important**
>
> S3 버킷을 더 안전하게 내보내려면 이제 로그 데이터를 S3 버킷으로 내보낼 수 있는 소스 계정 목록을 지정해야 합니다.
> 다음 예제에서 `aws:SourceAccount` 키의 계정 ID 목록은 사용자가 로그 데이터를 S3 버킷으로 내보낼 수 있는 계정입니다. `aws:SourceArn` 키는 작업이 수행되는 리소스가 됩니다. 이를 특정 로그 그룹으로 제한하거나 이 예와 같이 와일드카드를 사용할 수 있습니다.
> 동일한 계정 내에서 내보내기를 허용하려면 S3 버킷이 생성된 계정의 계정 ID도 포함하는 것이 좋습니다.

Amazon S3 버킷에 대한 권한을 설정하려면

1. Amazon S3 콘솔에서 1단계에서 생성한 버킷을 선택합니다.
2. 권한, 버킷 정책을 선택합니다.
3. Bucket Policy Editor(버킷 정책 편집기)에 다음 정책을 추가합니다. `my-exported-logs`을(를) S3 버킷의 이름으로 변경합니다. 보안 주체에 올바른 리전 엔드포인트(예: `us-west-1`)를 지정해야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs",
          "Principal": { "Service": "logs.Region.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "aws:SourceAccount": [
                    "AccountId1",
```

```
                    "AccountId2",
                    ...
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                     ]
            }
        }
    },
    {
        "Action": "s3:PutObject" ,
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::my-exported-logs/*",
        "Principal": { "Service": "logs.Region.amazonaws.com" },
        "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                    "AccountId1",
                    "AccountId2",
                    ...
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                    ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::create_export_task_caller_account:role/role_name"
        },
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::my-exported-logs/*",
        "Condition": {
```

```
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control"
            }
        }
    }
  ]
}
```

4. 저장을 선택하여 버킷에서 액세스 정책으로 방금 추가한 정책을 설정합니다. 이 정책을 사용하면 CloudWatch 로그가 로그 데이터를 S3 버킷으로 내보낼 수 있습니다. 버킷 소유자는 내보낸 모든 개체에 대해 모든 권한을 가집니다.

> **Warning**
>
> 기존 버킷에 이미 하나 이상의 정책이 연결되어 있는 경우 해당 정책 또는 정책에 CloudWatch 로그 액세스에 대한 설명을 추가하십시오. 발생한 권한 집합이 버킷에 액세스하는 사용자에게 적절한지를 여부를 평가하는 것이 좋습니다.

(선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기

이 단계는 서버 측 암호화를 사용하는 S3 버킷으로 내보내는 경우에만 필요합니다. AWS KMS keys이 암호화를 SSE-KMS라고 합니다.

SSE-KMS로 암호화된 버킷으로 내보내려면

1. https://console.aws.amazon.com/kms 에서 AWS KMS 콘솔을 엽니다.
2. 를 변경하려면 페이지 오른쪽 상단의 지역 선택기를 사용하십시오. AWS 리전
3. 탐색 창에서 Customer managed keys(고객 관리형 키)를 선택합니다.

   Create key(키 생성)를 선택합니다.
4. 키 유형에 대해 대칭을 선택합니다.
5. Key usage(키 사용)에서 Encrypt and decrypt(암호화 및 해독)를 선택한 다음 Next(다음)를 선택합니다.
6. Add labels(레이블 추가)에서 키의 별칭을 입력하고 필요에 따라 설명이나 태그를 추가합니다. 그런 다음 Next(다음)를 선택합니다.
7. Key administrators(키 관리자)에서 이 키를 관리할 수 있는 사용자를 선택한 다음 Next(다음)를 선택합니다.

8. Define key usage permissions(키 사용 권한 정의)에서 변경하지 않고 Next(다음)를 선택합니다.
9. 설정을 검토한 다음 Finish(완료)를 선택합니다.
10. Customer managed keys(고객 관리형 키) 페이지로 돌아가서 방금 생성한 키의 이름을 선택합니다.
11. Key policy(키 정책) 탭에서 Switch to policy view(정책 보기로 전환)를 선택합니다.
12. Key policy(키 정책) 섹션에서 Edit(편집)를 선택합니다.
13. 키 정책 문 목록에 다음 문을 추가합니다. 그런 다음 *##*을 로그의 리전으로 바꾸고 *account-ARN*을 KMS 키를 소유한 계정의 ARN으로 바꿉니다.

```
{
    "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": "*"
        },
        {
```

```
            "Sid": "Enable IAM Role Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::create_export_task_caller_account:role/role_name"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "ARN_OF_KMS_KEY"
        }
    ]
}
```

14. 변경 사항 저장를 선택합니다.
15. https://console.aws.amazon.com/s3/에서 S3 콘솔을 엽니다.
16. 1단계: S3 버킷 생성에서 생성한 버킷을 찾아 버킷 이름을 선택합니다.
17. 속성(Properties) 탭을 선택합니다. Default Encryption(기본 암호화)에서 Edit(편집)를 선택합니다.
18. Server-side encryption(서버 측 암호화)에서 Enable(활성화)을 선택합니다.
19. 암호화 유형에서 AWS Key Management Service 키(SSE-KMS)를 선택합니다.
20. 키에서 선택을 선택하고 생성한 AWS KMS 키를 찾으세요.
21. Bucket Key(버킷 키)에서 Enable(활성화)을 선택합니다.
22. 변경 사항 저장를 선택합니다.

5단계: 내보내기 작업 생성

이 단계에서는 로그 그룹에서 로그를 내보낼 수 있도록 내보내기 태스크를 생성합니다.

CloudWatch 콘솔을 사용하여 Amazon S3로 데이터를 내보내려면

1. 2단계: 액세스 권한 설정에 설명된 대로 충분한 권한으로 로그인합니다.
2. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
3. 탐색 창에서 로그 그룹을 선택합니다.
4. 로그 그룹 화면에서 로그 그룹의 이름을 선택합니다.
5. Actions(작업)에서 Export data to Amazon S3(Amazon S3로 데이터 내보내기)를 선택합니다.

6. Export data to Amazon S3(Amazon S3로 데이터 내보내기) 화면의 Define data export(데이터 내보내기 정의)에서 From(부터) 및 To(까지)를 사용하여 내보낼 데이터에 대한 시간 범위를 설정합니다.
7. 로그 그룹에 여러 개의 로그 스트림이 있는 경우에는 로그 스트림 접두사를 제공하여 로그 그룹 데이터를 특정 스트림으로 제한할 수 있습니다. 고급을 선택하고 스트림 접두사에 로그 스트림 접두사를 입력합니다.
8. Choose S3 bucket(S3 버킷 선택)에서 S3 버킷과 연관된 계정을 선택합니다.
9. S3 bucket name(S3 버킷 이름)에서 S3 버킷을 선택합니다.
10. S3 Bucket prefix(S3 버킷 접두사)에 버킷 정책에서 지정한 임의로 생성된 문자열을 입력합니다.
11. Export(내보내기)를 선택하여 Amazon S3로 로그 데이터를 내보냅니다.
12. Amazon S3로 내보낸 로그 데이터의 상태를 보려면 Actions(작업)을 선택한 다음 View all exports to Amazon S3(Amazon S3에 대한 모든 내보내기 보기)를 선택합니다.

를 사용하여 Amazon S3로 로그 데이터를 내보냅니다. AWS CLI

다음 예시에서는 내보내기 작업을 사용하여 이름이 지정된 로그 CloudWatch 로그 그룹에서 이름이 지정된 Amazon S3 `my-log-group` 버킷으로 모든 데이터를 내보냅니다`my-exported-logs`. 이 예제에서는 이름이 `my-log-group`인 로그 그룹을 이미 생성했다고 가정합니다.

에 의해 암호화된 S3 버킷으로 로그 데이터를 내보내는 AWS KMS 것이 지원됩니다. DSSE-KMS로 암호화된 버킷으로의 내보내기는 지원되지 않습니다.

내보내기 설정 방법에 대한 세부 정보는 내보내려는 Amazon S3 버킷이 내보내는 로그와 동일한 계정에 있는지, 또는 다른 계정에 있는지에 따라 다릅니다.

주제

- 동일한 계정에 내보내기
- 다른 계정에 내보내기

동일한 계정에 내보내기

Amazon S3 버킷이 내보내는 로그와 동일한 계정에 있는 경우 이 섹션의 지침을 사용합니다.

주제

- 1단계: S3 버킷 생성

1단계: S3 버킷 생성

CloudWatch 로그용으로 특별히 생성된 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

> **Note**
>
> S3 버킷은 내보낼 로그 데이터와 동일한 지역에 있어야 합니다. CloudWatch 로그는 다른 지역의 S3 버킷으로 데이터를 내보내는 것을 지원하지 않습니다.

를 사용하여 S3 버킷을 만들려면 AWS CLI

명령 프롬프트에서 다음 create-bucket 명령을 입력합니다. 여기서 `LocationConstraint`는 로그 데이터를 내보내는 리전입니다.

```
aws s3api create-bucket --bucket my-exported-logs --create-bucket-configuration
 LocationConstraint=us-east-2
```

출력의 예제는 다음과 같습니다.

```
{
    "Location": "/my-exported-logs"
}
```

2단계: 액세스 권한 설정

5단계에서 내보내기 작업을 생성하려면 `AmazonS3ReadOnlyAccess` IAM 역할과 다음 권한으로 로그인해야 합니다.

- `logs:CreateExportTask`
- `logs:CancelExportTask`

- `logs:DescribeExportTasks`
- `logs:DescribeLogStreams`
- `logs:DescribeLogGroups`

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center다음 지역의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.

- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.

- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

3단계: S3 버킷에 대한 권한 설정

기본적으로 모든 S3 버킷 및 객체는 비공개입니다. 리소스 소유자(버킷을 생성한 계정)만 해당 버킷과 버킷에 포함된 객체에 액세스할 수 있습니다. 그러나 리소스 소유자는 액세스 정책을 작성하여 다른 리소스 및 사용자에게 액세스 권한을 부여할 수 있습니다.

> **Important**
>
> S3 버킷을 더 안전하게 내보내려면 이제 로그 데이터를 S3 버킷으로 내보낼 수 있는 소스 계정 목록을 지정해야 합니다.
> 다음 예제에서 `aws:SourceAccount` 키의 계정 ID 목록은 사용자가 로그 데이터를 S3 버킷으로 내보낼 수 있는 계정입니다. `aws:SourceArn` 키는 작업이 수행되는 리소스가 됩니다. 이를 특정 로그 그룹으로 제한하거나 이 예와 같이 와일드카드를 사용할 수 있습니다.
> 동일한 계정 내에서 내보내기를 허용하려면 S3 버킷이 생성된 계정의 계정 ID도 포함하는 것이 좋습니다.

S3 버킷에 대한 권한을 설정하려면

1. 이름이 `policy.json`인 파일을 생성하고 다음 액세스 정책을 추가합니다. `my-exported-logs`을(를) S3 버킷의 이름으로 변경하고 `Principal`을(를) 로그 데이터(예: `us-west-1`)를 내보내는 리전의 엔드포인트로 변경합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성합니다. IAM 콘솔을 사용하지 마세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs",
          "Principal": { "Service": "logs.Region.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "aws:SourceAccount": [
                    "AccountId1",
                    "AccountId2",
                    ...
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                     ]
            }
          }
      },
      {
          "Action": "s3:PutObject" ,
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs/*",
          "Principal": { "Service": "logs.Region.amazonaws.com" },
          "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": [
                    "AccountId1",
                    "AccountId2",
```

```
                    ...
                ]
            },
            "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:logs:Region:AccountId1:log-group:*",
                        "arn:aws:logs:Region:AccountId2:log-group:*",
                        ...
                    ]
            }
        }
      }
    ]
}
```

2. put-bucket-policy명령을 사용하여 방금 버킷에 액세스 정책으로 추가한 정책을 설정합니다. 이 정책을 사용하면 CloudWatch 로그가 로그 데이터를 S3 버킷으로 내보낼 수 있습니다. 버킷 소유자는 내보낸 모든 개체에 대해 모든 권한을 갖게 됩니다.

```
aws s3api put-bucket-policy --bucket my-exported-logs --policy file://policy.json
```

> **Warning**
>
> 기존 버킷에 이미 하나 이상의 정책이 연결되어 있는 경우 해당 정책 또는 정책에 CloudWatch 로그 액세스에 대한 설명을 추가하십시오. 발생한 권한 집합이 버킷에 액세스하는 사용자에게 적절한지를 여부를 평가하는 것이 좋습니다.

(선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기

이 단계는 서버 측 암호화를 사용하는 S3 버킷으로 내보내는 경우에만 필요합니다. AWS KMS keys이 암호화를 SSE-KMS라고 합니다.

SSE-KMS로 암호화된 버킷으로 내보내려면

1. 텍스트 편집기를 사용하여 이름이 `key_policy.json`인 파일을 생성하고 다음 액세스 정책을 추가합니다. 정책을 추가할 때 다음과 같이 변경합니다.

   - *Region*을 로그의 리전으로 바꿉니다.
   - *account-ARN*을 KMS 키를 소유하는 계정의 ARN으로 바꿉니다.

```
{
    "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. 다음 명령을 입력합니다.

```
aws kms create-key --policy file://key_policy.json
```

다음은 이 명령의 출력 예시입니다.

```
{
```

```
    "KeyMetadata": {
        "AWSAccountId": "account_id",
        "KeyId": "key_id",
        "Arn": "arn:aws:kms:us-east-2:account_id: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. 텍스트 편집기를 사용하여 다음 내용을 포함하는 bucketencryption.json이라는 파일을 생성합니다.

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
    }
  ]
}
```

4. 다음 명령을 입력하여 *bucket-name*을 로그를 내보내려는 버킷의 이름으로 바꿉니다.

```
aws s3api put-bucket-encryption --bucket bucket-name --server-side-encryption-
configuration file://bucketencryption.json
```

명령이 오류를 반환하지 않으면 프로세스가 성공한 것입니다.

5단계: 내보내기 작업 생성

다음 명령을 사용하여 내보내기 작업을 생성합니다. 생성하고 나면 내보낼 데이터의 크기에 따라 내보내기 작업에 몇 초부터 몇 시간까지 소요될 수 있습니다.

를 사용하여 Amazon S3로 데이터를 내보내려면 AWS CLI

1. 2단계: 액세스 권한 설정에 설명된 대로 충분한 권한으로 로그인합니다.
2. 명령 프롬프트에서 다음 create-export-task명령을 사용하여 내보내기 작업을 생성합니다.

```
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 "my-exported-logs" --destination-prefix "export-
task-output"
```

출력의 예제는 다음과 같습니다.

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

다른 계정에 내보내기

Amazon S3 버킷이 내보내는 로그와 다른 계정에 있는 경우 이 섹션의 지침을 사용합니다.

주제

- 1단계: S3 버킷 생성
- 2단계: 액세스 권한 설정
- 3단계: S3 버킷에 대한 권한 설정
- (선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기
- 5단계: 내보내기 작업 생성

1단계: S3 버킷 생성

CloudWatch 로그용으로 특별히 만든 버킷을 사용하는 것이 좋습니다. 그러나 기존 버킷을 사용하고 싶으면 2단계로 건너뛸 수 있습니다.

> **Note**
>
> S3 버킷은 내보낼 로그 데이터와 동일한 지역에 있어야 합니다. CloudWatch 로그는 다른 지역의 S3 버킷으로 데이터를 내보내는 것을 지원하지 않습니다.

를 사용하여 S3 버킷을 만들려면 AWS CLI

명령 프롬프트에서 다음 create-bucket 명령을 입력합니다. 여기서 LocationConstraint는 로그 데이터를 내보내는 리전입니다.

```
aws s3api create-bucket --bucket my-exported-logs --create-bucket-configuration
 LocationConstraint=us-east-2
```

출력의 예제는 다음과 같습니다.

```
{
    "Location": "/my-exported-logs"
}
```

2단계: 액세스 권한 설정

먼저, CloudWatch 로그가 대상 Amazon S3 버킷에 대한 s3:PutObject 권한을 갖도록 하려면 새 IAM 정책을 생성해야 합니다.

5단계에서 내보내기 작업을 생성하려면 AmazonS3ReadOnlyAccess IAM 역할과 특정 기타 권한으로 로그인해야 합니다. 이러한 기타 필요한 권한 중 일부를 포함하는 정책을 생성할 수 있습니다.

생성하는 정책은 대상 버킷이 AWS KMS 암호화를 사용하는지 여부에 따라 달라집니다. AWS KMS 암호화를 사용하지 않는 경우 다음 내용이 포함된 정책을 생성하십시오.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-exported-logs/*"
        }
    ]
}
```

대상 버킷이 AWS KMS 암호화를 사용하는 경우 다음 내용이 포함된 정책을 생성하십시오.

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::my-exported-logs/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "ARN_OF_KMS_KEY"
        }
    ]
}
```

5단계에서 내보내기 작업을 생성하려면 AmazonS3ReadOnlyAccess IAM 역할, 방금 생성한 IAM 정책 및 다음 권한으로 로그인해야 합니다.

- logs:CreateExportTask
- logs:CancelExportTask
- logs:DescribeExportTasks
- logs:DescribeLogStreams
- logs:DescribeLogGroups

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center다음 지역의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.

- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.

- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

3단계: S3 버킷에 대한 권한 설정

기본적으로 모든 S3 버킷 및 객체는 비공개입니다. 리소스 소유자(버킷을 생성한 계정)만 해당 버킷과 버킷에 포함된 객체에 액세스할 수 있습니다. 그러나 리소스 소유자는 액세스 정책을 작성하여 다른 리소스 및 사용자에게 액세스 권한을 부여할 수 있습니다.

> **Important**
>
> S3 버킷을 더 안전하게 내보내려면 이제 로그 데이터를 S3 버킷으로 내보낼 수 있는 소스 계정 목록을 지정해야 합니다.
> 다음 예제에서 `aws:SourceAccount` 키의 계정 ID 목록은 사용자가 로그 데이터를 S3 버킷으로 내보낼 수 있는 계정입니다. `aws:SourceArn` 키는 작업이 수행되는 리소스가 됩니다. 이를 특정 로그 그룹으로 제한하거나 이 예와 같이 와일드카드를 사용할 수 있습니다.
> 동일한 계정 내에서 내보내기를 허용하려면 S3 버킷이 생성된 계정의 계정 ID도 포함하는 것이 좋습니다.

S3 버킷에 대한 권한을 설정하려면

1. 이름이 `policy.json`인 파일을 생성하고 다음 액세스 정책을 추가합니다. `my-exported-logs`을(를) S3 버킷의 이름으로 변경하고 `Principal`을(를) 로그 데이터(예: `us-west-1`)를 내보내는 리전의 엔드포인트로 변경합니다. 텍스트 편집기를 사용하여 이 정책 파일을 생성합니다. IAM 콘솔을 사용하지 마세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
          "Action": "s3:GetBucketAcl",
          "Effect": "Allow",
          "Resource": "arn:aws:s3:::my-exported-logs",
```

```
            "Principal": { "Service": "logs.Region.amazonaws.com" },
            "Condition": {
              "StringEquals": {
                  "aws:SourceAccount": [
                      "AccountId1",
                      "AccountId2",
                      ...
                  ]
              },
              "ArnLike": {
                      "aws:SourceArn": [
                          "arn:aws:logs:Region:AccountId1:log-group:*",
                          "arn:aws:logs:Region:AccountId2:log-group:*",
                          ...
                       ]
              }
            }
        },
        {
            "Action": "s3:PutObject" ,
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::my-exported-logs/*",
            "Principal": { "Service": "logs.Region.amazonaws.com" },
            "Condition": {
              "StringEquals": {
                  "s3:x-amz-acl": "bucket-owner-full-control",
                  "aws:SourceAccount": [
                      "AccountId1",
                      "AccountId2",
                      ...
                  ]
              },
              "ArnLike": {
                      "aws:SourceArn": [
                          "arn:aws:logs:Region:AccountId1:log-group:*",
                          "arn:aws:logs:Region:AccountId2:log-group:*",
                          ...
                      ]
              }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
```

```
            "AWS": "arn:aws:iam::create_export_task_caller_account:role/role_name"
          },
          "Action": "s3:PutObject",
          "Resource": "arn:aws:s3:::my-exported-logs/*",
          "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control"
            }
          }
       }
    ]
}
```

2. put-bucket-policy명령을 사용하여 방금 버킷에 액세스 정책으로 추가한 정책을 설정합니다. 이 정책을 사용하면 CloudWatch 로그가 로그 데이터를 S3 버킷으로 내보낼 수 있습니다. 버킷 소유자는 내보낸 모든 개체에 대해 모든 권한을 갖게 됩니다.

```
aws s3api put-bucket-policy --bucket my-exported-logs --policy file://policy.json
```

> **Warning**
>
> 기존 버킷에 이미 하나 이상의 정책이 연결되어 있는 경우 해당 정책 또는 정책에 CloudWatch 로그 액세스에 대한 설명을 추가하십시오. 발생한 권한 집합이 버킷에 액세스하는 사용자에게 적절한지를 여부를 평가하는 것이 좋습니다.

(선택 사항) 4단계: SSE-KMS로 암호화된 버킷으로 내보내기

이 단계는 서버 측 암호화를 사용하는 S3 버킷으로 내보내는 경우에만 필요합니다. AWS KMS keys이 암호화를 SSE-KMS라고 합니다.

SSE-KMS로 암호화된 버킷으로 내보내려면

1. 텍스트 편집기를 사용하여 이름이 key_policy.json인 파일을 생성하고 다음 액세스 정책을 추가합니다. 정책을 추가할 때 다음과 같이 변경합니다.

   - *Region*을 로그의 리전으로 바꿉니다.
   - *account-ARN*을 KMS 키를 소유하는 계정의 ARN으로 바꿉니다.

```
{
    "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": "*"
        },
        {
            "Sid": "Enable IAM Role Permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::create_export_task_caller_account:role/role_name"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
```

```
            "Resource": "ARN_OF_KMS_KEY"
        }
    ]
}
```

2. 다음 명령을 입력합니다.

```
aws kms create-key --policy file://key_policy.json
```

다음은 이 명령의 출력 예시입니다.

```
{
    "KeyMetadata": {
        "AWSAccountId": "account_id",
        "KeyId": "key_id",
        "Arn": "arn:aws:kms:us-east-2:account_id: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. 텍스트 편집기를 사용하여 다음 내용을 포함하는 bucketencryption.json이라는 파일을 생성합니다.

```
{
  "Rules": [
    {
      "ApplyServerSideEncryptionByDefault": {
        "SSEAlgorithm": "aws:kms",
        "KMSMasterKeyID": "{KMS Key ARN}"
      },
      "BucketKeyEnabled": true
```

```
    }
  ]
}
```

4. 다음 명령을 입력하여 *bucket-name*을 로그를 내보내려는 버킷의 이름으로 바꿉니다.

```
aws s3api put-bucket-encryption --bucket bucket-name --server-side-encryption-
configuration file://bucketencryption.json
```

명령이 오류를 반환하지 않으면 프로세스가 성공한 것입니다.

5단계: 내보내기 작업 생성

다음 명령을 사용하여 내보내기 작업을 생성합니다. 생성하고 나면 내보낼 데이터의 크기에 따라 내보내기 작업에 몇 초부터 몇 시간까지 소요될 수 있습니다.

를 사용하여 Amazon S3로 데이터를 내보내려면 AWS CLI

1. 2단계: 액세스 권한 설정에 설명된 대로 충분한 권한으로 로그인합니다.
2. 명령 프롬프트에서 다음 create-export-task명령을 사용하여 내보내기 작업을 생성합니다.

```
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 "my-exported-logs" --destination-prefix "export-
task-output"
```

출력의 예제는 다음과 같습니다.

```
{
    "taskId": "cda45419-90ea-4db5-9833-aade86253e66"
}
```

내보내기 작업 설명

내보내기 작업을 생성하고 나면 작업의 현재 상태를 파악할 수 있습니다.

를 사용하여 내보내기 작업을 설명하려면 AWS CLI

명령 프롬프트에서 다음 describe-export-tasks명령을 사용합니다.

```
aws logs --profile CWLExportUser describe-export-tasks --task-id
 "cda45419-90ea-4db5-9833-aade86253e66"
```

출력의 예제는 다음과 같습니다.

```
{
   "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
   }]
}
```

세 가지 방법으로 describe-export-tasks 명령을 사용할 수 있습니다.

- 필터가 없는 경우 - 생성 순서의 역순으로 모든 내보내기 작업을 나열합니다.
- 작업 ID 필터링 - 지정된 ID를 가진 내보내기 작업(존재하는 경우)을 나열합니다.
- 작업 상태 필터링 - 지정된 상태를 가진 내보내기 작업을 나열합니다.

예를 들어, 다음 명령을 사용하여 FAILED 상태를 필터링합니다.

```
aws logs --profile CWLExportUser describe-export-tasks --status-code "FAILED"
```

출력의 예제는 다음과 같습니다.

```
{
   "exportTasks": [
   {
```

```
      "destination": "my-exported-logs",
      "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
   }]
}
```

내보내기 작업 취소

PENDING 또는 RUNNING 상태인 경우 내보내기 작업을 취소할 수 있습니다.

를 사용하여 내보내기 작업을 취소하려면 AWS CLI

명령 프롬프트에서 다음 cancel-export-task명령을 사용합니다.

```
aws logs --profile CWLExportUser cancel-export-task --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

describe-export-tasks명령을 사용하여 작업이 성공적으로 취소되었는지 확인할 수 있습니다.

Amazon OpenSearch 서비스로 CloudWatch 로그 데이터 스트리밍

CloudWatch Logs 구독을 통해 Amazon OpenSearch Service 클러스터로 수신한 데이터를 거의 실시간으로 스트리밍하도록 CloudWatch 로그 로그 그룹을 구성할 수 있습니다. 자세한 정보는 구독을 통한 로그 데이터 실시간 처리을 참조하세요.

> **Note**
>
> OpenSearch 서비스로의 스트리밍은 표준 로그 클래스의 로그 그룹에만 지원됩니다. 로그 클래스에 대한 자세한 내용은 을 참조하십시오로그 클래스.

스트리밍되는 로그 데이터의 양에 따라 함수에 함수 수준의 동시 실행 제한을 설정할 수 있습니다. 자세한 내용은 Lambda 함수 크기 조정을 참조하세요.

> **Note**
>
> 대량의 CloudWatch 로그 데이터를 OpenSearch 서비스로 스트리밍하면 사용 요금이 높아질 수 있습니다. AWS Billing and Cost Management 콘솔에서 예산을 생성하는 것이 좋습니다. 자세한 내용은 AWS Budgets을 통해 비용 관리를 참조하세요.

필수 조건

시작하기 전에 OpenSearch 서비스 도메인을 생성하세요. 도메인은 퍼블릭 액세스 또는 VPC 액세스 중 하나를 가질 수 있지만 도메인을 생성한 후에는 액세스 유형을 수정할 수 없습니다. 나중에 OpenSearch 서비스 도메인 설정을 검토하고 클러스터에서 처리할 데이터의 양에 따라 클러스터 구성을 수정하는 것이 좋습니다. 도메인을 생성하는 방법에 대한 지침은 OpenSearch 서비스 도메인 생성을 참조하십시오.

OpenSearch 서비스에 대한 자세한 내용은 Amazon OpenSearch 서비스 개발자 안내서를 참조하십시오.

로그 그룹을 OpenSearch 서비스에 등록하십시오.

CloudWatch 콘솔을 사용하여 로그 그룹을 OpenSearch 서비스에 구독할 수 있습니다.

로그 그룹을 OpenSearch 서비스에 구독하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 로그 그룹을 선택합니다.
3. 로그 그룹의 이름을 선택합니다.
4. 작업, 구독 필터, Amazon OpenSearch 서비스 구독 필터 생성을 선택합니다.
5. 이 계정의 클러스터로 스트리밍할지 아니면 다른 계정의 클러스터로 스트리밍할지 선택합니다.
   - 이 계정을 선택한 경우 이전 단계에서 생성한 도메인을 선택합니다.
   - 다른 계정을 선택한 경우 도메인 ARN 및 엔드포인트를 제공합니다.
6. Lambda IAM 실행 역할의 경우, Lambda가 호출을 실행할 때 사용해야 하는 IAM 역할을 선택합니다. OpenSearch

   선택한 IAM 역할은 다음 요구 사항을 충족해야 합니다.

   - 신뢰 관계를 맺고 있는 `lambda.amazonaws.com`이 있어야 합니다.
   - 다음 정책이 포함되어야 합니다.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Action": [
                   "es:*"
               ],
               "Effect": "Allow",
               "Resource": "arn:aws:es:region:account-id:domain/target-domain-name/
   *"
           }
       ]
   }
   ```

   - 대상 OpenSearch 서비스 도메인이 VPC 액세스를 사용하는 경우 역할에 AWSLambdaVPCAccessExecutionRole정책이 연결되어 있어야 합니다. 이 Amazon 관리형 정책은 Lambda에 고객 VPC에 대한 액세스 권한을 부여하여 Lambda가 VPC의 엔드포인트에 쓸 수 있도록 합니다. OpenSearch
7. 로그 형식(Log format)에서 로그 형식을 선택합니다.

8. 구독 필터 패턴(Subscription filter pattern)에 로그 이벤트에서 찾을 용어나 패턴을 입력합니다. 이렇게 하면 원하는 데이터만 클러스터로 전송할 수 있습니다. OpenSearch 자세한 정보는 필터를 사용하여 로그 이벤트에서 지표 생성을 참조하세요.
9. (선택 사항) 테스트할 로그 데이터 선택에서 로그 스트림을 선택한 다음 패턴 테스트를 선택해서 검색 필터가 예상한 결과를 반환하고 있는지 확인합니다.
10. 스트리밍 시작을 선택합니다.

AWS SDK를 사용한 CloudWatch 로그의 코드 예제

다음 코드 예제는 AWS 소프트웨어 개발 키트 (SDK) 와 함께 CloudWatch 로그를 사용하는 방법을 보여줍니다.

작업은 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 작업은 개별 서비스 함수를 호출하는 방법을 보여 주며 관련 시나리오와 교차 서비스 예시에서 컨텍스트에 맞는 작업을 볼 수 있습니다.

시나리오는 동일한 서비스 내에서 여러 함수를 호출하여 특정 태스크를 수행하는 방법을 보여주는 코드 예시입니다.

교차 서비스 예시는 여러 AWS 서비스전반에서 작동하는 샘플 애플리케이션입니다.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

코드 예시

- SDK를 사용한 CloudWatch AWS 로그 작업
  - AWS SDK 또는 AssociateKmsKey CLI와 함께 사용
  - AWS SDK 또는 CancelExportTask CLI와 함께 사용
  - AWS SDK 또는 CreateExportTask CLI와 함께 사용
  - AWS SDK 또는 CreateLogGroup CLI와 함께 사용
  - AWS SDK 또는 CreateLogStream CLI와 함께 사용
  - AWS SDK 또는 DeleteLogGroup CLI와 함께 사용
  - AWS SDK 또는 DeleteSubscriptionFilter CLI와 함께 사용
  - AWS SDK 또는 DescribeExportTasks CLI와 함께 사용
  - AWS SDK 또는 DescribeLogGroups CLI와 함께 사용
  - AWS SDK 또는 DescribeSubscriptionFilters CLI와 함께 사용
  - AWS SDK 또는 GetQueryResults CLI와 함께 사용
  - AWS SDK 또는 PutSubscriptionFilter CLI와 함께 사용
  - AWS SDK 또는 StartLiveTail CLI와 함께 사용
  - AWS SDK 또는 StartQuery CLI와 함께 사용
- SDK를 사용한 CloudWatch AWS 로그 시나리오

  - CloudWatch 로그를 사용하여 대규모 쿼리를 실행할 수 있습니다.
- SDK를 사용한 CloudWatch 로그의 크로스 서비스 예제 AWS
  - 예약된 이벤트를 사용하여 Lambda 함수 호출

SDK를 사용한 CloudWatch AWS 로그 작업

다음 코드 예제는 AWS SDK를 사용하여 개별 CloudWatch Logs 작업을 수행하는 방법을 보여줍니다. 이들 발췌문은 CloudWatch Logs API를 호출하며, 컨텍스트에서 실행해야 하는 대규모 프로그램에서 발췌한 코드입니다. 각 예제에는 코드 설정 및 실행 지침을 찾을 수 있는 링크가 포함되어 있습니다. GitHub

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 Amazon CloudWatch Logs API 레퍼런스를 참조하십시오.

예

- AWS SDK 또는 AssociateKmsKey CLI와 함께 사용
- AWS SDK 또는 CancelExportTask CLI와 함께 사용
- AWS SDK 또는 CreateExportTask CLI와 함께 사용
- AWS SDK 또는 CreateLogGroup CLI와 함께 사용
- AWS SDK 또는 CreateLogStream CLI와 함께 사용
- AWS SDK 또는 DeleteLogGroup CLI와 함께 사용
- AWS SDK 또는 DeleteSubscriptionFilter CLI와 함께 사용
- AWS SDK 또는 DescribeExportTasks CLI와 함께 사용
- AWS SDK 또는 DescribeLogGroups CLI와 함께 사용
- AWS SDK 또는 DescribeSubscriptionFilters CLI와 함께 사용
- AWS SDK 또는 GetQueryResults CLI와 함께 사용
- AWS SDK 또는 PutSubscriptionFilter CLI와 함께 사용
- AWS SDK 또는 StartLiveTail CLI와 함께 사용
- AWS SDK 또는 StartQuery CLI와 함께 사용

AWS SDK 또는 **`AssociateKmsKey`** CLI와 함께 사용

다음 코드 예시에서는 `AssociateKmsKey`을 사용하는 방법을 보여 줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    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}.");
            }
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API AssociateKmsKey참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CancelExportTask** CLI와 함께 사용

다음 코드 예시에서는 CancelExportTask을 사용하는 방법을 보여 줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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.");
        }
    }
}
```

- API 세부 정보는 AWS SDK for .NET API CancelExportTask참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateExportTask** CLI와 함께 사용

다음 코드 예시에서는 CreateExportTask을 사용하는 방법을 보여 줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   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 = "doc-example-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.");
            }
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API CreateExportTask참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateLogGroup** CLI와 함께 사용

다음 코드 예제는 CreateLogGroup의 사용 방법을 보여줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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.");
            }
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API CreateLogGroup참조를 참조하십시오.

CLI

AWS CLI

다음 명령은 이름이 `my-logs`인 로그 그룹을 생성합니다.

```
aws logs create-log-group --log-group-name my-logs
```

- API 세부 정보는 AWS CLI 명령 CreateLogGroup참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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();
```

- API 세부 정보는 AWS SDK for JavaScript API CreateLogGroup참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateLogStream** CLI와 함께 사용

다음 코드 예제는 CreateLogStream의 사용 방법을 보여줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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.");
            }
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API CreateLogStream참조를 참조하십시오.

CLI

AWS CLI

다음 명령은 my-logs 로그 그룹에서 이름이 20150601인 로그 스트림을 생성합니다.

```
aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601
```

- API 세부 정보는 AWS CLI 명령 CreateLogStream참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteLogGroup** CLI와 함께 사용

다음 코드 예제는 DeleteLogGroup의 사용 방법을 보여줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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}.");
        }
    }
}
```

- API 세부 정보는 AWS SDK for .NET API DeleteLogGroup참조를 참조하십시오.

CLI

AWS CLI

다음 명령은 이름이 `my-logs`인 로그 그룹을 삭제합니다.

```
aws logs delete-log-group --log-group-name my-logs
```

- API 세부 정보는 AWS CLI 명령 DeleteLogGroup참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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();
```

- API 세부 정보는 AWS SDK for JavaScript API DeleteLogGroup참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteSubscriptionFilter** CLI와 함께 사용

다음 코드 예제는 DeleteSubscriptionFilter의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 더 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

필수 파일을 포함합니다.

```
#include <aws/core/Aws.h>
#include <aws/core/utils/Outcome.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/DeleteSubscriptionFilterRequest.h>
#include <iostream>
```

구독 필터를 삭제합니다.

```
        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;
        }
```

- API 세부 정보는 AWS SDK for C++ API DeleteSubscriptionFilter참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API DeleteSubscriptionFilter참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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();
```

- API 세부 정보는 AWS SDK for JavaScript API DeleteSubscriptionFilter참조를 참조하십시오.

JavaScript (v2) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

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

- 자세한 정보는 AWS SDK for JavaScript 개발자 안내서를 참조하십시오.
- API 세부 정보는 AWS SDK for JavaScript API DeleteSubscriptionFilter참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun deleteSubFilter(
    filter: String?,
    logGroup: String?,
) {
    val request =
```

```
        DeleteSubscriptionFilterRequest {
            filterName = filter
            logGroupName = logGroup
        }

    CloudWatchLogsClient { region = "us-west-2" }.use { logs ->
        logs.deleteSubscriptionFilter(request)
        println("Successfully deleted CloudWatch logs subscription filter named
 $filter")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DeleteSubscriptionFilter.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeExportTasks** CLI와 함께 사용

다음 코드 예시에서는 DescribeExportTasks을 사용하는 방법을 보여 줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeExportTasks참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeLogGroups** CLI와 함께 사용

다음 코드 예제는 DescribeLogGroups의 사용 방법을 보여줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeLogGroups참조를 참조하십시오.

CLI

AWS CLI

다음 명령은 이름이 `my-logs`인 로그 그룹을 설명합니다.

```
aws logs describe-log-groups --log-group-name-prefix my-logs
```

출력:

```
{
    "logGroups": [
        {
            "storedBytes": 0,
            "metricFilterCount": 0,
            "creationTime": 1433189500783,
            "logGroupName": "my-logs",
            "retentionInDays": 5,
            "arn": "arn:aws:logs:us-west-2:0123456789012:log-group:my-logs:*"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeLogGroups참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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 && page.logGroups.every((lg) => !!lg)) {
      logGroups.push(...page.logGroups);
    }
  }
```

```
  console.log(logGroups);
  return logGroups;
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeLogGroups참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSubscriptionFilters** CLI와 함께 사용

다음 코드 예제는 DescribeSubscriptionFilters의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 더 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

필수 파일을 포함합니다.

```
#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>
```

구독 필터를 나열합니다.

```
        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();
        }
```

- API 세부 정보는 AWS SDK for C++ API DescribeSubscriptionFilters참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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");
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeSubscriptionFilters참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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();
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeSubscriptionFilters참조를 참조하십시오.

JavaScript (v2) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

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

- 자세한 정보는 AWS SDK for JavaScript 개발자 안내서를 참조하십시오.
- API 세부 정보는 AWS SDK for JavaScript API DescribeSubscriptionFilters참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun describeFilters(logGroup: String) {
    val request =
        DescribeSubscriptionFiltersRequest {
            logGroupName = logGroup
            limit = 1
        }

    CloudWatchLogsClient { 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}")
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeSubscriptionFilters.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **GetQueryResults** CLI와 함께 사용

다음 코드 예제는 GetQueryResults의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 대용량 쿼리 실행

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/**
 * Simple wrapper for the GetQueryResultsCommand.
 * @param {string} queryId
 */
_getQueryResults(queryId) {
  return this.client.send(new GetQueryResultsCommand({ queryId }));
}
```

- API 세부 정보는 AWS SDK for JavaScript API GetQueryResults참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    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", [])
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 GetQueryResults.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **PutSubscriptionFilter** CLI와 함께 사용

다음 코드 예제는 PutSubscriptionFilter의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 더 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

필수 파일을 포함합니다.

```
#include <aws/core/Aws.h>
#include <aws/logs/CloudWatchLogsClient.h>
#include <aws/logs/model/PutSubscriptionFilterRequest.h>
#include <aws/core/utils/Outcome.h>
#include <iostream>
```

구독 필터를 생성합니다.

```
        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;
        }
```

- API 세부 정보는 AWS SDK for C++ API PutSubscriptionFilter참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API PutSubscriptionFilter참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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();
```

- API 세부 정보는 AWS SDK for JavaScript API PutSubscriptionFilter참조를 참조하십시오.

JavaScript (v2) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

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

- 자세한 정보는 AWS SDK for JavaScript 개발자 안내서를 참조하십시오.
- API 세부 정보는 AWS SDK for JavaScript API PutSubscriptionFilter참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **StartLiveTail** CLI와 함께 사용

다음 코드 예제는 StartLiveTail의 사용 방법을 보여줍니다.

.NET

AWS SDK for .NET

필수 파일을 포함합니다.

```
using Amazon;
using Amazon.CloudWatchLogs;
using Amazon.CloudWatchLogs.Model;
```

Live Tail 세션을 시작합니다.

```
            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;
            }
```

Live Tail 세션의 이벤트는 두 가지 방법으로 처리할 수 있습니다.

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

- API 세부 정보는 AWS SDK for .NET API StartLiveTail참조를 참조하십시오.

Go

SDK for Go V2

필수 파일을 포함합니다.

```
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"
)
```

Live Tail 세션의 이벤트를 처리합니다.

```
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)
   }
  }
 }
}
```

Live Tail 세션을 시작합니다.

```
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)
```

일정 시간이 경과하면 Live Tail 세션을 중단합니다.

```
// Close the stream (which ends the session) after a timeout
time.Sleep(10 * time.Second)
stream.Close()
log.Println("Event stream closed")
```

- API 세부 정보는 AWS SDK for Go API StartLiveTail참조를 참조하십시오.

Java

SDK for Java 2.x

필수 파일을 포함합니다.

```
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;
```

Live Tail 세션의 이벤트를 처리합니다.

```
    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();
    }
```

Live Tail 세션을 시작합니다.

```
        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));
```

일정 시간이 경과하면 Live Tail 세션을 중단합니다.

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

- API 세부 정보는 AWS SDK for Java 2.x API StartLiveTail참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

필수 파일을 포함합니다.

```
import { CloudWatchLogsClient, StartLiveTailCommand } from "@aws-sdk/client-
cloudwatch-logs";
```

Live Tail 세션의 이벤트를 처리합니다.

```
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)
    }
}
```

Live Tail 세션을 시작합니다.

```
    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);
    }
```

일정 시간이 경과하면 Live Tail 세션을 중단합니다.

```
    /* Set a timeout to close the client. This will stop the Live Tail session.
*/
    setTimeout(function() {
        console.log("Client timeout");
        client.destroy();
      }, 10000);
```

- API에 대한 자세한 내용은 API StartLiveTail레퍼런스를 참조하십시오.AWS SDK for JavaScript

Kotlin

SDK for Kotlin

필수 파일을 포함합니다.

```
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
```

Live Tail 세션을 시작합니다.

```
    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)
    }
```

- API 세부 정보는 Kotlin API 참조용AWS SDK를 참조하세요 StartLiveTail.

Python

SDK for Python(Boto3)

필수 파일을 포함합니다.

```
import boto3
import time
from datetime import datetime
```

Live Tail 세션을 시작합니다.

```
    # 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)
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 StartLiveTail.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **StartQuery** CLI와 함께 사용

다음 코드 예제는 StartQuery의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 대용량 쿼리 실행

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/**
 * 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;
    }
  }
```

- API 세부 정보는 AWS SDK for JavaScript API StartQuery참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    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_groups,
                    startTime=start_time,
                    endTime=end_time,
                    queryString="fields @timestamp, @message | sort @timestamp
asc",
                    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_groups,
                startTime=start_time,
                endTime=end_time,
                queryString="fields @timestamp, @message | sort @timestamp asc",
                limit=max_logs,
            )
            return response["queryId"]
        except client.exceptions.ResourceNotFoundException as e:
            raise DateOutOfBoundsError(f"Resource not found: {e}")
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 StartQuery.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK와 함께 CloudWatch 로그 사용 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

SDK를 사용한 CloudWatch AWS 로그 시나리오

다음 코드 예제는 CloudWatch Logs with AWS SDK에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. 이 시나리오는 로그 내에서 여러 함수를 호출하여 특정 작업을 수행하는 방법을 보여줍니다. CloudWatch 각 시나리오에는 코드 설정 및 실행 방법에 대한 지침을 찾을 수 있는 링크가 포함되어 있습니다. GitHub

예

- CloudWatch 로그를 사용하여 대규모 쿼리를 실행할 수 있습니다.

CloudWatch 로그를 사용하여 대규모 쿼리를 실행할 수 있습니다.

다음 코드 예제는 CloudWatch 로그를 사용하여 10,000개 이상의 레코드를 쿼리하는 방법을 보여줍니다.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

진입점입니다.

```
// 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(parseInt(process.env.QUERY_START_DATE)),
    new Date(parseInt(process.env.QUERY_END_DATE)),
  ],
});

await cloudWatchQuery.run();

console.log(
```

```
  `Queries finished in ${cloudWatchQuery.secondsElapsed} seconds.\nTotal logs
 found: ${cloudWatchQuery.results.length}`,
);
```

필요한 경우 쿼리를 여러 단계로 분할하는 클래스입니다.

```
// 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]);
  }

  // snippet-start:[javascript.v3.cloudwatch-logs.actions.GetQueryResults]
  /**
   * Simple wrapper for the GetQueryResultsCommand.
   * @param {string} queryId
   */
  _getQueryResults(queryId) {
    return this.client.send(new GetQueryResultsCommand({ queryId }));
  }
  // snippet-end:[javascript.v3.cloudwatch-logs.actions.GetQueryResults]

  /**
   * 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 [];
      } else {
        throw err;
      }
    }
  }

  // snippet-start:[javascript.v3.cloudwatch-logs.actions.StartQuery]
  /**
   * 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;
    }
  }
  // snippet-end:[javascript.v3.cloudwatch-logs.actions.StartQuery]

  /**
   * 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;
      },
    );
  }
}
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 다음 주제를 참조하십시오.

- GetQueryResults
- StartQuery

Python

SDK for Python(Boto3)

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

이 파일은 10,000개의 결과를 초과하는 CloudWatch 쿼리를 관리하기 위한 예제 모듈을 호출합니다.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
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",
)


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.

        :return: Tuple of query start date and end date as integers.
        :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)

        return query_start_date, query_end_date
```

```
    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",
    ):
        """
        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
        """
        cloudwatch_query = CloudWatchQuery(
            [start_date_iso8601, end_date_iso8601],
        )
        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 = 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)


if __name__ == "__main__":
    main()
```

이 모듈은 10,000개 이상의 결과를 처리하는 CloudWatch 쿼리를 처리합니다.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
import logging
import time
from datetime import datetime
import threading
import boto3

from date_utilities import DateUtilities


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.

    :ivar date_range: Start and end datetime for the query.
    :vartype date_range: tuple
    :ivar limit: Maximum number of log entries to return.
    :vartype limit: int
    """

    def __init__(self, date_range):
        self.lock = threading.Lock()
        self.log_groups = "/workflows/cloudwatch-logs/large-query"
        self.query_results = []
        self.date_range = date_range
        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}"
        )
        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

   # snippet-start:[python.example_code.cloudwatch_logs.start_query]
   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_groups,
                    startTime=start_time,
                    endTime=end_time,
                    queryString="fields @timestamp, @message | sort @timestamp
asc",
                    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_groups,
                startTime=start_time,
                endTime=end_time,
                queryString="fields @timestamp, @message | sort @timestamp asc",
                limit=max_logs,
            )
            return response["queryId"]
        except client.exceptions.ResourceNotFoundException as e:
            raise DateOutOfBoundsError(f"Resource not found: {e}")

    # snippet-end:[python.example_code.cloudwatch_logs.start_query]

    # snippet-start:[python.example_code.cloudwatch_logs.get_query_results]
    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", [])
```

```
    # snippet-end:[python.example_code.cloudwatch_logs.get_query_results]
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하십시오.
  - GetQueryResults
  - StartQuery

AWS SDK 개발자 가이드의 전체 목록과 코드 예제는 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

SDK를 사용한 CloudWatch 로그의 크로스 서비스 예제 AWS

다음 샘플 애플리케이션은 AWS SDK를 사용하여 CloudWatch 로그를 다른 Log와 결합합니다. AWS 서비스각 예제에는 애플리케이션 설정 및 실행 방법에 대한 지침을 찾을 수 있는 링크가 포함되어 있습니다. GitHub

예

- 예약된 이벤트를 사용하여 Lambda 함수 호출

예약된 이벤트를 사용하여 Lambda 함수 호출

다음 코드 예제는 Amazon EventBridge 예약 이벤트에 의해 호출되는 AWS Lambda 함수를 생성하는 방법을 보여줍니다.

Python

SDK for Python(Boto3)

이 예제에서는 AWS Lambda 함수를 예약된 Amazon EventBridge 이벤트의 대상으로 등록하는 방법을 보여줍니다. Lambda 핸들러는 나중에 검색할 수 있도록 CloudWatch Amazon Logs에 친숙한 메시지와 전체 이벤트 데이터를 기록합니다.

- Lambda 함수를 배포합니다.
- EventBridge 예약된 이벤트를 생성하고 Lambda 함수를 대상으로 설정합니다.
- Lambda EventBridge 함수를 호출할 수 있는 권한을 부여합니다.
- CloudWatch Logs의 최신 데이터를 인쇄하여 예약된 호출의 결과를 표시합니다.

- 데모 중에 생성된 모든 리소스를 정리합니다.

이 예시는 에서 가장 잘 보입니다. GitHub 전체 소스 코드와 설정 및 실행 방법에 대한 지침은 의 전체 예제를 참조하십시오 GitHub.

이 예시에서 사용되는 서비스

- CloudWatch 로그
- EventBridge
- Lambda

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK와 함께 CloudWatch 로그 사용 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Amazon CloudWatch 로그의 보안

클라우드 AWS 보안이 최우선 과제입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 기업과 기업 간의 AWS 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호하는 역할을 합니다. AWS AWS 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 적용되는 규정 준수 프로그램에 대해 자세히 알아보려면 규정 준수 프로그램별 범위 내AWS 서비스 규정 준수 참조하십시오. WorkSpaces
- 클라우드에서의 보안 - 귀하의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 여러분은 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다

이 설명서는 Amazon CloudWatch Logs를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 보안 및 규정 준수 목표를 충족하도록 Amazon CloudWatch Logs를 구성하는 방법을 보여줍니다. 또한 CloudWatch 로그 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

내용

Amazon CloudWatch 로그의 데이터 보호

> **Note**
>
> Logs를 사용하면 일반적인 데이터 보호에 대한 다음 정보 외에도 CloudWatch 로그 이벤트의 민감한 데이터를 마스킹하여 보호할 수 있습니다. AWS자세한 정보는 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원을 참조하세요.

AWS 공동 책임 모델 Amazon CloudWatch Logs의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 은 (는) 모두를 실행하는 글로벌 인프라를 보호할 책임이 AWS 클라우드있습니다. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스 의 보안 구성과 관리 작업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은AWS 보안 블로그에서 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 AWS 계정 자격 증명을 보호하고 AWS IAM Identity Center OR AWS Identity and Access Management (IAM) 을 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 멀티 팩터 인증 설정(MFA)을 사용하세요.
- SSL/TLS를 사용하여 리소스와 통신하세요. AWS TLS 1.2는 필수이며 TLS 1.3를 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다. AWS CloudTrail
- 포함된 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용하십시오 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 AWS 통해 액세스할 때 FIPS 140-2로 검증된 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용하십시오. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 FIPS(Federal Information Processing Standard) 140-2를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔, API 또는 AWS 서비스 SDK를 사용하여 CloudWatch 로그 또는 기타 작업을 하는 경우가 포함됩니다. AWS CLI AWS 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

저장된 데이터 암호화

CloudWatch 로그는 암호화를 사용하여 저장된 데이터를 보호합니다. 모든 로그 그룹이 암호화됩니다. 기본적으로 CloudWatch 로그 서비스는 서버 측 암호화 키를 관리합니다.

로그 암호화 및 암호 해독에 사용되는 키를 관리하려면 키를 사용하십시오. AWS KMS 자세한 정보는 다음을 사용하여 Logs의 로그 데이터를 암호화합니다. CloudWatch AWS Key Management Service을 참조하세요.

전송 중 데이터 암호화

CloudWatch 로그는 end-to-end 전송 중인 데이터의 암호화를 사용합니다. CloudWatch 로그 서비스는 서버 측 암호화 키를 관리합니다.

Amazon CloudWatch Logs의 자격 증명 및 액세스 관리

Amazon CloudWatch Logs에 액세스하려면 요청을 인증하는 데 사용할 AWS 수 있는 자격 증명이 필요합니다. 이러한 자격 증명에는 클라우드 AWS 리소스에 대한 CloudWatch 로그 데이터 검색 등 리소스에 액세스할 수 있는 권한이 있어야 합니다. 다음 섹션에서는 액세스 가능한 사용자를 제어하여 리소스를 보호하는 데 도움이 되는 AWS Identity and Access Management (IAM) 및 CloudWatch 로그를 사용하는 방법에 대한 세부 정보를 제공합니다.

- 인증
- 액세스 제어

인증

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- 내 AWS IAM Identity Center사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르십시오.

- 보안 인증 공급자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르십시오.

- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르십시오.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르십시오.

액세스 제어

요청을 인증하는 데 필요한 유효한 자격 증명을 가질 수 있지만 권한이 없으면 CloudWatch 로그 리소스를 생성하거나 액세스할 수 없습니다. 예를 들어 로그 스트림과 로그 그룹 생성 등의 권한이 있어야 합니다.

다음 섹션에서는 CloudWatch 로그에 대한 권한을 관리하는 방법을 설명합니다. 먼저 개요를 읽어 보면 도움이 됩니다.

- CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요
- 로그에 ID 기반 정책 (IAM 정책) 사용 CloudWatch
- CloudWatch 로그 권한 참조

CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center다음 지역의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르십시오.
- 보안 인증 공급자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르십시오.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르십시오.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르십시오.

주제

CloudWatch 리소스 및 작업을 기록합니다.

CloudWatch 로그에서 기본 리소스는 로그 그룹, 로그 스트림 및 대상입니다. CloudWatch 로그는 하위 리소스 (기본 리소스와 함께 사용할 다른 리소스) 를 지원하지 않습니다.

다음 표에 나와 있는 것처럼 이러한 리소스와 하위 리소스에는 고유한 Amazon 리소스 이름(ARN)이 연결되어 있습니다.

리소스 유형	ARN 형식
로그 그룹	다음 두 가지가 모두 사용됩니다. 끝에 가 있는 두 번째 항목은 `describe-log-groups` CLI 명령과 `:*` API에서 반환되는 값입니다. DescribeLogGroups arn:aws:logs:*region*:*account-id* :log-group:*log_group_name* arn:aws:logs:*region*:*account-id* :log-group:*log_group_name* :* 다음과 같은 상황에서는 후행 `:*` 없이 첫 번째 버전을 사용하십시오. • 여러 CloudWatch Logs API의 `logGroupIdentifier` 입력 필드에. • API 태깅 `resourceArn` 분야에서 • IAM 정책에서, 및 에 대한 TagResource권한을 지정할 때 UntagResourceListTagsForResource

리소스 유형	ARN 형식
	다른 모든 API 작업에 대한 IAM 정책에서 권한을 지정할 때 ARN을 참조하려면 두 번째 버전 (후행 :* 포함) 을 사용하십시오.
로그 스트림	*arn:aws:logs: ##: ## ID:## ##: log_group_name:log-stream: log-stream-name*
대상	arn:aws:logs:*region*:*account-id* :destination:*destination_name*

ARN에 대한 자세한 내용은 IAM 사용 설명서의 ARN을 참조하세요. CloudWatch 로그 ARN에 대한 자세한 내용은 의 Amazon 리소스 이름 (ARN) 을 참조하십시오. Amazon Web Services 일반 참조 CloudWatch 로그를 다루는 정책의 예는 을 참조하십시오. 로그에 ID 기반 정책 (IAM 정책) 사용 CloudWatch

CloudWatch 로그는 CloudWatch 로그 리소스로 작업하기 위한 일련의 작업을 제공합니다. 사용 가능한 작업 목록은 CloudWatch 로그 권한 참조 섹션을 참조하십시오.

리소스 소유권 이해

리소스를 만든 사람이 누구인지와 상관없이 계정에서 생성된 리소스는 계정을 소유합니다. AWS 구체적으로, 리소스 소유자는 리소스 생성 요청을 인증하는 보안 주체 (즉, 루트 계정, 사용자 또는 IAM 역할) 의 계정입니다. AWS 다음 예에서는 이러한 작동 방식을 설명합니다.

- 계정의 루트 계정 자격 증명을 사용하여 로그 그룹을 생성하는 경우 해당 AWS 계정이 CloudWatch Logs 리소스의 소유자가 됩니다. AWS
- AWS 계정에서 사용자를 생성하고 해당 사용자에게 CloudWatch 로그 리소스를 생성할 권한을 부여하면 사용자가 CloudWatch 로그 리소스를 생성할 수 있습니다. 하지만 사용자가 속한 AWS 계정이 CloudWatch 로그 리소스를 소유합니다.
- AWS 계정에 로그 리소스를 생성할 권한이 있는 IAM 역할을 생성하는 경우 해당 역할을 수임할 수 있는 사람은 누구나 CloudWatch CloudWatch 로그 리소스를 생성할 수 있습니다. 역할이 속한 AWS 계정이 CloudWatch Logs 리소스를 소유합니다.

리소스 액세스 관리

권한 정책은 누가 무엇에 액세스할 수 있는지를 나타냅니다. 다음 섹션에서는 권한 정책을 만드는 데 사용 가능한 옵션에 대해 설명합니다.

> **Note**
>
> 이 섹션에서는 로그 컨텍스트에서의 IAM을 사용하는 방법에 대해 설명합니다. CloudWatch IAM 서비스에 대한 자세한 정보는 다루지 않습니다. IAM 설명서 전체 내용은 IAM 사용 설명서의 IAM이란 무엇인가요? 단원을 참조하세요. IAM 정책 구문 및 설명에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 참조 단원을 참조하세요.

IAM ID에 연결된 정책을 ID 기반 정책 (IAM 정책) 이라고 하고 리소스에 연결된 정책을 리소스 기반 정책이라고 합니다. CloudWatch 로그는 ID 기반 정책과 대상 리소스 기반 정책을 지원하며, 이 정책은 교차 계정 구독을 활성화하는 데 사용됩니다. 자세한 정보는 계정 간 지역 간 구독을 참조하세요.

주제

- 로그 그룹 권한 및 Contributor Insights
- 리소스 기반 정책

로그 그룹 권한 및 Contributor Insights

Contributor Insights는 로그 그룹의 데이터를 분석하고 기여자 데이터를 표시하는 시계열을 생성할 수 있는 기능입니다. CloudWatch 상위 N개의 기고자, 총 고유 기고자 수 및 사용량에 대한 지표를 볼 수 있습니다. 자세한 내용은 Contributor Insights를 사용하여 높은 카디널리티 데이터 분석을 참조하세요.

사용자에게 `cloudwatch:PutInsightRule` 및 `cloudwatch:GetInsightRuleReport` 권한을 부여하면 해당 사용자는 Logs의 모든 로그 그룹을 평가한 다음 결과를 확인하는 규칙을 만들 수 있습니다. CloudWatch 결과에는 이러한 로그 그룹에 대한 기여자 데이터가 포함될 수 있습니다. 이 데이터를 볼 수 있어야 하는 사용자에게만 해당 권한을 부여해야 합니다.

리소스 기반 정책

CloudWatch 로그는 대상에 대한 리소스 기반 정책을 지원하며, 이를 사용하여 교차 계정 구독을 활성화할 수 있습니다. 자세한 정보는 1단계: 대상 생성을 참조하세요. 대상은 API를 사용하여 만들 수 있으며, PutDestinationAPI를 사용하여 대상에 리소스 정책을 추가할 수 있습니다. PutDestinationPolicy 다음 예에서는 AWS 계정 ID가 111122223333인 다른

계정이 자신의 로그 그룹을 대상으로 구독할 수 있도록 허용합니다. `arn:aws:logs:us-east-1:123456789012:destination:testDestination`

```
{
  "Version" : "2012-10-17",
  "Statement" : [
    {
      "Sid" : "",
      "Effect" : "Allow",
      "Principal" : {
        "AWS" : "111122223333"
      },
      "Action" : "logs:PutSubscriptionFilter",
      "Resource" : "arn:aws:logs:us-east-1:123456789012:destination:testDestination"
    }
  ]
}
```

정책 요소 지정: 작업, 효과, 보안 주체

각 CloudWatch 로그 리소스에 대해 서비스는 일련의 API 작업을 정의합니다. 이러한 API 작업에 대한 권한을 부여하기 위해 CloudWatch 로그는 정책에서 지정할 수 있는 일련의 작업을 정의합니다. 일부 API 작업에서는 API 작업을 수행하기 위해 복수의 작업에 대한 권한이 필요할 수 있습니다. 리소스 및 API 작업에 대한 자세한 내용은 CloudWatch 리소스 및 작업을 기록합니다. 및 CloudWatch 로그 권한 참조 섹션을 참조하세요.

다음은 기본 정책 요소입니다.

- 리소스 – Amazon 리소스 이름(ARN)을 사용하여 정책을 적용할 리소스를 식별합니다. 자세한 정보는 CloudWatch 리소스 및 작업을 기록합니다. 섹션을 참조하세요.
- 조치 – 조치 키워드를 사용하여 허용 또는 거부할 리소스 작업을 식별합니다. 예를 들어, `logs.DescribeLogGroups` 권한은 사용자에게 `DescribeLogGroups` 작업 수행 권한을 허용합니다.
- Effect - 사용자가 특정 작업을 요청할 때 허용할지 아니면 거부할지 그 결과를 지정합니다. 명시적으로 리소스에 대한 액세스 권한을 부여(허용)하지 않는 경우, 액세스는 묵시적으로 거부됩니다. 다른 정책에서 액세스 권한을 부여하는 경우라도 사용자가 해당 리소스에 액세스할 수 없도록 하기 위해 리소스에 대한 권한을 명시적으로 거부할 수도 있습니다.

- 보안 주체 – ID 기반 정책(IAM 정책)에서 정책이 연결되는 사용자는 암시적인 보안 주체입니다. 리소스 기반 정책의 경우 권한을 받을 사용자, 계정, 서비스 또는 기타 엔티티를 지정합니다 (리소스 기반 정책에만 적용). CloudWatch 로그는 대상에 대한 리소스 기반 정책을 지원합니다.

IAM 정책 구문과 설명에 대한 자세한 내용은 IAM 사용 설명서의 AWS IAM 정책 참조를 참조하십시오.

모든 CloudWatch Logs API 작업과 해당 작업이 적용되는 리소스를 보여주는 표는 을 참조하십시오. CloudWatch 로그 권한 참조

정책에서 조건 지정

권한을 부여할 때 액세스 정책 언어를 사용하여 조건이 적용되는 조건을 지정할 수 있습니다. 예를 들어, 특정 날짜 이후에만 정책을 적용할 수 있습니다. 정책 언어에서의 조건 지정에 관한 자세한 내용은 IAM 사용 설명서의 조건을 참조하십시오.

조건을 표시하려면 미리 정의된 조건 키를 사용합니다. 각 AWS 서비스에서 지원하는 컨텍스트 키 목록과 AWS-wide 정책 키 목록은 AWS 서비스의 작업, 리소스, 조건 키 및AWS 글로벌 조건 컨텍스트 키를 참조하십시오.

> **Note**
>
> 태그를 사용하여 로그 그룹 및 대상을 포함한 CloudWatch 로그 리소스에 대한 액세스를 제어할 수 있습니다. 로그 그룹과 로그 스트림 간의 계층적 관계로 인해 로그 스트림에 대한 액세스는 로그 그룹 수준에서 제어됩니다. 태그를 사용하여 액세스를 제어하는 방법에 대한 자세한 내용은 태그를 사용하여 Amazon Web Services 리소스에 대한 액세스 제어를 참조하세요.

로그에 ID 기반 정책 (IAM 정책) 사용 CloudWatch

이 항목에서는 계정 관리자가 IAM ID(사용자, 그룹, 역할)에 권한 정책을 연결할 수 있는 ID 기반 정책의 예를 제공합니다.

> **Important**
>
> 먼저 로그 리소스에 대한 액세스를 관리하는 데 사용할 수 있는 기본 개념과 옵션을 설명하는 소개 주제를 검토하는 것이 좋습니다. CloudWatch 자세한 정보는 CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요을 참조하세요.

이 주제는 다음을 다룹니다.

- CloudWatch 콘솔을 사용하는 데 필요한 권한
- AWS 로그에 대한 CloudWatch 관리형 (사전 정의된) 정책
- 고객 관리형 정책 예

다음은 권한 정책의 예제입니다.

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

이 정책에는 로그 그룹 및 로그 스트림을 생성하고 로그 스트림에 로그 이벤트를 업로드하며 로그 스트림에 대한 세부 사항을 나열할 수 있는 권한을 부여하는 명령문이 하나 포함되어 있습니다.

`Resource` 값 끝에 와일드카드 문자(*)가 있다는 것은 이 정책 명령문이 어떤 로그 그룹에서든 `logs:CreateLogGroup`, `logs:CreateLogStream`, `logs:PutLogEvents` 및 `logs:DescribeLogStreams` 작업에 대한 권한을 허용한다는 의미입니다. 이러한 권한을 특정 로그 그룹으로 제한하려면 리소스 ARN에 있는 와일드카드 문자(*)를 특정 로그 그룹 ARN으로 대체합니다. IAM 정책 설명 내 섹션에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 요소 참조를 참조하세요. 모든 CloudWatch 로그 작업을 보여주는 목록은 을 참조하십시오CloudWatch 로그 권한 참조.

CloudWatch 콘솔을 사용하는 데 필요한 권한

사용자가 CloudWatch 콘솔에서 CloudWatch 로그를 사용하려면 해당 사용자에게 AWS 계정의 다른 AWS 리소스를 설명할 수 있는 최소한의 권한이 있어야 합니다. CloudWatch 콘솔에서 CloudWatch 로그를 사용하려면 다음 서비스의 권한이 있어야 합니다.

- CloudWatch
- CloudWatch 로그
- OpenSearch 서비스
- IAM
- Kinesis
- Lambda
- Amazon S3

최소 필수 권한보다 더 제한적인 IAM 정책을 만들면 콘솔은 해당 IAM 정책에 연결된 사용자에 대해 의도대로 작동하지 않습니다. 이러한 사용자가 CloudWatch 콘솔을 계속 사용할 수 있도록 하려면 에 설명된 대로 `CloudWatchReadOnlyAccess` 관리형 정책도 사용자에게 AWS 로그에 대한 CloudWatch 관리형 (사전 정의된) 정책 연결하세요.

AWS CLI 또는 CloudWatch Logs API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요는 없습니다.

콘솔을 사용하여 로그 구독을 관리하지 않는 사용자가 CloudWatch 콘솔을 사용하는 데 필요한 전체 권한은 다음과 같습니다.

- 클라우드워치: GetMetricData
- 클라우드워치: ListMetrics
- 로그: CancelExportTask
- 로그: CreateExportTask
- 로그: CreateLogGroup
- 로그: CreateLogStream
- 로그: DeleteLogGroup
- 로그: DeleteLogStream
- 로그: DeleteMetricFilter
- 로그: DeleteQueryDefinition

- 로그: DeleteRetentionPolicy
- 로그: DeleteSubscriptionFilter
- 로그: DescribeExportTasks
- 로그: DescribeLogGroups
- 로그: DescribeLogStreams
- 로그: DescribeMetricFilters
- 로그: DescribeQueryDefinitions
- 로그: DescribeQueries
- 로그: DescribeSubscriptionFilters
- 로그: FilterLogEvents
- 로그: GetLogEvents
- 로그: GetLogGroupFields
- 로그: GetLogRecord
- 로그: GetQueryResults
- 로그: PutMetricFilter
- 로그: PutQueryDefinition
- 로그: PutRetentionPolicy
- 로그: StartQuery
- 로그: StopQuery
- 로그: PutSubscriptionFilter
- 로그: TestMetricFilter

사용자가 콘솔을 사용하여 로그 구독도 관리할 경우 다음 권한도 필요합니다.

- 예: DescribeElasticsearchDomain
- 예: ListDomainNames
- 목표: AttachRolePolicy
- 목표: CreateRole
- 목표: GetPolicy
- 목표: GetPolicyVersion
- 목표: GetRole

- 목표: ListAttachedRolePolicies
- 목표: ListRoles
- 키네시스: DescribeStreams
- 키네시스: ListStreams
- 람다: AddPermission
- 람다: CreateFunction
- 람다: GetFunctionConfiguration
- 람다: ListAliases
- 람다: ListFunctions
- 람다: ListVersionsByFunction
- 람다: RemovePermission
- s3: ListBuckets

AWS 로그에 대한 CloudWatch 관리형 (사전 정의된) 정책

AWS 에서 생성하고 관리하는 독립형 IAM 정책을 제공하여 많은 일반적인 사용 사례를 해결합니다. AWS관리형 정책은 사용자가 필요한 권한을 조사할 필요가 없도록 일반 사용 사례에 필요한 권한을 부여합니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하세요.

계정의 사용자 및 역할에 연결할 수 있는 다음과 같은 AWS 관리형 정책은 로그에만 해당됩니다. CloudWatch

- CloudWatchLogsFullAccess— CloudWatch 로그에 대한 전체 액세스 권한을 부여합니다.
- CloudWatchLogsReadOnlyAccess— CloudWatch 로그에 대한 읽기 전용 액세스 권한을 부여합니다.

CloudWatchLogsFullAccess

이 CloudWatchLogsFullAccess정책은 CloudWatch 로그에 대한 전체 액세스 권한을 부여합니다. 정책에는 `cloudwatch:GenerateQuery` 권한이 포함되므로 이 정책을 사용하는 사용자는 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있습니다. 내용은 다음과 같습니다.

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

```
            "Action": [
                "logs:*",
                "cloudwatch:GenerateQuery"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

CloudWatchLogsReadOnlyAccess

CloudWatchLogsReadOnlyAccess정책은 CloudWatch 로그에 대한 읽기 전용 액세스 권한을 부여합니다. 여기에는 이 정책을 사용하는 사용자가 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있는 `cloudwatch:GenerateQuery` 권한이 포함됩니다. 내용은 다음과 같습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:Describe*",
                "logs:Get*",
                "logs:List*",
                "logs:StartQuery",
                "logs:StopQuery",
                "logs:TestMetricFilter",
                "logs:FilterLogEvents",
                "logs:StartLiveTail",
                "logs:StopLiveTail",
                "cloudwatch:GenerateQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

CloudWatchLogsCrossAccountSharingConfiguration

CloudWatchLogsCrossAccountSharingConfiguration정책은 계정 간에 CloudWatch 로그 리소스를 공유하기 위한 Observability Access Manager 링크를 만들고, 관리하고, 볼 수 있는 액세스 권한을 부여합니다. 자세한 내용은 CloudWatch 계정 간 옵저버빌리티를 참조하십시오.

내용은 다음과 같습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:Link",
                "oam:ListLinks"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:DeleteLink",
                "oam:GetLink",
                "oam:TagResource"
            ],
            "Resource": "arn:aws:oam:*:*:link/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:CreateLink",
                "oam:UpdateLink"
            ],
            "Resource": [
                "arn:aws:oam:*:*:link/*",
                "arn:aws:oam:*:*:sink/*"
            ]
        }
    ]
}
```

CloudWatch 관리형 정책에 대한 업데이트를 기록합니다. AWS

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후 CloudWatch Logs의 AWS 관리형 정책 업데이트에 대한 세부 정보를 볼 수 있습니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 CloudWatch 로그 문서 기록 페이지에서 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
CloudWatchLogsFullAccess - 기존 정책에 대한 업데이트	CloudWatch 로그에 CloudWatchLogsFullAccess권한이 추가되었습니다. 이 정책을 사용하는 사용자가 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있도록 `cloudwatch:GenerateQuery` 권한이 추가되었습니다.	2023년 11월 27일
CloudWatchLogsReadOnlyAccess - 기존 정책에 대한 업데이트	CloudWatch 에 권한을 추가했습니다 CloudWatchLogsReadOnlyAccess. 이 정책을 사용하는 사용자가 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있도록 `cloudwatch:GenerateQuery` 권한이 추가되었습니다.	2023년 11월 27일
CloudWatchLogsReadOnlyAccess-기존 정책 업데이트	CloudWatch 로그에 권한이 추가되었습니다 CloudWatchLogsReadOnlyAccess. 이 정책을 사용하는 사용자가 콘솔을 사용하여 CloudWatch Logs live tail 세션을 시작 및 중지할 수 있도록 및 `logs:StopLiveTail` 권한이 추가되었습니다. `logs:Star`	2023년 6월 6일

변경 사항	설명	날짜
	`tLiveTail` 자세한 내용은 Live Tail을 사용하여 거의 실시간으로 로그 보기를 참조하세요.	
CloudWatchLogsCrossAccountSharingConfiguration - 새 정책	CloudWatch 로그에는 로그 로그 그룹을 공유하는 CloudWatch CloudWatch 계정 간 가시성 링크를 관리할 수 있는 새 정책이 추가되었습니다. 자세한 내용은 계정 간 옵저버빌리티를 참조하십시오CloudWatch .	2022년 11월 27일
CloudWatchLogsReadOnlyAccess-기존 정책 업데이트	CloudWatch 로그에 추가된 권한. CloudWatchLogsReadOnlyAccess `oam:ListSinks` 및 `oam:ListAttachedLinks` 권한이 추가되었으므로 이 정책을 사용하는 사용자가 콘솔을 사용하여 계정 CloudWatch 간 관찰 가능성에서 소스 계정에서 공유된 데이터를 볼 수 있습니다.	2022년 11월 27일

고객 관리형 정책 예

사용자 지정 IAM 정책을 생성하여 CloudWatch Logs 작업 및 리소스에 대한 권한을 허용할 수 있습니다. 해당 권한이 필요한 사용자 또는 그룹에 이러한 사용자 지정 정책을 연결할 수 있습니다.

이 섹션에서는 다양한 CloudWatch 로그 작업에 대한 권한을 부여하는 예제 사용자 정책을 확인할 수 있습니다. 이러한 정책은 CloudWatch Logs API, AWS SDK 또는 를 사용할 때 작동합니다. AWS CLI

예

- 예 1: 로그에 CloudWatch 대한 전체 액세스 허용
- 예 2: 로그에 CloudWatch 대한 읽기 전용 액세스 허용
- 예제 3: 로그 그룹 하나에 대한 액세스 허용

예 1: 로그에 CloudWatch 대한 전체 액세스 허용

다음 정책은 사용자가 모든 CloudWatch 로그 작업에 액세스할 수 있도록 허용합니다.

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

예 2: 로그에 CloudWatch 대한 읽기 전용 액세스 허용

AWS CloudWatch 로그 데이터에 대한 읽기 전용 액세스를 허용하는 CloudWatchLogsReadOnlyAccess정책을 제공합니다. 이 정책에는 다음 권한이 포함되어 있습니다.

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

예제 3: 로그 그룹 하나에 대한 액세스 허용

다음 정책에서는 사용자가 지정된 로그 그룹 하나에서 로그 이벤트를 읽고 쓸 수 있게 허용합니다.

> **Important**
>
> Resource 라인의 로그 그룹 이름 끝에 있는 :* 기호가 있어야 정책이 이 로그 그룹의 모든 로그 스트림에 적용됨을 나타냅니다. :* 기호를 생략하는 경우 정책이 시행되지 않습니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
      "Action": [
        "logs:CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:PutLogEvents",
        "logs:GetLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:us-west-2:123456789012:log-group:SampleLogGroupName:*"
      }
   ]
}
```

로그 그룹 수준의 제어를 위한 태깅 및 IAM 정책 사용

사용자에게 특정 로그 그룹에 대한 액세스 권한을 부여함과 동시에 다른 로그 그룹에 대한 액세스를 방지할 수 있습니다. 이렇게 하려면 로그 그룹에 태그를 지정하고 해당 태그를 참조하는 IAM 정책을 사용하세요. 로그 그룹에 태그를 적용하려면 logs:TagResource 또는 logs:TagLogGroup 권한이 있어야 합니다. 이는 로그 그룹을 생성할 때 로그 그룹에 태그를 할당하거나 나중에 할당하는 경우 모두에 적용됩니다.

로그 그룹의 태깅에 대한 자세한 내용은 Amazon Logs의 태그 CloudWatch 로그 그룹 섹션을 참조하세요.

로그 그룹에 태그를 지정할 때 특정 태그가 있는 로그 그룹에만 액세스를 허용할 수 있도록 사용자에게 IAM 정책에 대한 권한을 부여할 수 있습니다. 예를 들어, 다음 정책 명령문은 태그 키 Team에 대한 값이 Green인 로그 그룹에만 액세스하도록 권한을 부여합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:*"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/Team": "Green"
                }
            }
        }
    ]
}
```

StopQuery및 StopLiveTailAPI 작업은 전통적인 의미에서는 AWS 리소스와 상호 작용하지 않습니다. 즉, 어떤 식으로든 데이터를 반환하거나 데이터를 넣거나 리소스를 수정하지 않습니다. 대신 리소스로 분류되지 않는 지정된 라이브 테일 세션 또는 지정된 CloudWatch Logs Insights 쿼리에서만 작동합니다. 따라서 이러한 작업에 대해 IAM 정책에서 Resource 필드를 지정하는 경우 다음 예제와 같이 Resource 필드 값을 *로 설정해야 합니다.

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

```
        ]
}
```

IAM 정책에 자세한 내용은 IAM 사용 설명서에서 정책을 사용하여 액세스 제어를 참조하세요.

CloudWatch 로그 권한 참조

아래에 있는 표를 참조하여 IAM 자격 증명에 연결할 수 있는 액세스 제어 및 쓰기 권한 정책(자격 증명 기반 정책)을 설정할 수 있습니다. 표에는 각 CloudWatch Logs API 작업과 해당 작업을 수행할 권한을 부여할 수 있는 해당 작업이 나열되어 있습니다. 정책의 `Action` 필드에 작업을 지정합니다. `Resource`필드에 로그 그룹 또는 로그 스트림의 ARN을 지정하거나 모든 CloudWatch 로그 리소스를 `*` 나타내도록 지정할 수 있습니다.

CloudWatch 로그 정책에서 AWS-wide 조건 키를 사용하여 조건을 표현할 수 있습니다. AWS-wide 키의 전체 목록은 IAM 사용 설명서의 AWS 글로벌 및 IAM 조건 컨텍스트 키를 참조하십시오.

> **Note**
>
> 작업을 지정하려면 `logs:` 접두사 다음에 API 작업 이름을 사용합니다. 예: `logs:CreateLogGrouplogs:CreateLogStream`, 또는 `logs:*` (모든 CloudWatch 로그 작업의 경우).

CloudWatch 로그 API 작업 및 작업에 필요한 권한

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
CancelExportTask	`logs:CancelExportTask` 보류 또는 실행 중인 내보내기 작업을 취소하는 데 필요합니다.
CreateExportTask	`logs:CreateExportTask` 로그 그룹에서 Amazon S3 버킷으로 데이터를 내보내는 데 필요합니다.
CreateLogGroup	`logs:CreateLogGroup` 새 보안 그룹을 생성하는 데 필요합니다.

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
CreateLogStream	`logs:CreateLogStream` 로그 그룹에서 로그 스트림을 새로 생성하는 데 필요합니다.
DeleteDestination	`logs:DeleteDestination` 로그 대상을 삭제하고 이에 대한 모든 구독 필터를 비활성화하는 데 필요합니다.
DeleteLogGroup	`logs:DeleteLogGroup` 로그 그룹과 보관되는 모든 연관 로그 이벤트를 삭제하는 데 필요합니다.
DeleteLogStream	`logs:DeleteLogStream` 로그 스트림과 보관되는 모든 연관 로그 이벤트를 삭제하는 데 필요합니다.
DeleteMetricFilter	`logs:DeleteMetricFilter` 로그 그룹과 연관된 지표 필터를 삭제하는 데 필요합니다.
DeleteQueryDefinition	`logs:DeleteQueryDefinition` CloudWatch Logs Insights에서 저장된 쿼리 정의를 삭제하는 데 필요합니다.
DeleteResourcePolicy	`logs:DeleteResourcePolicy` CloudWatch 로그 리소스 정책을 삭제하는 데 필요합니다.
DeleteRetentionPolicy	`logs:DeleteRetentionPolicy` 로그 그룹의 보존 정책을 삭제하는 데 필요합니다.

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
DeleteSubscriptionFilter	`logs:DeleteSubscriptionFilter` 로그 그룹과 연관된 구독 필터를 삭제하는 데 필요합니다.
DescribeDestinations	`logs:DescribeDestinations` 계정과 연결된 모든 대상을 보는 데 필요합니다.
DescribeExportTasks	`logs:DescribeExportTasks` 계정과 연관된 모든 내보내기 작업을 보는 데 필요합니다.
DescribeLogGroups	`logs:DescribeLogGroups` 계정과 연관된 모든 로그 그룹들을 보는 데 필요합니다.
DescribeLogStreams	`logs:DescribeLogStreams` 로그 그룹과 연관된 모든 로그 스트림을 보는 데 필요합니다.
DescribeMetricFilters	`logs:DescribeMetricFilters` 로그 그룹과 연관된 모든 지표를 보는 데 필요합니다.
DescribeQueryDefinitions	`logs:DescribeQueryDefinitions` CloudWatch Logs Insights에서 저장된 쿼리 정의 목록을 보는 데 필요합니다.
DescribeQueries	`logs:DescribeQueries` 예약되어 있거나, 실행 중이거나, 최근에 실행된 CloudWatch Logs Insights 쿼리의 목록을 보는 데 필요합니다.

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
DescribeResourcePolicies	`logs:DescribeResourcePolicies` CloudWatch Logs 리소스 정책 목록을 보는 데 필요합니다.
DescribeSubscriptionFilters	`logs:DescribeSubscriptionFilters` 로그 그룹과 연관된 모든 구독 필터를 보는 데 필요합니다.
FilterLogEvents	`logs:FilterLogEvents` 로그 그룹 필터 패턴에 따라 로그 이벤트를 정렬하는 데 필요합니다.
GetLogEvents	`logs:GetLogEvents` 로그 스트림에서 로그 이벤트를 검색하는 데 필요합니다.
GetLogGroupFields	`logs:GetLogGroupFields` 로그 그룹의 로그 이벤트에 포함된 필드 목록을 검색하는 데 필요합니다.
GetLogRecord	`logs:GetLogRecord` 단일 로그 이벤트에서 세부 정보를 검색하는 데 필요합니다.
GetQueryResults	`logs:GetQueryResults` CloudWatch Logs Insights 쿼리 결과를 검색하는 데 필요합니다.
ListTagsLogGroup	`logs:ListTagsLogGroup` 로그 그룹과 연결된 태그를 나열하는 데 필요합니다.

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
PutDestination	`logs:PutDestination` 대상 로그 스트림(예: Kinesis 스트림)을 생성하거나 업데이트하는 데 필요합니다.
PutDestinationPolicy	`logs:PutDestinationPolicy` 기존 로그 대상과 연관된 액세스 정책을 생성 또는 업데이트하는 데 필요합니다.
PutLogEvents	`logs:PutLogEvents` 로그 스트림에서 로그 이벤트 배치를 업로드하는 데 필요합니다.
PutMetricFilter	`logs:PutMetricFilter` 지표 필터를 생성 또는 업데이트하고 이를 로그 그룹과 연관시키는 데 필요합니다.
PutQueryDefinition	`logs:PutQueryDefinition` CloudWatch Logs Insights에 쿼리를 저장하는 데 필요합니다.
PutResourcePolicy	`logs:PutResourcePolicy` CloudWatch 로그 리소스 정책을 생성하는 데 필요합니다.
PutRetentionPolicy	`logs:PutRetentionPolicy` 로그 그룹에 로그 이벤트를 유지하는 일수(보존 일수)를 설정하는 데 필요합니다.
PutSubscriptionFilter	`logs:PutSubscriptionFilter` 구독 필터를 생성 또는 업데이트하고 이를 로그 그룹과 연관시키는 데 필요합니다.

CloudWatch API 작업을 기록합니다.	필요한 권한(API 작업)
StartQuery	`logs:StartQuery` CloudWatch 로그 인사이트 쿼리를 시작하는 데 필요합니다.
StopQuery	`logs:StopQuery` 진행 중인 CloudWatch 로그 인사이트 쿼리를 중지하는 데 필요합니다.
TagLogGroup	`logs:TagLogGroup` 로그 그룹 태그를 추가하거나 업데이트하는 데 필요합니다.
TestMetricFilter	`logs:TestMetricFilter` 로그 이벤트 메시지 샘플을 기준으로 필터 패턴을 테스트하는 데 필요합니다.

로그에 서비스 연결 역할 사용 CloudWatch

Amazon CloudWatch Logs는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 로그에 직접 연결되는 고유한 유형의 IAM 역할입니다. CloudWatch 서비스 연결 역할은 CloudWatch 로그에 의해 미리 정의되며 서비스가 사용자를 대신하여 다른 서비스를 호출하는 데 필요한 모든 권한을 포함합니다. AWS

서비스 연결 역할을 사용하면 필요한 권한을 수동으로 추가할 필요가 없으므로 CloudWatch 로그를 보다 효율적으로 설정할 수 있습니다. CloudWatch 로그는 서비스 연결 역할의 권한을 정의하며, 달리 정의되지 않는 한 CloudWatch 로그만 해당 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함됩니다. 이 권한 정책은 다른 어떤 IAM 엔터티에도 연결할 수 없습니다.

서비스 연결 역할을 지원하는 다른 서비스에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스를 참조하세요. 서비스 연결 역할 열에 예가 있는 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

로그에 대한 서비스 연결 역할 권한 CloudWatch

CloudWatch 로그는 이름이 지정된 서비스 연결 역할을 사용합니다. AWSServiceRoleForLogDelivery CloudWatch Logs는 이 서비스 연결 역할을 사용하여 Firehose에 직접 로그를 기록합니다. 자세한 정보는 AWS 서비스에서 로깅 활성화을 참조하세요.

AWSServiceRoleForLogDelivery 서비스 연결 역할은 역할을 위임하기 위해 다음 서비스를 신뢰합니다.

- `logs.amazonaws.com`

역할 권한 정책에 따라 CloudWatch 로그는 지정된 리소스에서 다음 작업을 완료할 수 있습니다.

- 조치: `firehose:PutRecord` 그리고 `firehose:PutRecordBatch` 값이 인 `LogDeliveryEnabled` 키가 있는 태그가 있는 모든 Firehose 스트림에서 `True` 이 태그는 Firehose에 로그를 전송하기 위한 구독을 만들 때 Firehose 스트림에 자동으로 연결됩니다.

IAM 엔터티가 서비스 연결 역할을 생성, 편집 또는 삭제할 수 있도록 권한을 구성해야 합니다. 이 엔터티는 사용자, 그룹 또는 역할일 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

로그에 대한 서비스 연결 역할 생성 CloudWatch

서비스 연결 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console AWS CLI, 또는 AWS API에서 Firehose 스트림으로 직접 전송되도록 CloudWatch 로그를 설정하면 Logs가 서비스 연결 역할을 자동으로 생성합니다.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. Firehose 스트림으로 직접 전송되도록 로그를 다시 설정하면 CloudWatch Logs에서 서비스 연결 역할을 다시 생성합니다.

로그의 서비스 연결 역할 편집 CloudWatch

CloudWatch 로그에서는 역할을 만든 후에는 AWSServiceRoleForLogDelivery편집이나 다른 서비스 연결 역할을 사용할 수 없습니다. 다양한 엔터티티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

로그의 서비스 연결 역할 삭제 CloudWatch

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 단, 서비스 링크 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려고 할 때 CloudWatch 로그 서비스가 역할을 사용하는 경우 삭제가 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForLogDelivery서비스 연결 역할에서 사용하는 CloudWatch 로그 리소스를 삭제하려면

- Firehose 스트림에 직접 로그를 보내는 것을 중지하세요.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForLogDelivery서비스 연결 역할을 삭제하세요. 자세한 내용은 서비스 연결 역할 삭제를 참조하세요.

로그 지원 지역: 서비스 연결 역할 CloudWatch

CloudWatch 로그는 서비스를 사용할 수 있는 모든 AWS 지역에서 서비스 연결 역할을 사용할 수 있도록 지원합니다. 자세한 내용은 CloudWatch 로그 지역 및 엔드포인트를 참조하십시오.

Amazon CloudWatch 로그의 규정 준수 검증

타사 감사자는 여러 규정 AWS 준수 프로그램의 일환으로 Amazon CloudWatch Logs의 보안 및 규정 준수를 평가합니다. 여기에는 SOC, PCI, FedRAMP, HIPAA 등이 포함됩니다.

특정 규정 준수 프로그램 범위 내 AWS 서비스 목록은 규정 준수 프로그램별 범위 내AWS 서비스 규정 준수 참조하십시오. 일반적인 정보는 AWS 규정 준수 프로그램을 참조하십시오.

를 사용하여 타사 감사 보고서를 다운로드할 수 AWS Artifact있습니다. 자세한 내용은 의 보고서 https://docs.aws.amazon.com/artifact/latest/ug/downloading-documents.html 참조하십시오 AWS Artifact.

Amazon CloudWatch Logs를 사용할 때의 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률 및 규정에 따라 결정됩니다. AWS 규정 준수에 도움이 되는 다음 리소스를 제공합니다.

- 보안 및 규정 준수 빠른 시작 안내서: 이 배포 안내서에서는 아키텍처 고려 사항에 관해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- Amazon Web Services의 HIPAA 보안 및 규정 준수를 위한 설계 — 이 백서에서는 기업이 HIPAA 준수 애플리케이션을 만드는 AWS 데 사용할 수 있는 방법을 설명합니다.
- AWS 규정 준수 리소스 — 이 통합 문서 및 가이드 모음은 해당 산업 및 지역에 적용될 수 있습니다.
- AWS Config 개발자 안내서의 규칙을 통한 리소스 평가 — AWS Config; 는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.
- AWS Security Hub— 이 AWS 서비스는 보안 업계 표준 및 모범 사례를 준수하는지 확인하는 데 도움이 AWS 되는 내부 보안 상태를 종합적으로 보여줍니다.

Amazon CloudWatch Logs의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. 리전은 물리적으로 분리되고 격리된 다수의 가용 영역을 제공하며 이러한 가용 영역은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크를 통해 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

Amazon CloudWatch 로그의 인프라 보안

Amazon CloudWatch Logs는 관리형 서비스로서 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및 인프라 AWS 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하십시오. 인프라 보안 모범 사례를 사용하여 AWS 환경을 설계하려면 Security Pillar AWS Well-Architected Framework의 인프라 보호를 참조하십시오.

AWS 게시된 API 호출을 사용하여 네트워크를 통해 CloudWatch 로그에 액세스할 수 있습니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)을 사용하여 임시 보안 자격 증명을 생성하여 요청에 서명할 수 있습니다.

인터페이스 CloudWatch VPC 엔드포인트와 함께 로그 사용

Amazon VPC (Virtual Private Cloud) 를 사용하여 AWS 리소스를 호스팅하는 경우 VPC와 로그 간에 프라이빗 연결을 설정할 수 있습니다. CloudWatch 인터넷을 통해 로그를 전송하지 않고도 이 연결을 사용하여 CloudWatch 로그에 로그를 보낼 수 있습니다.

Amazon VPC는 사용자가 AWS 정의한 가상 네트워크에서 AWS 리소스를 시작하는 데 사용할 수 있는 서비스입니다. VPC가 있으면 IP 주소 범위, 서브넷, 라우팅 테이블, 네트워크 게이트웨이 등 네트워크 설정을 제어할 수 있습니다. VPC를 Logs에 연결하려면 CloudWatch Logs에 대한 인터페이스 VPC 엔드포인트를 정의합니다. CloudWatch 이 유형의 엔드포인트를 사용하여 VPC를 AWS 서비스에 연결할 수 있습니다. 엔드포인트는 인터넷 게이트웨이, 네트워크 주소 변환 (NAT) 인스턴스 또는 VPN 연결 없이 안정적이고 확장 가능한 CloudWatch 로그 연결을 제공합니다. 자세한 내용은 Amazon VPC 사용 설명서의 Amazon VPC란 무엇입니까? 섹션을 참조하세요.

인터페이스 VPC 엔드포인트는 사설 IP 주소가 있는 Elastic Network Interface를 사용하여 AWS 서비스 간 사설 통신을 가능하게 하는 AWS 기술인 에 의해 AWS PrivateLink 구동됩니다. 자세한 내용은 신규 — AWS PrivateLink for AWS Services를 참조하십시오.

다음은 Amazon VPC 사용자를 위한 단계들입니다. 자세한 내용은 Amazon VPC 사용 설명서의 시작하기를 참조하세요.

가용성

CloudWatch 로그는 현재 AWS 지역을 포함한 모든 지역의 VPC 엔드포인트를 지원합니다. AWS GovCloud (US)

로그용 VPC 엔드포인트 생성 CloudWatch

VPC에서 CloudWatch Logs를 사용하려면 Logs용 인터페이스 VPC 엔드포인트를 생성하십시오. CloudWatch 선택할 서비스 이름은 com.amazonaws.***Region***.logs입니다. 로그의 설정은 변경할 필요가 없습니다. CloudWatch 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트 생성을 참조하세요.

VPC와 로그 간의 연결 테스트 CloudWatch

엔드포인트를 생성한 후에 연결을 테스트할 수 있습니다.

VPC와 Logs 엔드포인트 간의 연결을 테스트하려면 CloudWatch

1. VPC에 있는 Amazon EC2 인스턴스에 연결합니다. 연결에 대한 자세한 내용은 Amazon EC2 설명서의 Linux 인스턴스에 연결 또는 Windows 인스턴스에 연결을 참조하세요.
2. 인스턴스에서 AWS CLI 를 사용하여 기존 로그 그룹 중 하나에 로그 항목을 생성합니다.

   먼저 로그 이벤트 JSON 파일을 생성합니다. 타임스탬프는 1970년 1월 1일 1:1970 00:00:00 UTC 이후 경과된 시간(밀리초)으로 지정해야 합니다.

   ```
   [
     {
       "timestamp": 1533854071310,
       "message": "VPC Connection Test"
     }
   ]
   ```

   그런 후 `put-log-events` 명령을 사용하여 로그 항목을 생성합니다.

   ```
   aws logs put-log-events --log-group-name LogGroupName --log-stream-
   name LogStreamName --log-events file://JSONFileName
   ```

   명령이 성공해 VPC 엔드포인트를 사용할 수 있는 상태가 되었다면 명령에 대한 응답에 `nextSequenceToken` 명령이 포함됩니다.

CloudWatch Logs VPC 엔드포인트에 대한 액세스 제어

VPC 엔드포인트 정책은 엔드포인트를 만들거나 수정 시 엔드포인트에 연결하는 IAM 리소스 정책입니다. 엔드포인트를 만들 때 정책을 추가하지 않으면 서비스에 대한 모든 액세스를 허용하는 기본 정책이 추가됩니다. 엔드포인트 정책은 IAM 정책 또는 서비스별 정책을 재정의하거나 대체하지 않습니다. 이는 엔드포인트에서 지정된 서비스로의 액세스를 제어하기 위한 별도의 정책입니다.

엔드포인트 정책은 JSON 형식으로 작성해야 합니다.

자세한 내용은 Amazon VPC 사용 설명서의 VPC 엔드포인트를 통해 서비스에 대한 액세스 제어를 참조하세요.

다음은 로그에 대한 CloudWatch 엔드포인트 정책의 예시입니다. 이 정책은 VPC를 통해 CloudWatch Logs에 연결하는 사용자가 로그 스트림을 생성하고 로그를 Logs로 전송하도록 허용하며 다른 CloudWatch CloudWatch Logs 작업을 수행하지 못하도록 합니다.

```
{
  "Statement": [
    {
      "Sid": "PutOnly",
      "Principal": "*",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

로그에 대한 VPC 엔드포인트 정책을 수정하려면 CloudWatch

1. https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. 탐색 창에서 엔드포인트를 선택합니다.
3. CloudWatch 로그용 엔드포인트를 아직 생성하지 않은 경우 엔드포인트 생성을 선택합니다. 그런 다음 com.amazonaws.***Region***.logs를 선택하고 엔드포인트 생성을 선택합니다.
4. com.amazonaws.***Region***.logs 엔드포인트를 선택하고 화면 하단의 정책 탭을 선택합니다.
5. 정책 편집(Edit Policy)을 선택하고 정책을 변경합니다.

VPC 컨텍스트 키에 대한 지원

CloudWatch 로그는 특정 VPC 또는 특정 VPC 엔드포인트에 대한 액세스를 제한할 수 있는 `aws:SourceVpc` 및 `aws:SourceVpce` 컨텍스트 키를 지원합니다. 이러한 키는 사용자가 VPC 엔드포인트를 사용하고 있을 때만 작동합니다. 자세한 내용은 IAM 사용 설명서의 일부 서비스에 사용 가능한 키를 참조하세요.

로깅 CloudWatch 로그 API 및 콘솔 작업 AWS CloudTrail

Amazon CloudWatch Logs는 Logs 내에서 CloudWatch 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업의 기록을 제공하는 AWS 서비스와 통합되어 있습니다. CloudTrail 계정에서 또는 AWS 계정을 대신하여 이루어진 API 호출을 캡처합니다. 캡처된 호출에는 CloudWatch 콘솔에서의 호출과 CloudWatch Logs API 작업에 대한 코드 호출이 포함됩니다. 트레일을 생성하면 CloudWatch Logs CloudTrail 이벤트를 포함하여 Amazon S3 버킷에 이벤트를 지속적으로 전송할 수 있습니다. 트레일을 구성하지 않아도 CloudTrail 콘솔의 이벤트 기록에서 가장 최근 이벤트를 계속 볼 수 있습니다. 에서 수집한 CloudTrail 정보를 사용하여 CloudWatch Logs에 이루어진 요청, 요청이 이루어진 IP 주소, 요청한 사람, 요청 시기 및 추가 세부 정보를 확인할 수 있습니다.

구성 및 활성화 방법을 CloudTrail 포함하여 자세한 내용은 사용 AWS CloudTrail 설명서를 참조하십시오.

주제

- CloudWatch 로그인 정보 CloudTrail
- 쿼리 생성 정보는 CloudTrail
- 로그 파일 항목 이해

CloudWatch 로그인 정보 CloudTrail

CloudTrail 계정을 만들면 AWS 계정에서 활성화됩니다. 지원되는 이벤트 활동이 CloudWatch 로그에서 발생하면 해당 활동이 CloudTrail 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 이벤트에 기록됩니다. AWS 계정에서 최근 이벤트를 보고, 검색하고, 다운로드할 수 있습니다. 자세한 내용은 이벤트 기록으로 CloudTrail 이벤트 보기를 참조하십시오.

CloudWatch 로그용 이벤트를 포함하여 AWS 계정에서 진행 중인 이벤트 기록을 보려면 트레일을 생성하세요. 트레일을 사용하면 CloudTrail Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 기본적으로 콘솔에서 트레일을 생성하면 트레일이 모든 AWS 지역에 적용됩니다. 트레일은 AWS 파티션에 있는 모든 지역의 이벤트를 기록하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 또한 CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 이에 따라 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하십시오.

- 추적 생성 개요
- CloudTrail 지원되는 서비스 및 통합
- 에 대한 Amazon SNS 알림 구성 CloudTrail

- 여러 지역에서 CloudTrail 로그 파일 수신 및 여러 계정으로부터 CloudTrail 로그 파일 수신

CloudWatch 로그는 다음과 같은 작업을 CloudTrail 로그 파일에 이벤트로 기록하는 것을 지원합니다.

- CancelExportTask
- CreateExportTask
- CreateLogGroup
- CreateLogStream
- DeleteDestination
- DeleteLogGroup
- DeleteLogStream
- DeleteMetricFilter
- DeleteRetentionPolicy
- DeleteSubscriptionFilter
- PutDestination
- PutDestinationPolicy
- PutMetricFilter
- PutResourcePolicy
- PutRetentionPolicy
- PutSubscriptionFilter
- StartQuery
- StopQuery
- TestMetricFilter

이러한 CloudWatch Logs API 작업에는 요청 요소만 CloudTrail 로그인됩니다.

- DescribeDestinations
- DescribeExportTasks
- DescribeLogGroups
- DescribeLogStreams
- DescribeMetricFilters
- DescribeQueries

- DescribeResourcePolicies
- DescribeSubscriptionFilters
- FilterLogEvents
- GetLogEvents
- GetLogGroupFields
- GetLogRecord
- GetQueryResults

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. ID 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 IAM 사용자 보안 인증 정보로 했는지 여부.
- 역할 또는 페더레이션 사용자의 임시 보안 인증을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에 의해 이루어졌는지 여부.

자세한 내용은 CloudTrail 사용자 ID 요소를 참조하십시오.

쿼리 생성 정보는 CloudTrail

CloudTrail 쿼리 생성기 콘솔 이벤트 로깅도 지원됩니다. 쿼리 생성기는 현재 CloudWatch 로그 인사이트 및 CloudWatch 지표 인사이트에 지원됩니다. 이러한 CloudTrail 이벤트의 경우 `eventSource` is가 발생합니다`monitoring.amazonaws.com`.

다음 예제는 CloudWatch Logs Insights에서의 GenerateQuery작업을 보여주는 CloudTrail 로그 항목을 보여줍니다.

```
{
    "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"
}
```


로그 파일 항목 이해

트레일은 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 전송할 수 있는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함되어 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜 및 시간, 요청 매개 변수 등에 대한 정보를 포함합니다. CloudTrail 로그 파일은 공개 API 호출의 정렬된 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음 로그 파일 항목은 사용자가 CloudWatch Logs CreateExportTask작업을 호출했음을 보여줍니다.


```
{
        "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"
}
```

CloudWatch 로그 에이전트 참조

> **Important**
>
> 이 참조는 더 이상 사용되지 않는 이전의 CloudWatch Logs 에이전트를 위한 것입니다. 인스턴스 메타데이터 서비스 버전 2 (IMDSv2) 를 사용하는 경우 새 통합 에이전트를 사용해야 합니다. CloudWatch IMDSv2를 사용하지 않는 경우에도 이전 로그 CloudWatch 에이전트 대신 최신 통합 에이전트를 사용하는 것이 좋습니다. 최신 통합 에이전트에 대한 자세한 내용은 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버에서 지표 및 로그 수집을 참조하십시오. CloudWatch
>
> 이전 CloudWatch Logs 에이전트에서 통합 에이전트로 마이그레이션하는 방법에 대한 자세한 내용은 마법사를 사용하여 CloudWatch 에이전트 구성 파일 생성을 참조하십시오.

CloudWatch Logs 에이전트는 Amazon EC2 인스턴스에서 CloudWatch 로그 데이터를 Logs로 전송하는 자동화된 방법을 제공합니다. 에이전트에는 다음 구성 요소가 포함되어 있습니다.

- 로그 데이터를 Logs로 AWS CLI 푸시하는 플러그인입니다. CloudWatch
- 데이터를 Logs로 푸시하는 프로세스를 시작하는 스크립트 (데몬). CloudWatch
- 데몬이 항상 실행되도록 하는 cron 작업입니다.

에이전트 구성 파일

CloudWatch Logs 에이전트 구성 파일은 Logs 에이전트에 필요한 정보를 설명합니다. CloudWatch 에이전트 구성 파일의 [general] 섹션은 모든 로그 스트림에 적용되는 공통 구성을 정의합니다. [logstream] 섹션은 원격 로그 스트림에 로컬 파일을 전송하는 데 필요한 정보를 정의합니다. [logstream] 섹션은 하나 이상 정의가 가능하지만, 각 섹션은 구성 파일 내에 고유한 이름을 가져야 합니다(예: [logstream1], [logstream2] 등). 로그 파일에 있는 데이터의 첫 줄과 함께 [logstream] 값은 로그 파일의 ID를 정의합니다.

```
[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]
...
```

상태 파일

상태 파일이 저장되는 장소를 지정합니다.

logging_config_file

(선택 사항) 에이전트 로깅 구성 파일의 위치를 지정합니다. 여기에서 에이전트 로깅 구성 파일을 지정하지 않으면 기본 파일인 awslogs.conf가 사용됩니다. 스크립트를 통해 에이전트를 설치한 경우 기본 파일 위치는 `/var/awslogs/etc/awslogs.conf`이고, rpm을 통해 에이전트를 설치한 경우 `/etc/awslogs/awslogs.conf`입니다. 파일은 Python 구성 파일 형식 (https://docs.python.org/2/library/logging.config.html #logging-config-fileformat) 입니다. 다음 이름을 가진 로거를 사용자가 지정할 수 있습니다.

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

아래 샘플은 기본 값이 INFO인 경우 독자와 게시자의 수준을 WARNING으로 수준을 변경합니다.

```
[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

true (기본값) 로 설정하면 gzip http 콘텐츠 인코딩을 활성화하여 압축된 페이로드를 Logs로 보낼 수 CloudWatch 있습니다. 이렇게 하면 CPU 사용량이 줄어들고, 풋 레이턴시가 줄고 NetworkOut, 줄어듭니다. 이 기능을 사용하지 않도록 설정하려면 로그 에이전트 구성 파일의 [일반] 섹션에 use_gzip_http_content_encoding = false를 추가한 다음 에이전트를 다시 시작하십시오. CloudWatch

> **Note**
>
> 이 설정은 awscli-cwlogs 버전 1.3.3 이상에서만 사용할 수 있습니다.

log_group_name

대상 로그 그룹을 지정합니다. 로그 그룹이 없는 경우 이 설정에 따라 로그 그룹이 자동으로 생성됩니다. 로그 그룹 이름에 포함되는 문자 길이는 1~512자입니다. 허용되는 문자: a-z, A-Z, 0-9, '_'(밑줄), '- (하이픈), '/'(슬래시) 및 '.'(마침표)

log_stream_name

대상 로그 스트림을 지정합니다. 리터럴 문자열이나 미리 정의된 변수({instance_id}, {hostname}, {ip_address}) 또는 이 둘의 조합을 사용하여 로그 스트림 이름을 정의할 수 있습니다. 로그 스트림이 없는 경우 이 설정에 따라 로그 스트림이 자동으로 생성됩니다.

datetime_format

타임스탬프가 로그에서 추출되는 방법을 지정합니다. 타임스탬프는 로그 이벤트를 검색하고 지표를 생성하는 데 사용됩니다. datetime_format이 제공되지 않는 경우에는 각 로그 이벤트에서 현재 시간이 사용됩니다. 제공된 datetime_format 값이 해당 로그 메시지에서 유효하지 않으면 타임스탬프가 성공적으로 구문 분석된 마지막 로그 이벤트에서 나온 타임스탬프가 사용됩니다. 이전 로그 이벤트가 존재하지 않는 경우 현재 시간이 사용됩니다.

공통적인 datetime_format 코드가 아래에 나열되어 있습니다. Python, datetime.strptime()에서 지원되는 datetime_format 코드면 무엇이든 사용 가능합니다. 시간대 오프셋(%z)도 지원되기는 하지만, 콜론(:)이 없는 python 3.2, [+-]HHMM이 나올 때까지는 지원되지 않습니다. 자세한 내용은 strftime() 및 strptime() 동작을 참조하세요.

%y: 제로 패딩된 10진수 형태의 세기를 제외한 연도. 00, 01, ..., 99

%Y: 10진수 형태의 세기를 포함한 연도. 1970, 1988, 2001, 2013

%b: 로캘 약어 형태의 월. Jan, Feb, ..., Dec (en_US);

%B: 로캘 전체 이름 형태의 월. January, February, ..., December (en_US);

%m: 제로 패딩된 10진수 형태의 월. 01, 02, ..., 12

%d: 제로 패딩된 10진수 형태의 월 날짜. 01, 02, ..., 31

%H: 제로 패딩된 10진수 형태의 시간(24시간 방식). 00, 01, ..., 23

%I: 제로 패딩된 10진수 형태의 시간(12시간 방식). 01, 02, ..., 12

%p: AM 또는 PM 중 하나에 상응하는 로캘.

%M: 제로 패딩된 10진수 형태의 분. 00, 01, ..., 59

%S: 제로 패딩된 10진수 형태의 초. 00, 01, ..., 59

%f: 왼쪽이 제로 패딩된 10진수 형태의 마이크로초. 000000, ..., 999999

%z: +HHMM 또는 -HHMM 형태의 UTC 오프셋. +0000, -0400, +1030

형식의 예:

```
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

로그 이벤트 타임스탬프의 시간대를 지정합니다. UTC 및 LOCAL 등 2개의 값이 지원됩니다. 기본값인 LOCAL은 datetime_format에 따라 시간대를 추정할 수 없을 때 사용됩니다.

파일

로그로 푸시하려는 로그 파일을 지정합니다. CloudWatch 파일은 특정 파일을 가리키거나 여러 개의 파일을 가리킬 수 있습니다(/var/log/system.log* 같은 와일드카드를 사용). 파일 수정 시간을 기준으로 최신 파일만 CloudWatch 로그에 푸시됩니다. 와일드카드는 여러 종류의 파일(예: access_log_80 및 access_log_443)이 아니라 종류가 같은 일련의 파일(예: access_log.2014-06-01-01, access_log.2014-06-01-02 등)을 지정할 때 사용하는 것이 좋습니다. 여러 종류의 파일을 지정하려면 로그 파일의 종류에 따라 다른 로그 스트림에 들어가도록 구성 파일에 또 다른 로그 스트림 항목을 추가합니다. 압축 파일은 지원되지 않습니다.

file_fingerprint_lines

파일을 식별하기 위한 줄의 범위를 지정합니다. 유효한 값은 하나의 숫자(예: '1')나 대시로 구분된 두 개의 숫자(예: '2-5')입니다. 기본값은 '1'이기 때문에 지문 산출을 위해 첫 번째 줄이 사용됩니다. 지정된 라인을 모두 사용할 수 있는 경우가 아니면 핑거프린트 라인이 CloudWatch Logs로 전송되지 않습니다.

multi_line_start_pattern

로그 메시지의 시작을 식별하기 위해 패턴을 지정합니다. 로그 메시지는 패턴과 일치하는 하나의 줄과 패턴과 일치하지 않는 나머지 줄들로 이루어져 있습니다. 유효한 값은 정규식 또는

{datetime_format}입니다. {datetime_format}을 사용할 때는 반드시 datetime_format 옵션이 지정되어 있어야 합니다. 기본값은 '^[^\s]'이기 때문에 공백이 아닌 문자로 시작되는 줄이 있으면 이전의 로그 메시지가 종료되고 새로운 로그 메시지가 시작됩니다.

initial_position

데이터 읽기를 시작할 지점(start_of_file 또는 end_of_file)을 지정합니다. 기본값은 파일의 시작 지점입니다. 해당 로그 스트림에서 지속되는 상태가 없을 때만 사용됩니다.

인코딩

파일을 정확하게 읽을 수 있도록 로그 파일의 인코딩을 설정합니다. 기본값은 utf_8입니다. Python codecs.decode()에서 지원되는 인코딩을 여기에서 사용할 수 있습니다.

> **Warning**
> 인코딩을 잘못 지정하면 디코딩할 수 없는 문자를 다른 문자들이 대체하면서 데이터 손실이 야기될 수 있습니다.

다음은 몇 가지 일반적인 인코딩입니다.

```
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

로그 이벤트를 일괄 처리하는 기간을 지정합니다. 최솟값은 5,000ms이고, 기본값은 5,000ms입니다.

batch_count

일괄 처리할 로그 이벤트의 최대 수를 지정합니다(10,000까지 가능). 기본값은 10,000입니다.

batch_size

일괄 처리할 로그 이벤트의 최대 크기를 바이트로 지정합니다(1,048,576바이트까지 가능). 기본값은 1048576바이트입니다. 이 크기는 UTF-8에서 모든 이벤트 메시지를 합한 값에 각 로그 이벤트마다 26바이트를 추가하여 계산한 값입니다.

HTTP 프록시와 함께 CloudWatch 로그 에이전트 사용

HTTP 프록시와 함께 CloudWatch 로그 에이전트를 사용할 수 있습니다.

> **Note**
>
> HTTP awslogs-agent-setup 프록시는.py 버전 1.3.8 이상에서 지원됩니다.

HTTP 프록시와 함께 CloudWatch 로그 에이전트를 사용하려면

1. 다음 중 하나를 수행하십시오.

   a. CloudWatch Logs 에이전트를 새로 설치하려면 다음 명령을 실행합니다.

   ```
   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
   ```

   EC2 인스턴스에서 Amazon EC2 메타데이터 서비스에 대한 액세스를 유지하려면 --no-proxy 169.254.169.254를 사용하세요(권장 사항). 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 및 사용자 데이터를 참조하십시오.

   `http-proxy` 및 `https-proxy` 값에서 전체 URL을 지정합니다.

   b. CloudWatch 로그 에이전트의 기존 설치의 경우 /var/awslogs/etc/proxy.conf를 편집하고 프록시를 추가하십시오.

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

2. 에이전트를 다시 시작하면 변경 사항이 적용됩니다.

```
sudo service awslogs restart
```

Amazon Linux 2를 사용 중인 경우 다음 명령과 함께 에이전트를 다시 시작합니다.

```
sudo service awslogsd restart
```

로그 에이전트 구성 파일 분류하기 CloudWatch

awslogs-agent-setup.py 버전 1.3.8 이상을 awscli-cwlogs 1.3.3 이상과 함께 사용하는 경우 /var/awslogs/etc/config/ 디렉터리에 추가 구성 파일을 생성하여 다양한 구성 요소의 서로 다른 스트림 구성을 서로 독립적으로 가져올 수 있습니다. Logs 에이전트가 CloudWatch 시작되면 이러한 추가 구성 파일에 모든 스트림 구성이 포함됩니다. [general] 섹션의 구성 속성들은 반드시 기본 구성 파일(/var/awslogs/etc/awslogs.conf)에 정의되어 있어야 하며, /var/awslogs/etc/config/에 있는 추가 구성 파일에서는 무시가 됩니다.

rpm을 통해 에이전트를 설치하지 않았기 때문에 /var/awslogs/etc/config/ 디렉터리가 없는 경우 /etc/awslogs/config/ 디렉터리를 대신 사용할 수 있습니다.

에이전트를 다시 시작하면 변경 사항이 적용됩니다.

```
sudo service awslogs restart
```

Amazon Linux 2를 사용 중인 경우 다음 명령과 함께 에이전트를 다시 시작합니다.

```
sudo service awslogsd restart
```

CloudWatch 로그 에이전트 FAQ

어떤 종류의 파일 로테이션이 지원됩니까?

다음과 같은 파일 로테이션 메커니즘이 지원되고 있습니다.

- 숫자 접미사를 붙여서 기존 로그 파일의 이름을 바꾸고 원래 빈 로그 파일을 다시 생성하는 방법입니다. 예를 들어 /var/log/syslog.log는 /var/log/syslog.log.1로 이름이 바뀝니다. 이전 로테이션에서 /var/log/syslog.log.1이 이미 존재하는 경우에는 /var/log/syslog.log.2로 이름이 바뀝니다.
- 복사본을 생성한 후에 원래 로그 파일을 잘라냅니다. 예를 들어 /var/log/syslog.log는 /var/log/syslog.log.1에 복사가 된 후 잘립니다. 이 경우 데이터 손실이 발생할 수 있기 때문에 이 파일 로테이션 메커니즘을 사용할 때는 조심해야 합니다.
- 기존 파일과 같은 공통 패턴을 따르는 파일을 새로 생성하는 방법입니다. 예를 들어 /var/log/syslog.log.2014-01-01이 그대로 남아 있는 상태에서 /var/log/syslog.log.2014-01-02가 생성됩니다.

파일의 지문(소스 ID)은 로그 스트림 키와 파일 콘텐츠의 첫 줄을 해싱하여 산출됩니다. 이 동작을 재정의하기 위해 file_fingerprint_lines 옵션을 사용할 수 있습니다. 파일 로테이션이 일어나면 새 파일은 새 콘텐츠를 가진 것으로 간주되지만, 기존 파일은 콘텐츠가 추가된 것으로 간주되지 않습니다. 따라서 에이전트는 기존 파일에 대한 읽기를 마치고 나면 새 파일을 푸시합니다.

사용 중인 에이전트의 버전을 어떻게 확인할 수 있습니까?

설치 스크립트를 사용하여 CloudWatch 로그 에이전트를 설치한 경우 /var/awslogs/bin/awslogs-version.sh 를 사용하여 사용 중인 에이전트 버전을 확인할 수 있습니다. 에이전트의 버전과 중요한 플러그인들이 출력됩니다. yum을 사용하여 CloudWatch 로그 에이전트를 설치한 경우 “yum info awslogs” 및 “yum info aws-cli-plugin-cloudwatch -logs”를 사용하여 로그 에이전트 및 플러그인의 버전을 확인할 수 있습니다. CloudWatch

로그 항목들은 어떻게 로그 이벤트로 변환됩니까?

로그 이벤트에는 이벤트가 발생한 시점에 대한 타임스탬프와 원시 로그 메시지 등 두 개의 속성이 포함되어 있습니다. 기본적으로 공백이 아닌 문자로 시작되는 줄이 있으면 이전의 로그 메시지가 종료되고 새로운 로그 메시지가 시작됩니다. 이 동작을 재정의하기 위해 multi_line_start_pattern을 사용할 수 있으면 패턴과 일치하는 모든 줄에서 새로운 로그 메시지가 시작됩니다. 어떤 regex 또는 {datetime_format}이든 패턴이 될 수 있습니다. 예를 들어 모든 로그 메시지의 첫 줄에 '2014-01-02T13:13:01Z' 같은 타임스탬프가 포함되어 있으면 multi_line_start_pattern을 '\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}Z'로 설정할 수 있습니다. 간편한 구성을 위해 datetime_format option이 지정된 경우에 '{datetime_format}' 변수를 사용할 수 있습니다. 같은 예에서 datetime_format이 '%Y-%m-%dT%H:%M:%S%z'로 설정되어 있으면 multi_line_start_pattern이 간단히 '{datetime_format}'이 될 수 있습니다.

datetime_format이 제공되지 않는 경우에는 각 로그 이벤트에서 현재 시간이 사용됩니다. 제공된 datetime_format 값이 해당 로그 메시지에서 유효하지 않으면 타임스탬프가 성공적으로 구문 분석된 마지막 로그 이벤트에서 나온 타임스탬프가 사용됩니다. 이전 로그 이벤트가 존재하지 않는 경

우 현재 시간이 사용됩니다. 로그 이벤트가 현재 시간이나 이전 로그 이벤트 시간으로 돌아가면 경고 메시지가 기록됩니다.

타임스탬프는 로그 이벤트를 검색하고 지표를 생성하는 데 사용되기 때문에 형식을 잘못 지정하면 로그 이벤트 검색이 불가능해지고 잘못된 지표가 생성될 수 있습니다.

로그 이벤트는 어떻게 일괄 처리됩니까?

아래 조건 중 어떤 것이든 충족이 되면 배치(batch)가 가득 차면서 게시가 됩니다.

1. 첫 번째 로그 이벤트가 추가되었기 때문에 buffer_duration 시간이 경과되었습니다.
2. batch_size 보다 작은 로그 이벤트들이 누적되었지만 새로운 로그 이벤트를 추가하면 batch_size를 초과하게 됩니다.
3. 로그 이벤트의 수가 batch_count에 도달했습니다.
4. 배치에서 나온 로그 이벤트들이 24시간 이상 지속되지 않지만, 새 로그 이벤트를 추가하면 24시간이라는 제약 조건을 넘어서게 됩니다.

로그 항목, 로그 이벤트 또는 배치는 어떤 이유로 건너 뛰기가 되거나 잘라집니까?

`PutLogEvents` 작업의 제약 조건을 따르다 보면 다음과 같은 문제들로 인해 로그 이벤트나 배치를 건너 뛰는 상황이 발생할 수 있습니다.

> **Note**
>
> CloudWatch Logs 에이전트는 데이터를 건너뛰면 로그에 경고를 기록합니다.

1. 로그 이벤트의 크기가 256KB를 초과하면 로그 이벤트가 완전히 건너 뛰기 됩니다.
2. 로그 이벤트의 타임스탬프가 미래에 2시간 이상이면 해당 로그 이벤트가 건너 뛰기 됩니다.
3. 로그 이벤트의 타임스탬프가 과거에 14일 이상이었으면 해당 로그 이벤트가 건너 뛰기 됩니다.
4. 로그 그룹의 보존 기간을 지난 로그 이벤트가 있으면 전체 배치가 건너 뛰기 됩니다.
5. 단일 `PutLogEvents` 요청에서 로그 이벤트에 대한 배치가 24시간 이상 지속된 경우에는 `PutLogEvents` 작업이 실패합니다.

에이전트 중지로 인해 데이터 손실/중복이 발생하고 있습니까?

상태 파일이 사용 가능하고 마지막 실행 이후로 파일 로테이션이 발생하지 않은 한 그렇지 않습니다. CloudWatch 로그 에이전트는 중지된 지점부터 시작하여 로그 데이터를 계속 푸시할 수 있습니다.

같은 호스트나 다른 호스트에서 나온 서로 다른 로그 파일이 동일한 로그 스트림으로 가리키도록 할 수 있습니까?

단일 로그 스트림으로 데이터를 전송하기 위해 여러 개의 로그 소스를 구성하는 것이 불가능합니다.

에이전트가 어떤 API를 호출합니까? (또는 IAM 정책에 어떤 작업을 추가해야 합니까?)

CloudWatch Logs 에이전트에는`CreateLogGroup`, `CreateLogStream``DescribeLogStreams`, 및 `PutLogEvents` 작업이 필요합니다. 최신 에이전트를 사용하고 있는 경우에는 `DescribeLogStreams`가 필요하지 않습니다. IAM 정책 샘플은 아래를 참조하세요.

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

CloudWatch 로그 에이전트가 로그 그룹이나 로그 스트림을 자동으로 생성하는 것을 원하지 않습니다. 어떻게 하면 에이전트가 로그 그룹 또는 로그 스트림을 다시 생성하지 않게 할 수 있습니까?

IAM 정책에서 `DescribeLogStreams`, `PutLogEvents`작업만 수행하도록 에이전트 작업을 제한할 수 있습니다.

에이전트에서 `CreateLogGroup` 및 `CreateLogStream` 권한을 취소하기 전에 에이전트에서 사용할 로그 그룹과 로그 스트림을 모두 생성해야 합니다. 로그 에이전트는 `CreateLogGroup` 및 `CreateLogStream` 권한이 모두 없는 한 사용자가 생성한 로그 그룹에 로그 스트림을 생성할 수 없습니다.

문제 해결 시 어떤 로그를 살펴봐야 합니까?

에이전트 설치 로그는 /var/log/awslogs-agent-setup.log에, 에이전트 로그는 /var/log/awslogs.log에 있습니다.

CloudWatch 메트릭을 사용한 모니터링

CloudWatch 로그는 CloudWatch 1분마다 지표를 Amazon으로 전송합니다.

CloudWatch 로그 지표

AWS/Logs 네임스페이스에 포함된 지표는 다음과 같습니다.

지표	설명
CallCount	계정에서 수행된 지정된 API 작업 수 CallCount CloudWatch 로그 서비스 사용량 지표입니다. 자세한 정보는 CloudWatch 서비스 사용 지표를 기록합니다.을 참조하세요. 적합한 차원: 클래스, 리소스, 서비스, 유형 유효한 통계: Sum 단위: 없음
DeliveryErrors	구독 대상으로 데이터를 전달할 때 CloudWatch 로그에서 오류가 발생한 로그 이벤트 수입니다. 대상 서비스가 제한 예외 또는 재시도 가능한 서비스 예외 (예: HTTP 5xx) 와 같은 재시도 가능한 오류를 반환하는 경우 CloudWatch Logs는 최대 24시간 동안 전송을 계속 재시도합니다. CloudWatch 오류가 또와 같이 재시도할 수 없는 오류인 경우 로그는 재전송을 시도하지 않습니다. AccessDeniedException ResourceNotFoundException 유효 크기:,,, LogGroupName DestinationType FilterName PolicyLevel 유효한 통계: Sum 단위: 없음
DeliveryThrottling	구독 대상으로 데이터를 전달할 때 CloudWatch 로그가 병목 현상이 발생한 로그 이벤트 수입니다.

지표	설명
	대상 서비스가 제한 예외 또는 재시도 가능한 서비스 예외 (예: HTTP 5xx) 와 같은 재시도 가능한 오류를 반환하는 경우 Logs는 최대 24시간 동안 전송을 계속 재시도합니다. CloudWatch CloudWatch 오류가 또와 같이 재시도할 수 없는 오류인 경우 로그는 재전송을 시도하지 않습니다. `AccessDeniedException` `ResourceNotFoundException` 유효 크기:,,, LogGroupName DestinationType FilterName PolicyLevel 유효한 통계: Sum 단위: 없음
`EMFParsingErrors`	임베디드 지표 형식 로그를 처리하는 동안 발생한 구문 분석 오류 수입니다. 이러한 오류는 로그가 임베디드 지표 형식으로 식별되지만 올바른 형식을 따르지 않을 때 발생합니다. 임베디드 지표 형식에 대한 자세한 정보는 사양: 임베디드 지표 형식을 참조하세요. 유효한 차원: `LogGroupName` 유효한 통계: Sum 단위: 없음

지표	설명
EMFValidationErrors	임베디드 지표 형식 로그를 처리하는 동안 발생한 유효성 검사 오류 수입니다. 이러한 오류는 임베디드 지표 형식 로그 내의 지표 정의가 임베디드 지표 형식 및 MetricDatum 사양을 준수하지 않을 때 발생합니다. CloudWatch 내장된 지표 형식에 대한 자세한 내용은 사양: 내장된 지표 형식을 참조하십시오. 데이터 유형에 MetricDatum 대한 자세한 내용은 Amazon CloudWatch API 참조를 참조하십시오 MetricDatum. **Note** 특정 유효성 검사 오류로 인해 EMF 로그 내의 여러 지표가 게시되지 않을 수 있습니다. 예를 들어, 유효하지 않은 네임스페이스로 설정된 모든 지표는 삭제됩니다. 유효한 차원: LogGroupName 유효한 통계: Sum 단위: 없음
ErrorCount	계정에서 수행되어 오류가 발생한 API 작업 수 ErrorCount CloudWatch 로그 서비스 사용량 지표입니다. 자세한 정보는 CloudWatch 서비스 사용 지표를 기록합니다.을 참조하세요. 적합한 차원: 클래스, 리소스, 서비스, 유형 유효한 통계: Sum 단위: 없음
ForwardedBytes	구독 대상으로 전송되는 압축 바이트 단위의 로그 이벤트 볼륨 유효 치수: LogGroupName, DestinationType, FilterName 유효한 통계: Sum 단위: 바이트

지표	설명
ForwardedLogEvents	구독 대상으로 전송되는 로그 이벤트 수 유효 치수: LogGroupName, DestinationType, FilterName, PolicyLevel 유효한 통계: Sum 단위: 없음
IncomingBytes	Logs에 업로드된 압축되지 않은 바이트 단위의 로그 이벤트 양. CloudWatch LogGroupName 차원과 함께 사용하면 로그 그룹으로 업로드되는 비압축 바이트 단위의 로그 이벤트 볼륨이 됩니다. 유효 크기: LogGroupName 유효한 통계: Sum 단위: 바이트
IncomingLogEvents	CloudWatch Logs에 업로드된 로그 이벤트 수입니다. LogGroupName 차원과 함께 사용하면 로그 그룹으로 업로드되는 로그 이벤트 수가 됩니다. 유효 크기: LogGroupName 유효한 통계: Sum 단위: 없음
LogEventsWithFindings	로그 데이터 보호 기능을 사용하여 감사하는 데이터 문자열과 일치하는 CloudWatch 로그 이벤트 수입니다. 자세한 정보는 마스킹 처리를 통해 민감한 로그 데이터를 보호하도록 지원을 참조하세요. 유효한 차원: 없음 유효한 통계: Sum 단위: 없음

지표	설명
ThrottleCount	사용 할당량으로 인해 제한된 계정에서 수행된 API 작업 수. ThrottleCount CloudWatch 로그 서비스 사용량 지표입니다. 자세한 정보는 CloudWatch 서비스 사용 지표를 기록합니다.을 참조하세요. 적합한 차원: 클래스, 리소스, 서비스, 유형 유효한 통계: Sum 단위: 없음

CloudWatch 로그 지표의 크기

CloudWatch 로그 지표에 사용할 수 있는 측정기준은 다음 표에 나와 있습니다.

측정기준	설명
LogGroupName	지표를 표시할 CloudWatch 로그 로그 그룹의 이름.
DestinationType	CloudWatch 로그 데이터의 구독 대상 (Amazon Kinesis Data Streams 또는 Amazon Data Firehose) 일 AWS Lambda수 있습니다.
FilterName	데이터를 로그 그룹에서 대상으로 전송하는 구독 필터 이름. 구독 필터 이름은 자동으로 CloudWatch ASCII로 변환되며 지원되지 않는 문자는 물음표 (?) 로 바뀝니다.

계정 수준 구독 필터와 관련된 지표의 크기는 다음 표에 나열되어 있습니다.

측정기준	설명
PolicyLevel	정책이 적용되는 수준. 현재 이 차원에 사용할 수 있는 유일한 값은 다음과 같습니다. AccountPolicy

측정기준	설명
DestinationType	CloudWatch 로그 데이터의 구독 대상 (Amazon Kinesis Data Streams 또는 Amazon Data Firehose) 일 AWS Lambda수 있습니다.
FilterName	데이터를 로그 그룹에서 대상으로 전송하는 구독 필터 이름. 구독 필터 이름은 자동으로 CloudWatch ASCII로 변환되며 지원되지 않는 문자는 물음표 (?) 로 바뀝니다.

CloudWatch 서비스 사용 지표를 기록합니다.

CloudWatch 로그는 사용량 CloudWatch 로그 API 작업을 CloudWatch 추적하는 지표를 전송합니다. 이러한 지표는 AWS 서비스 할당량에 해당합니다. 이러한 지표를 추적하면 할당량을 사전 예방적으로 관리하는 데 도움이 될 수 있습니다. 자세한 내용은 Service Quotas 통합 및 사용량 지표를 참조하세요.

예를 들어, ThrottleCount 지표를 추적하거나 해당 지표에 대해 경보를 설정할 수 있습니다. 이 지표의 값이 증가하면 제한되는 API 작업에 대한 할당량 증가를 요청하는 것이 좋습니다. CloudWatch 로그 서비스 할당량에 대한 자세한 내용은 을 참조하십시오. CloudWatch 로그 할당량

CloudWatch Logs는 매분마다 및 네임스페이스에 서비스 할당량 사용량 지표를 게시합니다AWS/Usage. AWS/Logs

다음 표에는 Logs에서 게시한 서비스 사용량 지표가 나와 있습니다. CloudWatch 이러한 지표에는 지정된 단위가 없습니다. 지표에 가장 유용한 통계는 SUM이며 1분 동안의 총 작업 수를 나타냅니다.

이러한 각 지표는 Service, Class, Type 및 Resource 차원 모두에 대해 값과 함께 게시됩니다. 또한 해당 지표는 Account Metrics라는 단일 차원으로 게시됩니다. Account Metrics 차원을 사용하여 계정의 모든 API 작업에 대한 지표 합계를 확인할 수 있습니다. 다른 차원을 사용하고 특정 API에 대한 지표를 찾을 수 있도록 Resource 차원에 대한 API 작업의 이름을 지정합니다.

지표

지표	설명
CallCount	계정에서 수행된 지정된 작업 수

지표	설명
	CallCount 는 AWS/Usage 및 AWS/Logs 네임스페이스 둘 다에서 게시됩니다.
ErrorCount	계정에서 수행되어 오류가 발생한 API 작업 수 ErrorCount 는 AWS/Logs에서만 게시됩니다.
ThrottleCount	사용 할당량으로 인해 제한된 계정에서 수행된 API 작업 수. ThrottleCount 는 AWS/Logs에서만 게시됩니다.

측정기준

차원	설명
Account metrics	이 측정기준을 사용하여 모든 CloudWatch Logs API의 측정치 합계를 구할 수 있습니다. 특정 API에 대한 지표를 보려면 이 표에 나열된 다른 차원을 사용하고 API 이름을 Resource의 값으로 지정합니다.
Service	리소스가 포함된 AWS 서비스의 이름. CloudWatch 로그 사용량 지표의 경우 이 측정기준 값은 입니다Logs.
Class	추적 중인 리소스 클래스. CloudWatch 로그 API 사용 지표는 값이 1인 이 차원을 사용합니다. None
Type	추적 중인 리소스의 유형. 현재, Service 치수가 Logs인 경우 Type에 대한 유일한 유효한 값은 API입니다.
Resource	API 작업의 이름. 유효한 값에는 작업에 나열된 모든 API 작업 이름이 포함됩니다. 예: PutLogEvents

CloudWatch 로그 할당량

다음 표에는 계정의 CloudWatch 로그에 대한 기본 서비스 할당량 (한도라고도 함) 이 나와 있습니다. AWS 전부는 아니지만 대부분의 서비스 할당량은 Service Quotas 콘솔의 Amazon CloudWatch Logs 네임스페이스에 나열되어 있습니다. 이러한 할당량에 대한 할당량 증가를 요청하려면 이 섹션의 뒷부분에 있는 절차를 참조하세요.

Resource	기본 할당량
계정 수준 정책	계정당 하나의 계정 수준 구독 필터 정책. 계정당 하나의 계정 수준 데이터 보호 정책. 이러한 할당량은 변경할 수 없습니다.
이상 탐지기	계정당 이상 탐지기 10개. 이 할당량은 변경할 수 없습니다.
배치 크기	최대 배치 크기는 1,048,576바이트입니다. 이 크기는 UTF-8에서 모든 이벤트 메시지를 합한 값에 각 로그 이벤트마다 26바이트를 추가하여 계산한 값입니다. 이 할당량은 변경할 수 없습니다.
데이터 보관	최대 5GB까지 데이터를 무료 보관할 수 있습니다. 이 할당량은 변경할 수 없습니다.
CreateLogGroup	초당 10건의 트랜잭션 (TPS/계정/지역), 이후 트랜잭션은 병목 현상이 발생합니다. 할당량 증가를 요청할 수 있습니다.
CreateLogStream	50건의 초당 트랜잭션(TPS/계정/리전). 이후 트랜잭션에 병목 현상이 발생합니다. 할당량 증가를 요청할 수 있습니다.
사용자 지정 데이터 식별자	각 데이터 보호 정책에는 최대 10개의 사용자 지정 데이터 식별자가 포함될 수 있습니다. 할당량 증가를 요청할 수 있습니다.

Resource	기본 할당량
	사용자 지정 데이터 식별자를 정의하는 각 정규 표현식은 최대 200자까지 포함할 수 있습니다. 이 할당량은 변경할 수 없습니다.
DeleteLogGroup	초당 10개의 트랜잭션 (TPS/계정/지역), 그 이후에는 트랜잭션이 병목 현상이 발생합니다. 할당량 증가를 요청할 수 있습니다.
DeleteLogStream	초당 15건의 트랜잭션 (TPS/계정/지역), 이후 트랜잭션은 병목 현상이 발생합니다. 할당량 증가를 요청할 수 있습니다.
DescribeLogGroups	초당 10건의 트랜잭션 (TPS/계정/지역). 할당량 증가를 요청할 수 있습니다.
DescribeLogStreams	초당 25건의 거래 (TPS/계정/지역). 할당량 증가를 요청할 수 있습니다.
검색된 로그 필드	CloudWatch Logs Insights는 로그 그룹에서 최대 1000개의 로그 이벤트 필드를 검색할 수 있습니다. 이 할당량은 변경할 수 없습니다. 자세한 정보는 지원되는 로그 및 검색되는 필드 섹션을 참조하세요.
JSON 로그에서 추출된 로그 필드	CloudWatch 로그 인사이트는 JSON 로그에서 최대 200개의 로그 이벤트 필드를 추출할 수 있습니다. 이 할당량은 변경할 수 없습니다. 자세한 정보는 지원되는 로그 및 검색되는 필드을 참조하세요.
내보내기 작업	계정당 한 번에 하나씩 (실행 중이거나 보류 중) 내보내기 작업이 활성화됩니다. 이 할당량은 변경할 수 없습니다.

Resource	기본 할당량
FilterLogEvents	미국 동부(버지니아 북부)에서 초당 25개 요청. 다음 지역에서 초당 요청 5개: • 아시아 태평양(자카르타) • 아시아 태평양(오사카) • 유럽(프랑크푸르트) • 캐나다 서부(캘거리) • 이스라엘(텔아비브) 다른 지역에서는 초당 요청 10개. 이 할당량은 변경할 수 없습니다.
GetLogEvents	유럽(파리)에서 초당 30개 요청. 다음 리전에서 초당 10개 요청: • 미국 서부(오레곤) • 아시아 태평양(자카르타) • 아시아 태평양(오사카) • 캐나다 서부(캘거리) • 유럽(아일랜드) • 유럽(프랑크푸르트) • 이스라엘(텔아비브) 기타 모든 지역에서는 초당 25개 요청. 이 할당량은 변경할 수 없습니다. 새로운 데이터를 지속적으로 처리하는 경우 구독을 사용하는 것이 좋습니다. 기록 데이터가 필요한 경우 데이터를 Amazon S3로 내보내는 것이 좋습니다.

Resource	기본 할당량
수신 데이터	최대 5GB까지 데이터를 무료로 수신할 수 있습니다. 이 할당량은 변경할 수 없습니다.
Live Tail 동시 세션.	15개의 동시 세션. 할당량 증가를 요청할 수 있습니다.
Live Tail: 한 세션에서 검색된 로그 그룹입니다.	하나의 Live Tail 세션에서 최대 10개의 로그 그룹 스캔됨. 이 할당량은 변경할 수 없습니다.
로그 이벤트 크기	256KB(최대). 이 할당량은 변경할 수 없습니다.
로그 그룹	리전별 계정당 1,000,000개의 로그 그룹. 할당량 증가를 요청할 수 있습니다. 하나의 로그 그룹에서 포함할 수 있는 로그 스트림의 수에는 할당량이 없습니다.
지표 필터	로그 그룹당 100개. 이 할당량은 변경할 수 없습니다.
임베디드 지표 형식 지표	로그 이벤트당 100개의 지표 및 지표당 30개의 차원. 내장된 지표 형식에 대한 자세한 내용은 Amazon CloudWatch User Guide의 사양: 내장된 지표 형식을 참조하십시오.
PutLogEvents	PutLogEvents 요청의 최대 배치 크기는 1MB입니다. 이 크기는 UTF-8에서 모든 이벤트 메시지를 합한 값에 각 로그 이벤트마다 26바이트를 추가하여 계산한 값입니다. 지역별 계정당 초당 트랜잭션 5000건 서비스를 사용하여 초당 제한 할당량을 늘리도록 요청할 수 있습니다. Service Quotas
쿼리 실행 제한 시간	60분이 지나면 CloudWatch 로그 인사이트의 쿼리 시간이 초과됩니다. 이 시간 제한은 변경할 수 없습니다.
쿼리된 로그 그룹	단일 CloudWatch Logs Insights 쿼리에서 최대 50개의 로그 그룹을 쿼리할 수 있습니다. 이 할당량은 변경할 수 없습니다.

Resource	기본 할당량
쿼리 동시성	표준 클래스 로그 그룹의 경우 대시보드에 추가된 쿼리를 포함하여 최대 30개의 동시 CloudWatch Logs Insights 쿼리가 가능합니다. Infrequent Access 클래스 로그 그룹의 경우 대시보드에 추가된 쿼리를 포함하여 최대 5개의 동시 CloudWatch Logs Insights 쿼리가 가능합니다. 이러한 할당량은 변경할 수 없습니다.
자연어로 생성된 쿼리	자연어로 생성되는 동시 쿼리 요청 최대 5개.
쿼리 사용 가능 여부	콘솔에서 구성된 쿼리는 History 명령을 통해 30일 동안 사용할 수 있습니다. 이 가용 기간은 변경할 수 없습니다. 를 사용하여 만든 쿼리 정의는 만료되지 PutQueryDefinition않습니다.
쿼리 결과 가용 시간	쿼리 결과는 7일 동안 검색할 수 있습니다. 이 가용 시간은 변경할 수 없습니다.
콘솔에 표시되는 쿼리 결과	기본적으로 최대 1,000행의 쿼리 결과가 콘솔에 표시됩니다. 쿼리에서 **`limit`** 명령을 사용하여 이 제한을 10,000개 행으로 늘릴 수 있습니다. 자세한 정보는 CloudWatch 로그 인사이트 쿼리 구문을 참조하세요.
정규식	지표 필터 또는 구독 필터를 생성하는 경우 각 로그 그룹에 대한 정규식이 포함된 최대 5개의 필터 패턴. 이 할당량은 변경할 수 없습니다. 지표 필터 및 구독 필터에 대해 구분된 필터 패턴 또는 JSON 필터 패턴을 생성하거나 로그 이벤트를 필터링하는 경우 각 필터 패턴에 대한 최대 2개의 정규식.
리소스 정책	계정당 지역당 최대 10개의 CloudWatch 로그 리소스 정책. 이 할당량은 변경할 수 없습니다.

Resource	기본 할당량
저장된 쿼리	계정당 지역당 최대 1,000개의 CloudWatch Logs Insights 쿼리를 저장할 수 있습니다. 이 할당량은 변경할 수 없습니다.
구독 필터	로그 그룹당 2개. 이 할당량은 변경할 수 없습니다.

CloudWatch 로그 서비스 할당량 관리

CloudWatch Logs는 중앙 위치에서 할당량을 보고 관리할 수 있는 AWS 서비스인 Service Quotas와 통합되었습니다. 자세한 내용은 Service Quotas 사용 설명서의 Service Quotas는 무엇인가요?를 참조하세요.

Service Quotas를 사용하면 로그 서비스 할당량의 가치를 쉽게 조회할 CloudWatch 수 있습니다.

AWS Management Console

콘솔을 사용하여 CloudWatch 로그 서비스 할당량을 보려면

1. https://console.aws.amazon.com/servicequotas/에서 Service Quotas 콘솔을 엽니다.
2. 탐색 창에서 AWS 서비스를 선택합니다.
3. AWS 서비스 목록에서 Amazon CloudWatch Logs를 검색하여 선택합니다.

   Service quotas 목록에서 서비스 할당량 이름, 적용된 값(제공된 경우), AWS 기본 할당량 및 할당량 값 조정 가능 여부를 확인할 수 있습니다.
4. 설명 등 서비스 할당량에 대한 추가 정보를 보려면 할당량 이름을 선택합니다.
5. (선택 사항) 할당량 증가를 요청하려면 증가시킬 할당량을 선택하고 할당량 증가 요청(Request quota increase)을 선택한 다음 필요한 정보를 입력하거나 선택한 다음 요청(Request)을 선택합니다.

콘솔을 사용하여 서비스 할당량에 대한 추가 작업을 수행하려면 Service Quotas 사용 설명서를 참조하세요. 할당량 증가를 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

AWS CLI

를 사용하여 CloudWatch 로그 서비스 할당량을 보려면 AWS CLI

다음 명령을 실행하여 기본 CloudWatch 로그 할당량을 확인합니다.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code logs \
    --output table
```

를 사용하여 서비스 할당량을 자세히 알아보려면 Service Quotas 명령 참조서를 참조하십시오. AWS CLI AWS CLI 할당량 증가를 요청하려면 AWS CLI 명령 참조에서 `request-service-quota-increase` 명령을 참조하세요.

문서 기록

다음 표에는 2018년 6월에 시작되는 CloudWatch Logs User Guide의 각 릴리스에서 변경된 주요 내용이 설명되어 있습니다. 이 설명서에 대한 업데이트 알림을 받으려면 RSS 피드를 구독하면 됩니다.

변경 사항	설명	날짜
CloudWatch 자연어 쿼리 생성을 위한 Logs Insights 지원은 일반적으로 제공됩니다.	CloudWatch Logs Insights는 쿼리를 생성하고 업데이트하기 위한 자연어를 지원합니다. 자세한 내용은 자연어를 사용하여 CloudWatch Logs Insights 쿼리 생성 및 업데이트를 참조하십시오.	2024년 6월 20일
CloudWatchLogsReadOnlyAccess정책 업데이트	CloudWatch 이 정책을 사용하는 사용자가 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있도록 로그에 `cloudwatch:GenerateQuery` 권한이 추가되었습니다. CloudWatchLogsReadOnlyAccess	2023년 11월 26일
CloudWatchLogsFullAccess정책 업데이트	CloudWatch 이 정책을 사용하는 사용자가 자연어 프롬프트에서 CloudWatch Logs Insights 쿼리 문자열을 생성할 수 있도록 로그에 `cloudwatch:GenerateQuery` 권한이 추가되었습니다. CloudWatchLogsFullAccess	2023년 11월 26일
CloudWatch 로그는 로그 패턴 분석을 추가합니다.	CloudWatch 이제 로그는 Logs Insights 쿼리를 수행할 때마다 로그 이벤트의 패턴을 CloudWatch 스캔합니다. 자세	2023년 11월 26일

	한 내용은 패턴 분석을 참조하십시오.	
CloudWatch 로그는 로그 이상 탐지를 추가합니다.	로그 그룹에 대한 로그 이상 탐지기를 만들 수 있습니다. 예외 항목 탐지기는 로그 그룹에 수집된 로그 이벤트를 스캔하고 로그 데이터에서 이상을 찾습니다. 자세한 내용은 로그 이상 탐지를 참조하세요.	2023년 11월 26일
CloudWatch 로그에는 비교 기능이 추가되었습니다.	이제 CloudWatch Logs Insights를 사용하여 시간 경과에 따른 로그 이벤트의 변화를 비교할 수 있습니다. 자세한 내용은 이전 시간 범위와 비교 (diff) 를 참조하십시오.	2023년 11월 26일
CloudWatch 로그는 새 로그 클래스를 추가합니다.	CloudWatch 로그는 두 가지 로그 그룹 클래스를 지원하므로 자주 액세스하지 않는 로그에 대해 비용 효율적인 옵션을 사용할 수 있으며 실시간 모니터링 또는 기타 기능이 필요한 로그에 대해서도 모든 기능을 갖춘 옵션을 사용할 수 있습니다. 자세한 내용은 로그 클래스를 참조하세요.	2023년 11월 26일
CloudWatch Logs Insights는 자연어 쿼리 생성을 지원합니다.	CloudWatch Logs Insights는 쿼리를 생성하고 업데이트하기 위한 자연어를 지원합니다. 자세한 내용은 자연어를 사용하여 CloudWatch Logs Insights 쿼리 생성 및 업데이트를 참조하십시오.	2023년 11월 26일

CloudWatch 로그에는 Live Tail에 대한 정규 표현식 필터 패턴 구문 지원이 추가되었습니다.	이제 Live Tail 필터 패턴 내에서 유연한 정규식을 사용하여 필요에 맞게 검색 및 매칭 작업을 추가로 사용자 지정할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 필터 패턴 구문을 참조하십시오.	2023년 11월 13일
CloudWatch 로그에는 메트릭 필터, 구독 필터 및 필터 로그 이벤트에 대한 정규 표현식 필터 패턴 구문 지원이 추가되었습니다.	이제 필터 패턴 내에서 유연한 정규식을 사용하여 필요에 맞게 검색 및 매칭 작업을 추가로 사용자 지정할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 필터 패턴 구문을 참조하십시오.	2023년 9월 5일
CloudWatch Logs Insights는 패턴 명령을 추가합니다.	이제 CloudWatch Logs Insights 쿼리에서 패턴을 사용하여 로그 데이터를 패턴으로 자동 클러스터링할 수 있습니다. 패턴은 로그 필드 간에 반복되는 공유 텍스트 구조입니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 패턴을 참조하십시오.	2023년 7월 17일
CloudWatch Logs Insights는 dedup 명령을 추가합니다.	이제 CloudWatch Logs Insights 쿼리에서 중복 제거를 사용하여 지정한 필드의 특정 값을 기반으로 하는 중복 결과를 제거할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 dedup을 참조하십시오.	2023년 6월 20일

계정 수준 데이터 보호 정책	이제 계정 수준에서 데이터 보호 정책을 설정할 수 있습니다. 이러한 계정 수준 정책은 계정의 모든 로그 그룹에 있는 로그 이벤트의 민감한 정보를 감사하고 마스킹할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 마스킹을 통한 민감한 로그 데이터 보호 지원을 참조하십시오.	2023년 6월 8일
Live Tail 기능 추가됨	CloudWatch 로그에는 Live Tail 기능이 추가되어 로그를 수집할 때 검사하여 문제 해결에 도움이 될 수 있습니다. 필요에 따라 지정된 항을 기준으로 표시된 로그 이벤트 스트림을 필터링하고 지정된 항이 있는 로그 이벤트를 강조 표시할 수도 있습니다. 자세한 내용은 Live Tail을 사용하여 거의 실시간으로 로그 보기를 참조하세요.	2023년 6월 6일

CloudWatchLogsRead OnlyAccess정책 업데이트	CloudWatch 에 추가된 권한을 CloudWatchLogsRead OnlyAccess기록합니다. 이 정책을 사용하는 사용자가 콘솔을 사용하여 CloudWatch Logs live tail 세션을 시작 및 중지할 수 있도록 및 `logs:StopLiveTail` 권한이 추가되었습니다. `logs:StartLiveTail` 자세한 내용은 Live Tail을 사용하여 거의 실시간으로 로그 보기를 참조하세요.	2023년 6월 6일
CloudWatch 로그 인사이트 출시	CloudWatch Logs Insights를 사용하여 대화형 방식으로 로그 데이터를 검색하고 분석할 수 있습니다. 자세한 내용은 Amazon Logs 사용 설명서의 CloudWatch Logs Insights를 사용한 CloudWatch 로그 데이터 분석을 참조하십시오.	2018년 11월 27일
Amazon VPC 엔드포인트에 대한 지원	이제 VPC와 CloudWatch Logs 사이에 프라이빗 연결을 설정할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 인터페이스 VPC 엔드포인트가 있는 CloudWatch 로그 사용을 참조하십시오.	2018년 6월 28일

다음 표에는 Amazon CloudWatch Logs 사용 설명서의 중요한 변경 사항이 설명되어 있습니다.

변경 사항	설명	릴리스 날짜
인터페이스 VPC 엔드포인트	일부 지역에서는 인터페이스 VPC 엔드포인트를 사용하여 Amazon VPC와 CloudWatch Logs 간의 트래픽이 Amazon 네트워크를 벗어나지 않도록 할 수 있습니다. 자세한 내용은 인터페이스 CloudWatch VPC 엔드포인트와 함께 로그 사용 섹션을 참조하세요.	2018년 3월 7일
Route 53 DNS 쿼리 로그	CloudWatch 로그를 사용하여 Route 53에서 수신한 DNS 쿼리에 대한 로그를 저장할 수 있습니다. 자세한 내용은 Amazon Route 53 개발자 안내서의 아마존 CloudWatch 로그란 무엇입니까? 또는 DNS 쿼리 로깅을 참조하세요.	2017년 9월 7일
로그 그룹 태그 지정	태그를 사용하여 로그 그룹을 분류할 수 있습니다. 자세한 정보는 Amazon Logs의 태그 CloudWatch 로그 그룹 섹션을 참조하세요.	2016년 12월 13일
콘솔 개선 사항	지표 그래프부터 관련 로그 그룹까지 검색할 수 있습니다. 자세한 정보는 지표에서 로그로 피벗 적용 섹션을 참조하세요.	2016년 11월 7일
콘솔 활용도 개선	보다 손쉬운 검색, 필터링 및 문제 해결을 위해 경험을 개선했습니다. 예를 들어 날짜 및 시간 범위로 로드 데이터를 필터링할 수 있습니다. 자세한 정보는 Logs로 전송된 로그 데이터 보기 CloudWatch 섹션을 참조하세요.	2016년 8월 29일
Amazon CloudWatch Logs 및 새 CloudWatch Logs 지표에 대한 AWS CloudTrail 지원	CloudWatch 로그에 대한 AWS CloudTrail 지원이 추가되었습니다. 자세한 정보는 로깅 CloudWatch 로그 API 및 콘솔 작업 AWS CloudTrail을 참조하세요.	2016년 3월 10일

변경 사항	설명	릴리스 날짜
이 추가되었습니다.		
Amazon S3로 CloudWatch 로그 내보내기에 대한 지원이 추가되었습니다.	Amazon S3로 CloudWatch 로그 데이터를 내보내는 지원이 추가되었습니다. 자세한 정보는 Amazon S3로 로그 데이터 내보내기을 참조하세요.	2015년 12월 7일
Amazon CloudWatch Logs에 AWS CloudTrail 기록된 이벤트에 대한 지원이 추가되었습니다.	에서 캡처한 특정 API 활동에 대한 알림을 CloudWatch 생성하고 수신한 후 알림을 사용하여 문제 해결을 수행할 수 있습니다. CloudTrail	2014년 11월 10일
Amazon CloudWatch 로그에 대한 지원 추가	Amazon CloudWatch Logs를 사용하여 Amazon Elastic Compute Cloud (Amazon EC2) 인스턴스 또는 기타 소스에서 시스템, 애플리케이션 및 사용자 지정 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. 그런 다음 Amazon CloudWatch 콘솔, 의 CloudWatch Logs 명령 또는 Logs SDK를 사용하여 CloudWatch Logs에서 관련 CloudWatch 로그 데이터를 검색할 수 있습니다. AWS CLI자세한 정보는 아마존 CloudWatch 로그란 무엇입니까?을 참조하세요.	2014년 7월 10일

AWS 용어집

최신 AWS 용어는 참조의 AWS 용어집을 참조하십시오.AWS 용어집

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.