사용 설명서

Microsoft Windows용 Amazon Kinesis 에이전트

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

Microsoft Windows용 Amazon Kinesis 에이전트: 사용 설명서

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

Amazon의 상표 및 트레이드 드레스는 Amazon이 제공하지 않는 제품 또는 서비스와 관련하여 고객에게 혼동을 유발할 수 있는 방식 또는 Amazon을 폄하하거나 평판에 악영향을 주는 방식으로 사용될 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계 여부에 관계없이 해당 소유자의 자산입니다.

Table of Contents

Windows용 Kinesis 에이전트란 무엇입니까? 1
- AWS 3
- Windows용 Kinesis 에이전트로 무엇을 할 수 있습니까? 3
- Benefits 5
- Windows용 Kinesis 에이전트 시작하기 8

Windows용 Kinesis 에이전트 개념 9
- Data Pipeline 10
- Sources 11
- Sinks 11
- Pipes 11

시작하기 12
- Prerequisites 12
- AWS 계정 설정 13
- 윈도우용 Kinesis 에이전트 설치 15
  - MSI를 사용하여 Windows용 Kinesis 에이전트 설치 16
  - AWS Systems Manager 사용하여 Windows용 Kinesis 에이전트 설치 16
  - PowerShell 을 사용하여 Windows용 Kinesis 에이전트 설치 18
- Windows용 Kinesis 에이전트 구성 및 시작 21

Windows용 Kinesis 에이전트 구성 23
- 기본 구성 구조 23
  - 구성 대소문자 구분 25
- 소스 선언 25
  - 디렉토리소스 구성 26
  - Exchange 로그소스 구성 38
  - W3SV클로그소스 구성 39
  - UlsSource 구성 40
  - 창 세븐로그소스 구성 41
  - 창 세븐로그폴링소스 구성 43
  - 창 집합소스 구성 45
  - 창기능카운터 소스 구성 47
  - Windows용 Kinesis 에이전트 기본 제공 메트릭 소스 50
  - Windows용 Kinesis 에이전트 메트릭 목록 52
  - 책갈피 구성 57
- sink 선언 58

KinesisStreamsink 구성 ... 61
KinesisFirehosesink 구성 ... 62
CloudWatch 싱크 구성 ... 63
CloudWatchLogssink 구성 ... 64
로컬FileSystemsink 구성 ... 65
sink 보안 구성 ... 67
구성ProfileRefreshingAWSCredentialProviderAWS 자격 증명 새로 고침 ... 73
싱크 장식 구성 ... 74
싱크 변수 대체 구성 ... 79
싱크대 대기열 구성 ... 80
싱크에 대한 프록시 구성 ... 80
더 많은 싱크 속성에서 해석 변수 구성 ... 81
AWS 싱크에서 RoleARN 속성을 사용할 때 AWS STS 리전 엔드포인트 구성 ... 81
AWS 싱크용 VPC 엔드포인트 구성 ... 81
대체 프록시 수단 구성 ... 82
파이프 선언 ... 82
파이프 구성 ... 83
Windows 메트릭 파이프용 Kinesis 에이전트 구성 ... 84
자동 업데이트 구성 ... 85
Windows용 Kinesis 에이전트 구성 예 ... 90
다양한 소스에서 Kinesis Data Streams 으로 스트리밍 ... 90
Windows 응용 프로그램 이벤트 로그에서 싱크로 스트리밍 ... 97
파이프 사용 ... 98
여러 소스 및 파이프 사용 ... 99
원격 분석 구성 ... 101
자습서: Amazon S3 로 JSON 로그 파일 스트리밍 ... 103
단계 1: AWS 서비스 구성 ... 103
IAM 정책 및 역할 구성 ... 104
Amazon S3 버킷을 생성합니다. ... 109
Kinesis Data Firehose 전송 스트림 생성 ... 109
Windows용 Kinesis 에이전트를 실행하기 위한 Amazon EC2 인스턴스 생성 ... 114
다음 단계 ... 114
단계 2: Windows용 Kinesis 에이전트 설치, 구성 및 실행 ... 114
다음 단계 ... 118
단계 3: Amazon S3 에서 로그 데이터 쿼리 ... 118
다음 단계 ... 121

문제 해결 123
데스크톱 또는 서버에서 예상 AWS 서비스로 스트리밍되는 데이터가 없음 123
Symptoms 123
Causes 123
Resolutions 124
적용 대상 129
예상 데이터가 때때로 누락됨 129
Symptoms 129
Causes 129
Resolutions 129
적용 대상 130
데이터가 잘못된 형식으로 도착함 130
Symptoms 130
Causes 130
Resolutions 130
적용 대상 131
성능 문제 131
Symptoms 131
Causes 131
Resolutions 132
적용 대상 135
디스크 공간 부족 135
Symptoms 135
Causes 135
Resolutions 135
적용 대상 136
도구 문제 해결 136
플러그인 생성 139
윈도 플러그인용 Kinesis 에이전트 시작하기 139
Windows 플러그인 팩토리를 위한 Kinesis 에이전트 구현 140
Windows 플러그인 소스용 Kinesis 에이전트 구현 143
Windows 플러그인 싱크용 Kinesis 에이전트 구현 146
문서 기록 151
AWS 용어집 152
..... cliii

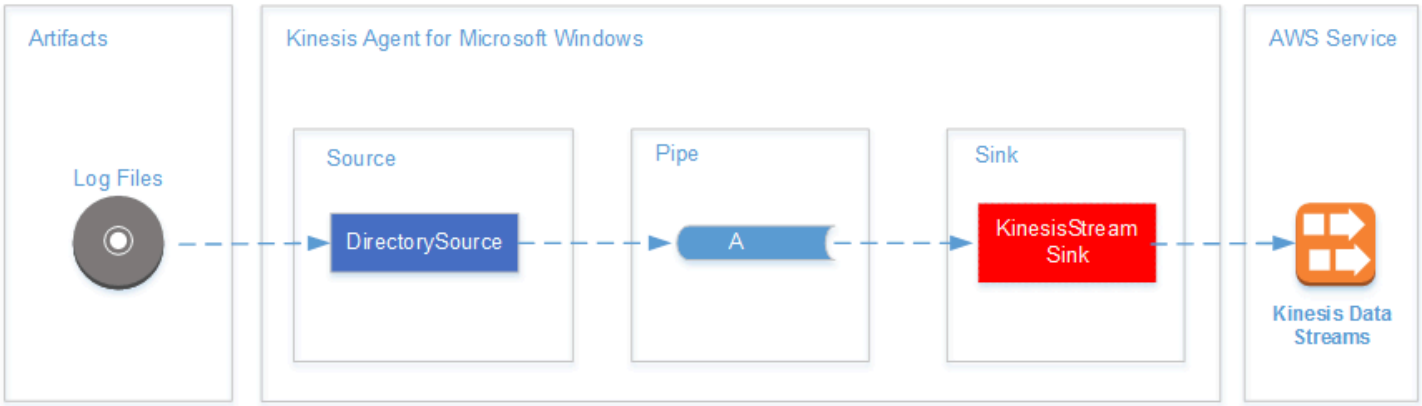
마이크로소프트 윈도우용 Amazon Kinesis 에이전트란 무엇입니까?

마이크로소프트 윈도우용 Amazon Kinesis 에이전트 (윈도우용 Kinesis 에이전트) 는 구성 가능하고 확장 가능한 에이전트입니다. 또한 온프레미스 또는 AWS 클라우드의 Windows 데스크톱 컴퓨터 및 서버에서 실행됩니다. Kinesis for Windows용 에이전트는 로그, 이벤트 및 메트릭을 효율적이고 안정적으로 수집, 구문 분석, 변환 및 스트리밍합니다.Kinesis Data Streams,Kinesis Data Firehose,Amazon CloudWatch, 및CloudWatch Logs(CloudWatch 로그).

그런 다음 이러한 서비스를 통해 다음을 비롯한 다양한 AWS 서비스를 사용하여 데이터를 저장, 분석 및 시각화할 수 있습니다.

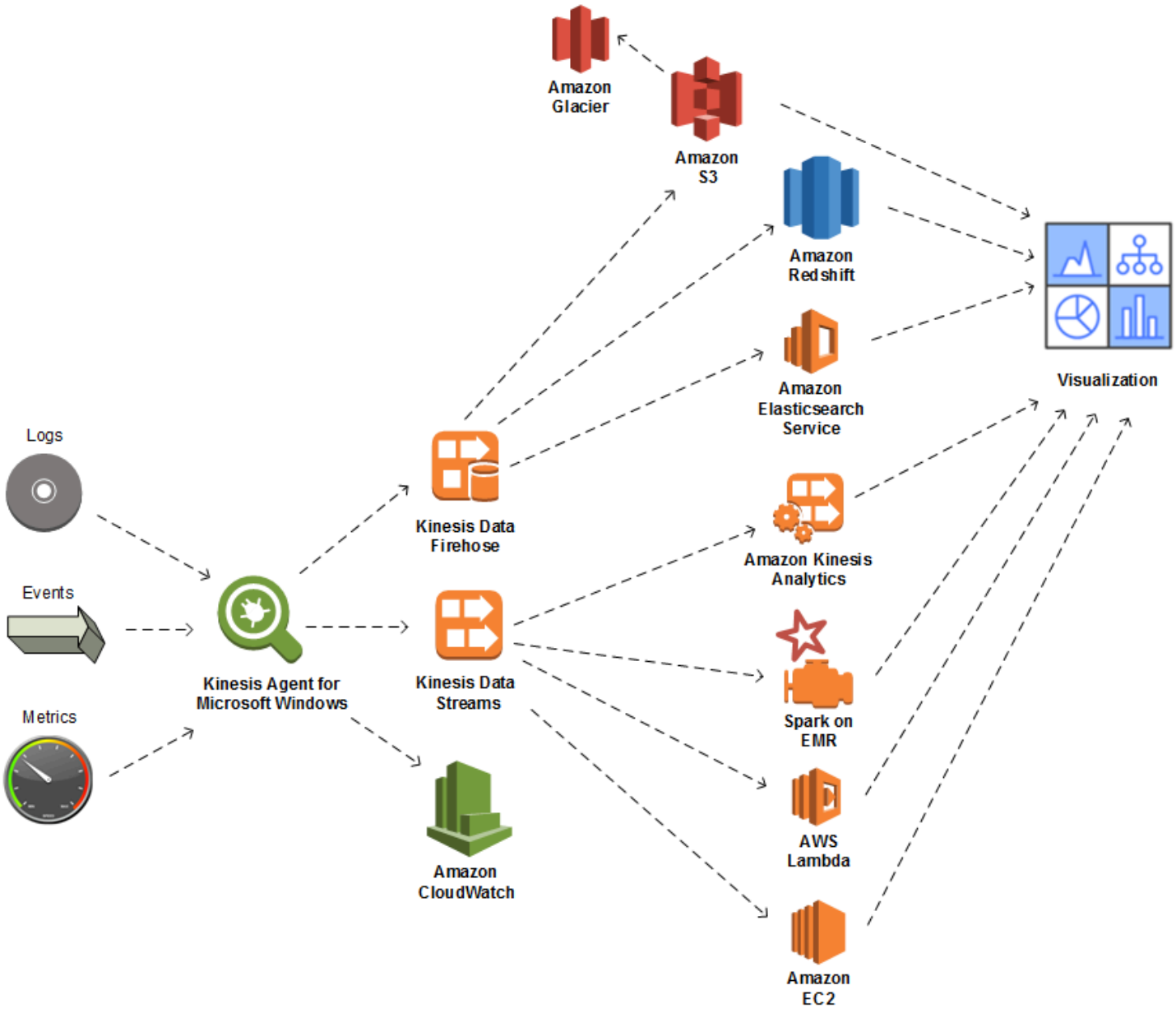
- Amazon Simple Storage Service(Amazon S3)
- Amazon Redshift
- Amazon Elasticsearch Service (Amazon ES)
- Kinesis Data Analytics
- Amazon QuickSight
- Amazon Athena
- Kibana

다음 다이어그램은 Kinesis 데이터 스트림에 로그 파일을 스트리밍하는 Windows용 Kinesis 에이전트의 간단한 구성을 보여줍니다.



소스, 파이프 및 싱크에 대한 자세한 내용은 단원을 참조하십시오.마이크로소프트 윈도우용 Amazon Kinesis 에이전트 개념.

다음 다이어그램에서는 스트림 처리 프레임워크를 사용하여 사용자 지정 실시간 데이터 파이프라인을 빌드할 수 있는 몇 가지 방법을 보여 줍니다. 이러한 프레임워크에는 Kinesis Data Analytics, Amazon EMR 의 아파치 스파크 및 AWS Lambda 가 포함됩니다.



주제

- AWS
- Windows용 Kinesis 에이전트로 무엇을 할 수 있습니까?
- Benefits
- Windows용 Kinesis 에이전트 시작하기

AWS

AWS (Amazon Web Services) 는 애플리케이션을 개발할 때 사용할 수 있는 디지털 인프라 서비스의 컬렉션입니다. 이 서비스에는 컴퓨팅, 스토리지, 데이터베이스, 분석, 애플리케이션 동기화 (메시징 및 대기열) 가 있습니다. AWS는 "사용한 만큼 지불" 서비스 모델을 사용합니다. 이 경우, 또는 애플리케이션에서 사용한 서비스에 대해서만 청구됩니다. 또한 서비스를 프로토타입 생성 및 실험용으로 더욱 쉽게 이용할 수 있도록 프리 티어를 제공합니다. 이 계층에서 특정 사용 수준 미만의 서비스는 무료입니다. AWS 비용에 대한 자세한 내용은프리 티어에 대한 자세한 내용은리소스 센터 시작하기. AWS 계정을 만들려면AWS 홈 페이지을 방문하여 가입하십시오.

Windows용 Kinesis 에이전트로 무엇을 할 수 있습니까?

Windows용 Kinesis 에이전트는 다음과 같은 기능을 제공합니다.

로그, 이벤트 및 메트릭 데이터 수집

Kinesis Agent는 서버 및 데스크톱 집합에서 하나 이상의 AWS 서비스로 로그, 이벤트 및 메트릭을 수집, 구문 분석, 변환 및 스트리밍합니다. 서비스에서 수신 한 페이로드는 원래 소스와 다른 형식일 수 있습니다. 예를 들어 로그는 특정 텍스트 형식 (예: syslog 형식) 으로 서버에 저장될 수 있습니다. Windows용 Kinesis 에이전트는 해당 텍스트를 수집 및 구문 분석하고 필요에 따라 JSON 형식으로 변환할 수 있습니다 (예: AWS 로 스트리밍하기 전에). 이렇게 하면 JSON을 사용하는 일부 AWS 서비스에서 간편하게 처리할 수 있습니다. Kinesis Data Streams 으로 스트리밍되는 데이터는 KinKinesis Data Analytics 지속적으로 처리하여 추가 지표와 집계된 지표를 생성할 수 있으며, 이를 통해 라이브 대시보드를 강화할 수 있습니다. 데이터 파이프라인에서 다운스트림에 사용되는 방식에 따라 다양한 AWS 서비스 (예: Amazon S3) 를 사용하여 데이터를 저장할 수 있습니다.

AWS 서비스와 통합

다음과 같은 여러 AWS 서비스에 로그 파일, 이벤트 및 메트릭을 전송하도록 Windows용 Kinesis 에이전트를 구성할 수 있습니다.

- Kinesis Data Firehose— Amazon S3, Amazon Redshift, Amazon ES 또는Splunk를 사용하여 추가 분석을 수행합니다.
- Kinesis Data Streams— Kinesis 데이터 분석 또는 Apache Spark에서 호스팅되는 사용자 지정 애플리케이션을 사용하여 스트리밍 데이터를 처리합니다.Amazon EMR. 또는에서 실행되는 사용자 정의 코드를 사용하십시오.Amazon EC2인스턴스 또는 사용자 지정 서버리스 함수AWS Lambda.
- CloudWatch— 스트리밍된 지표를 그래프로 보고 대시보드로 결합할 수 있습니다. 그런 다음 사전 설정된 임계값을 위반하는 지표 값에 의해 트리거되는 CloudWatch 경보를 설정합니다.
- CloudWatch Logs(CloudWatch 로그)— 스트리밍된 로그 및 이벤트를 저장하고, AWS Management Console에서 이를 보고 검색하거나, 데이터 파이프라인에서 다운스트림을 추가로 처리할 수 있습니다.

신속한 설치 및 구성

몇 단계만으로 Windows용 Kinesis 에이전트를 설치하고 구성할 수 있습니다. 자세한 내용은 윈도우용 Kinesis 에이전트 설치 및 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성 섹션을 참조하세요. 간단한 선언적 구성 파일은 다음을 지정합니다.

- 수집할 로그, 이벤트 및 메트릭의 소스 및 형식
- 수집된 데이터에 적용할 변환입니다. 추가 데이터를 포함할 수 있으며 기존 데이터를 변환하고 필터링할 수 있습니다.
- 최종 데이터가 스트리밍되는 대상과 스트리밍 페이로드의 버퍼링, 샤딩 및 형식입니다.

Windows용 Kinesis 에이전트에는 다음과 같은 일반적인 Microsoft 엔터프라이즈 서비스에서 생성된 로그 파일에 대한 파서가 내장되어 있습니다.

- Microsoft Exchange
- SharePoint
- Active Directory 도메인
- DHCP 서버

지속적인 관리

Windows용 Kinesis 에이전트는 데이터 손실 없이 다양한 상황에 자동으로 적응합니다. 여기에는 로그 순환, 재부팅 후 복구, 일시적인 네트워크 또는 서비스 중단이 포함됩니다. Windows용 Kinesis 에이전트가 새 버전으로 자동으로 업데이트되도록 구성할 수 있습니다. 이러한 상황에서는 운영자의 개입이 필요하지 않습니다.

개방형 아키텍처를 사용하여 확장

선언적 기능과 내장 플러그인이 서버 또는 데스크톱 시스템을 모니터링하기에 충분하지 않은 경우 플러그인을 생성하여 Kinesis Windows용 에이전트를 확장할 수 있습니다. 새로운 플러그인을 사용하면 로그, 이벤트 및 메트릭에 대한 새로운 소스와 대상을 사용할 수 있습니다. Windows용 Kinesis 에이전트의 소스 코드는https://github.com/awslabs/kinesis-agent-windows.

Benefits

Windows용 Kinesis 에이전트는 데이터 파이프라인에 대한 로그, 이벤트 및 메트릭에 대한 초기 데이터 수집, 변환 및 스트리밍을 수행합니다. 이러한 데이터 파이프라인을 구축하면 다음과 같은 다양한 이점이 있습니다.

분석 및 시각화

Kinesis Data Firehose와 Windows용 Kinesis 에이전트의 통합 및 변환 기능을 통해 다음과 같은 다양한 분석 및 시각화 서비스와 쉽게 통합할 수 있습니다.

- Amazon QuickSight— 다양한 소스에서 수집할 수 있는 클라우드 기반 BI 서비스입니다. Windows용 Kinesis 에이전트는 데이터를 변환하여 KinKinesis Data Firehose 스를 통해 Amazon S3 및 Amazon

Redshift로 스트리밍할 수 있습니다. 이 프로세스를 통해 Amazon QuickSight 시각화를 사용하여 데이터에서 심층적인 인사이트를 검색할 수 있습니다.

- Athena— SQL 기반 데이터 쿼리를 가능하게 하는 대화형 쿼리 서비스입니다. Windows용 Kinesis 에이전트는 Kinesis 데이터 파이어호스를 통해 데이터를 변환하고 Amazon S3 로 스트리밍할 수 있습니다. 그런 다음 Athena 해당 데이터에 대해 대화식으로 SQL 쿼리를 실행하여 로그와 이벤트를 신속하게 검사하고 분석할 수 있습니다.
- Kibana— 오픈 소스 데이터 시각화 도구입니다. Windows용 Kinesis 에이전트는 Kinesis 데이터 파이어호스를 통해 데이터를 변환하고 Amazon ES로 스트리밍할 수 있습니다. 그런 다음 Kibana를 사용하여 해당 데이터를 탐색할 수 있습니다. 히스토그램, 선 그래프, 파이 차트, 열 지도, 지리 공간 그래픽을 비롯한 다양한 시각화를 만들고 엽니다.

Security

Kinesis Agent for Windows가 포함된 로그 및 이벤트 데이터 분석 파이프라인은 조직의 보안 침해를 감지하고 경고하여 공격을 차단하거나 중지하는 데 도움이 됩니다.

애플리케이션 성능

Windows용 Kinesis 에이전트는 애플리케이션 또는 서비스 성능에 대한 로그, 이벤트 및 메트릭 데이터를 수집할 수 있습니다. 그러면 전체 데이터 파이프라인이 이 데이터를 분석할 수 있습니다. 이 분석은 분명하지 않을 수 있는 결함을 감지하고 보고하여 응용 프로그램 및 서비스 성능 및 안정성을 개선하는 데 도움이 됩니다. 예를 들어 서비스 API 호출의 실행 시간에 중요한 변경 사항을 감지할 수 있습니다. 배포와 상관 관계가 있는 경우 이 기능을 사용하면 소유한 서비스의 새로운 성능 문제를 찾아 해결할 수 있습니다.

서비스 작업

데이터 파이프라인은 수집된 데이터를 분석하여 잠재적인 운영 문제를 예측하고 서비스 중단을 방지하는 방법에 대한 통찰력을 제공할 수 있습니다. 예를 들어 로그, 이벤트 및 메트릭을 분석하여 현재 및 예상 용량 사용량을 파악하여 서비스 사용자에게 영향을 미치기 전에 추가 용량을 온라인으로 전환할 수 있습니다. 서비스 중단이 발생하면 데이터를 분석하여 중단 기간 동안 고객에게 미치는 영향을 파악할 수 있습니다.

Auditing

데이터 파이프라인은 Windows용 Kinesis 에이전트가 수집하고 변환하는 로그, 이벤트 및 메트릭을 처리할 수 있습니다. 그런 다음 다양한 AWS 서비스를 사용하여 이 처리된 데이터를 감사할 수 있습니다. 예를 들어, Kinesis Data Firehose 는 Amazon S3 데이터를 저장하는 Windows용 Kinesis 에이전트로부터 데이터 스트림을 수신할 수 있습니다. 그런 다음 Athena 를 사용하여 대화형 SQL 쿼리를 실행하여 이 데이터를 감사할 수 있습니다.

Archiving

종종 가장 중요한 운영 데이터는 최근에 수집 된 데이터입니다. 그러나 몇 년 동안 응용 프로그램 및 서비스에 대해 수집 된 데이터 분석은 예를 들어 장거리 계획과 같이 유용 할 수 있습니다. 대용량 데이터를 유지하는 데 많은 비용이 소요될 수 있습니다. Windows용 Kinesis 에이전트는 Kinesis 데이터 파이어호스를 통해 Amazon S3 데이터를 수집, 변환 및 저장할 수 있습니다. 따라서Amazon S3 Glacier를 사용하여 오래된 데이터를 보관하는 비용을 절감할 수 있습니다.

Alerting

Windows용 Kinesis 에이전트는 메트릭을 CloudWatch 스트리밍합니다. 그런 다음 CloudWatch 경보를 생성하여 알림을 전송할 수 있습니다.Amazon Simple Notification Service (Amazon SNS)측정 단위가

특정 임계값을 일관되게 위반하는 경우 이를 통해 엔지니어는 응용 프로그램 및 서비스와 관련된 운영 문제를 보다 잘 인식할 수 있습니다.

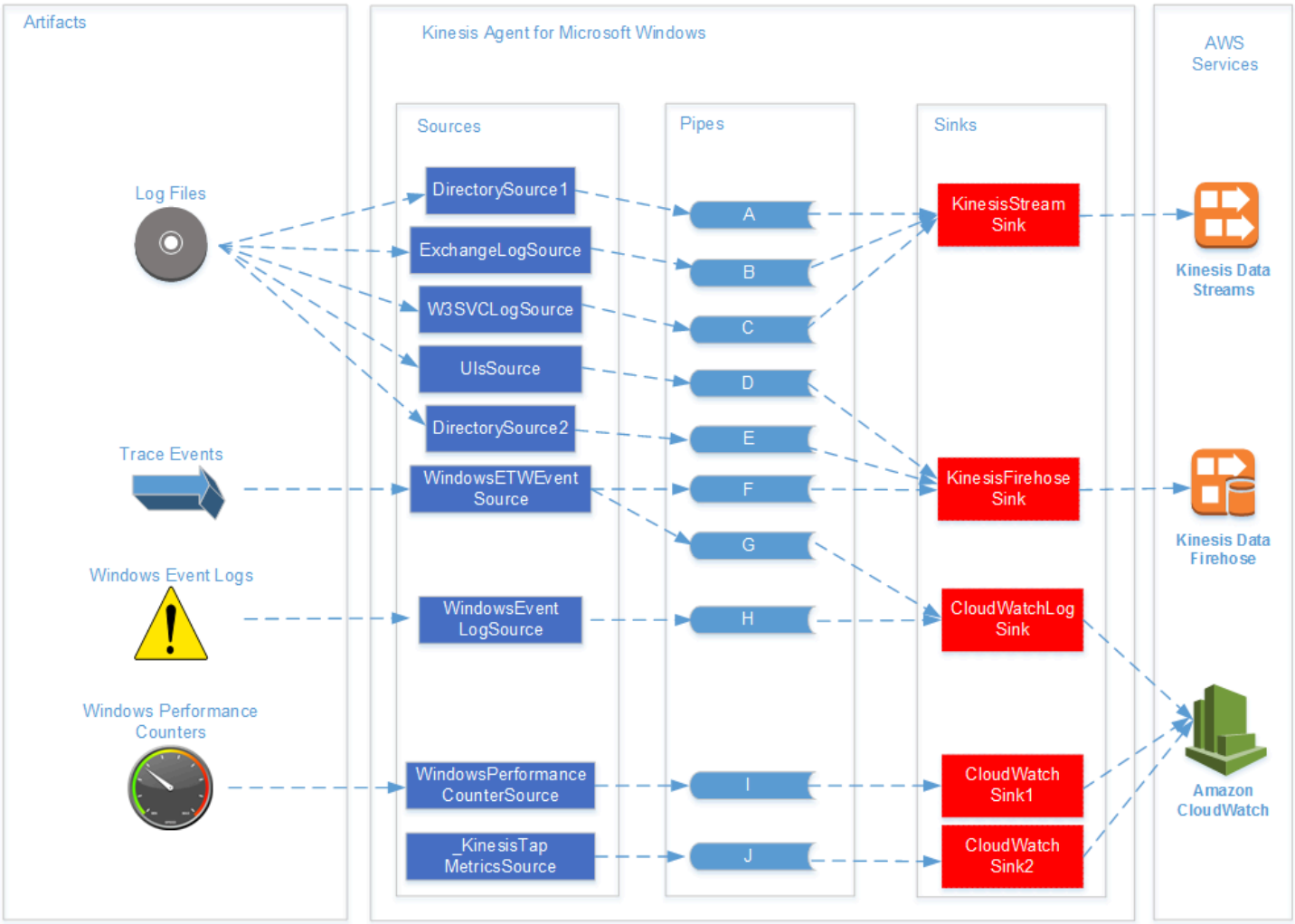
Windows용 Kinesis 에이전트 시작하기

Windows용 Kinesis 에이전트에 대해 자세히 알아보려면 다음 단원을 시작할 것을 권장합니다.

- 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 개념
- Microsoft Windows용 Amazon Kinesis 에이전트 시작하기

마이크로소프트 윈도우용 Amazon Kinesis 에이전트 개념

Microsoft Windows용 Amazon Kinesis 에이전트 (Windows용 Kinesis 에이전트) 의 주요 개념을 이해하면 데스크톱 및 서버 함대의 데이터를 데이터 파이프라인의 나머지 부분까지 더 쉽게 수집하고 스트리밍할 수 있습니다.



이 데이터 파이프라인 다이어그램은 다음 구성 요소 및 프로세스를 보여 줍니다.

서버 및 데스크톱에는 하나 이상의 Windows용 Kinesis 에이전트에서 수집한 로그 파일, 이벤트 및 메트릭과 같은 아티팩트가 있습니다.sources. 데이터를 선택적으로 플랫 파일 텍스트 형식에서 객체로 변환할 수 있습니다.

그러면 객체 또는 텍스트 형식으로 데이터가 하나 이상의 Windows용 Kinesis 에이전트로 흐를 수 있습니다.파이프. 파이프는 하나의 소스를 하나의 Windows용 Kinesis 에이전트에 연결합니다.sink. 파이프는 선택적으로 불필요한 데이터를 필터링할 수 있습니다.

싱크는 선택적으로 JSON 또는 XML로 객체로 구문 분석 된 데이터를 변환 할 수 있습니다. 싱크는 Kinesis 데이터 스트림, Kinesis 데이터 파이어호스 또는 Amazon CloudWatch 와 같은 특정 AWS 서비스로 데이터를 전송합니다.

여러 파이프를 사용하면 단일 소스에서 동일한 데이터를 여러 싱크에 보낼 수 있습니다 (예: 파이프F및G다이어그램에서). 여러 파이프를 사용하면 여러 소스에서 데이터를 단일 싱크로 스트리밍할 수 있습니다 (예: 파이프A,B, 및C다이어그램에서). 여러 개의 파이프를 사용하여 여러 싱크에서 여러 소스로 데이터를 스트리밍할 수도 있습니다. 소스, 싱크 및 파이프에는 유형이 있으며 동일한 유형의 소스, 싱크 또는 파이프가 둘 이상 있을 수 있습니다.

원본, 싱크 및 파이프를 선언하는 구성 파일의 예는Windows용 Kinesis 에이전트 구성 예.

주제

- Data Pipeline
- Sources
- Sinks
- Pipes

Data Pipeline

A데이터 파이프라인는 응용 프로그램 및 서비스에 대한 경보를 수집, 처리, 시각화 및 생성하는 데 사용됩니다. Windows용 Kinesis 에이전트는 데스크톱 컴퓨터 또는 서버에서 로그, 이벤트 및 메트릭이 수집되는 시작 시 데이터 파이프라인에 적합합니다. Windows용 Kinesis 에이전트는 수집된 데이터를 나머지 데이터 파이프라인을 구성하는 다양한 AWS 서비스로 스트리밍합니다. 데이터 파이프라인은 엔지니어가 해당 서비스를 보다 효과적으로 운영할 수 있도록 특정 서비스의 상태를 실시간으로 시각화하는 등의 목적을 가지고 있습니다. 서비스 상태 데이터 파이프라인은 다음 중 무엇이든 수행할 수 있습니다.

- 이러한 문제가 서비스 고객의 경험에 영향을 미치기 전에 엔지니어에게 문제를 알립니다.
- 리소스 사용 추세를 표시하여 엔지니어가 서비스 비용을 효율적으로 관리할 수 있도록 지원합니다. 이러한 추세를 통해 리소스 수준을 적절하게 조정하거나 자동 조정 시나리오를 구현할 수도 있습니다.
- 서비스 고객이 보고한 문제의 근본 원인에 대한 통찰력을 제공합니다. 이렇게 하면 이러한 문제의 해결 속도가 빨라지고 지원 비용이 절감됩니다.

Windows용 Kinesis 에이전트를 사용하여 데이터 파이프라인을 만드는 단계별 예제는 를 참조합니다.자습서: 윈도우용 Kinesis 에이전트를 사용하여 JSON 로그 파일을 Amazon S3 로 스트리밍.

Sources

윈도우용 Kinesis 에이전트source는 로그, 이벤트 또는 메트릭을 수집합니다. 소스는 소스 유형에 따라 해당 데이터의 특정 생산자로부터 특정 종류의 데이터를 수집합니다. 예를 들어,`DirectorySource`유형은 파일 시스템의 특정 디렉토리에서 로그 파일을 수집합니다. 데이터가 아직 구조화되지 않은 경우 (일부 종류의 로그 파일과 마찬가지로) 소스는 텍스트 표현을 구조화 된 형식으로 구문 분석하는 데 유용 할 수 있습니다. 각 소스는 특정소스 선언윈도우용 Kinesis 에이전트에서`appsettings.json`파일 소스 선언은 특정 데이터 수집 요구 사항에 따라 소스를 맞춤화하기 위해 소스를 구성하는 데 필요한 필수 세부 정보를 제공합니다. 구성할 수 있는 세부 정보의 종류는 소스 유형에 따라 다릅니다. 예를 들어,`DirectorySource`소스 유형에는 로그 파일이 상주하는 디렉토리의 지정이 필요합니다.

소스 유형 및 소스 선언에 대한 자세한 내용은소스 선언.

Sinks

윈도우용 Kinesis 에이전트sink는 Kinesis for Windows용 에이전트 소스에서 수집한 데이터를 가져와 나머지 데이터 파이프라인을 구성하는 여러 가능한 AWS 서비스 중 하나로 해당 데이터를 스트리밍합니다. 각 싱크는 특정싱크 선언윈도우용 Kinesis 에이전트에서`appsettings.json`파일 싱크 선언은 특정 데이터 스트리밍 요구 사항에 따라 싱크를 맞추기 위해 싱크를 구성하기위한 필수 세부 사항을 제공합니다. 구성 할 수있는 세부 사항의 종류는 싱크 유형에 따라 다릅니다. 예를 들어, 일부 싱크 유형은 싱크 선언이 특정 직렬화를 지정할 수 있도록`Format`그들에게 제공된 데이터에 대한. 싱크 선언에서 이 옵션을 지정하면 데이터를 싱크와 연결된 AWS 서비스로 스트리밍하기 전에 수집된 데이터의 직렬화가 수행됩니다.

싱크 유형 및 싱크 선언에 대한 자세한 내용은sink 선언.

Pipes

윈도우용 Kinesis 에이전트파이프는 Windows용 Kinesis 에이전트 소스의 출력을 Windows용 Kinesis 에이전트 싱크의 입력에 연결합니다. 선택적으로 파이프를 통해 흐를 때 데이터를 변환합니다. 각 파이프는 Windows용 Kinesis 에이전트의 특정 파이프 선언에 해당합니다.`appsettings.json`파일 파이프 선언은 파이프의 소스 및 싱크와 같은 싱크 구성을 위한 필수 세부 정보를 제공합니다.

파이프 유형 및 파이프 선언에 대한 자세한 내용은파이프 선언.

Microsoft Windows용 Amazon Kinesis 에이전트 시작하기

Microsoft Windows용 Amazon Kinesis 에이전트 (Windows용 Kinesis 에이전트) 를 사용하여 Windows 집합에서 다양한 AWS 서비스로 로그, 이벤트 및 지표를 수집, 구문 분석, 변환 및 스트리밍할 수 있습니다. 다음 정보에는 Windows용 Kinesis 에이전트를 설치 및 구성하기 위한 필수 구성 요소 및 단계별 지침이 포함되어 있습니다.

주제

- Prerequisites
- AWS 계정 설정
- 윈도우용 Kinesis 에이전트 설치
- Windows용 Kinesis 에이전트 구성 및 시작

Prerequisites

Kinesis 에이전트를 설치하기 전에 다음 사전 조건을 확인하십시오.

- Windows용 Kinesis 에이전트 개념에 익숙합니다. 자세한 내용은 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 개념 섹션을 참조하세요.
- 데이터 파이프라인과 관련된 다양한 AWS 서비스를 사용하기 위한 AWS 계정입니다. AWS 계정 생성 및 구성에 대한 자세한 내용은 단원을 참조하십시오.AWS 계정 설정.
- Windows용 Kinesis 에이전트를 실행할 각 데스크톱 또는 서버에서 Microsoft .NET 프레임워크 4.6 이상 버전을 사용할 수 있습니다. 자세한 내용은 단원을 참조하십시오.개발자를위한 .NET Framework 설치에서 Microsoft .NET 설명서를 참조하십시오.

  데스크톱 또는 서버에 설치된.NET Framework의 최신 버전을 확인하려면 다음 PowerShell 스크립트를 사용하십시오.

```
[System.Version](
(Get-ChildItem 'HKLM:\SOFTWARE\Microsoft\NET Framework Setup\NDP' -recurse `
| Get-ItemProperty -Name Version -ErrorAction SilentlyContinue `
| Where-Object { ($_.PSChildName -match 'Full') } `
| Select-Object Version | Sort-Object -Property Version -Descending)[0]).Version
```

- Windows용 Kinesis 에이전트에서 데이터를 전송하려는 스트림 (Amazon Kinesis Data Streams 사용하는 경우) 사용 하 여 스트림 만들기Kinesis Data Streams 콘솔,AWS CLI또는Windows PowerShell용 AWS 도구. 자세한 내용은 단원을 참조하십시오.데이터 스트림 만들기 및 업데이트의Amazon Kinesis Data Streams 개발자 가이드.
- Windows용 Kinesis 에이전트로부터 데이터를 전송하려는 파이어호스 전송 스트림입니다 (Amazon Kinesis Data Firehose 사용하는 경우). 사용 하 여 전송 스트림 만들기Kinesis Data Firehose 콘솔,AWS CLI또는Windows PowerShell용 AWS 도구. 자세한 내용은 단원을 참조하십시오.Amazon Kinesis Data Firehose 배송 스트림 생성의Amazon Kinesis Data Firehose 개발자 가이드.

AWS 계정 설정

AWS 계정이 없는 경우 다음 절차에 따라 계정을 생성하십시오.

AWS 계정에 가입하려면

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따릅니다.

   등록 절차 중 전화를 받고 전화 키패드를 사용하여 확인 코드를 입력하는 과정이 있습니다.

관리자 사용자를 직접 생성하여 관리자 그룹에 추가하려면(콘솔)

1. 에 로그인합니다.IAM 콘솔를 선택하여 계정 소유자로루트 사용자를 클릭하고 AWS 계정 이메일 주소를 입력합니다. 다음 페이지에서 암호를 입력합니다.

   > **Note**
   >
   > 사용하는 모범 사례를 준수해 두는 것이 좋습니다.**Administrator**IAM 사용자는 루트 사용자 자격 증명을 준수하고 안전하게 보관해 둡니다. 몇 가지 계정 및 서비스 관리 태스크를 수행하려면 반드시 루트 사용자로 로그인해야 합니다.

2. 탐색 창에서 사용자와 사용자 추가를 차례로 선택합니다.
3. 사용자 이름에 **Administrator**를 입력합니다.
4. AWS Management Console 액세스(AWS Management Console access)옆에 있는 확인란을 선택합니다. 그런 다음 Custom password(사용자 지정 암호)를 선택하고 텍스트 상자에 새 암호를 입력합니다.

5. (선택 사항) 기본적으로 AWS에서는 새 사용자가 처음 로그인할 때 새 암호를 생성해야 합니다. User must create a new password at next sign-in(사용자가 다음에 로그인할 때 새 암호를 생성해야 합니다) 옆에 있는 확인란의 선택을 취소하면 새 사용자가 로그인한 후 암호를 재설정할 수 있습니다.
6. 선택다음: 권한.
7. 권한 설정 아래에서 그룹에 사용자 추가를 선택합니다.
8. 그룹 생성을 선택합니다.
9. 그룹 생성 대화 상자의 그룹 이름에 **`Administrators`**를 입력합니다.
10. 선택정책 필터링을 선택한 다음AWS 관리형 - 직무를 클릭하여 테이블 내용을 필터링합니다.
11. 정책 목록에서 AdministratorAccess 확인란을 선택합니다. 그런 다음 Create group을 선택합니다.

> **Note**
>
> `AdministratorAccess` 권한을 사용하여 AWS 결제 및 비용 관리 콘솔에 액세스하려면 먼저 결제에 대한 IAM 사용자 및 역할 액세스를 활성화해야 합니다. 이를 위해 결제 콘솔에 액세스를 위임하기 위한 자습서 1단계의 지침을 따르십시오.

12. 그룹 목록으로 돌아가 새로 만든 그룹 옆의 확인란을 선택합니다. 목록에서 그룹을 확인하기 위해 필요한 경우 Refresh(새로 고침)를 선택합니다.
13. 선택다음: 태그.
14. (선택 사항) 태그를 키-값 페어로 연결하여 메타데이터를 사용자에게 추가합니다. IAM에서의 태그 사용에 대한 자세한 내용은 단원을 참조하십시오.IAM 엔터티 태그 지정의IAM 사용 설명서.
15. 선택다음: 검토를 선택하여 새 사용자에 추가될 그룹 멤버십의 목록을 확인합니다. 계속 진행할 준비가 되었으면 Create user를 선택합니다.

이제 동일한 절차에 따라 그룹이나 사용자를 추가 생성하여 AWS 계정 리소스에 액세스할 수 있는 권한을 사용자에게 부여할 수 있게 되었습니다. 특정 AWS 리소스에 대한 사용자 권한을 제한하는 정책을 사용하는 방법을 알아보려면 액세스 관리 및 정책 예제를 참조하십시오.

AWS 에 가입하고 관리자 계정을 생성하려면 다음을 수행합니다.

1. AWS 계정이 없는 경우,https://aws.amazon.com/. 선택AWS 계정 생성를 선택한 다음 온라인 지시사항을 따릅니다.

   등록 절차 중 전화를 받고 전화 키패드를 사용하여 PIN을 입력하는 과정이 있습니다.

2. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
3. 탐색 창에서 그룹을 선택한 다음, 새 그룹 생성을 선택합니다.
4. 그룹 이름에 **Administrators**와 같은 그룹 이름을 입력한 후 다음 단계를 선택합니다.
5. 정책 목록에서 [AdministratorAccess] 정책 옆의 확인란을 선택합니다. 필터 메뉴와 검색 상자를 사용하여 정책 목록을 필터링합니다.
6. [Next Step]을 선택합니다. 선택그룹 생성을 클릭하면 새 그룹이그룹 이름.
7. 탐색 창에서 사용자와 Create New Users(새 사용자 만들기)를 차례대로 선택합니다.
8. 상자 내1옆에 있는 확인란의 선택을 취소하려면 사용자 이름을 입력하고각 사용자에 대해 액세스 키 생성를 선택한 다음 를 선택합니다.생성.
9. 사용자 목록에서 방금 생성한 사용자의 이름 (확인란 아님) 을 선택합니다. 다음을 수행할 수 있습니다.검색상자를 사용하여 사용자 이름을 검색합니다.
10. 를 선택합니다.그룹탭을 선택한 다음그룹에 사용자 추가.
11. 관리자 그룹 옆의 확인란을 선택한 다음그룹에 추가.
12. [Security Credentials] 탭을 선택합니다. [Sign-In Credentials]에서 [Manage Password]를 선택합니다.
13. Select사용자 정의 암호 할당에 암호를 입력합니다.Password및암호 확인상자를 클릭한 다음Apply.

윈도우용 Kinesis 에이전트 설치

다음과 같은 세 가지 방법으로 Windows용 Kinesis 에이전트를 설치할 수 있습니다.

- MSI (윈도우 설치 프로그램 패키지) 를 사용하여 설치합니다.
- 에서 설치AWS Systems Manager, 서버 및 데스크톱 관리를 위한 서비스 집합입니다.
- PowerShell 스크립트를 실행합니다.

> **Note**
>
> 다음 지침에서는 용어를 사용하는 경우가 있습니다.키네스탭및AW스키네시스탭. 이러한 단어는 Windows용 Kinesis 에이전트와 동일한 것을 의미하지만 이러한 명령을 실행할 때는 그대로 지정해야 합니다.

MSI를 사용하여 Windows용 Kinesis 에이전트 설치

최신 Kinesis 경우:GitHub 의 키네시스-에이전트 - 윈도우 저장소. MSI를 다운로드한 후 Windows를 사용하여 MSI를 시작하고 설치 프로그램의 프롬프트를 따릅니다. 설치 후에는 Windows 응용 프로그램과 마찬가지로 제거할 수 있습니다.

또는 다음을 수행할 수 있습니다.엠시엑서크명령을 실행하여 다음 예에 표시된 대로 로깅을 설정하고 제거합니다. Replace*AWSKinesisTap.1.1.216.4.msi* `with the appropriate version of Kinesis Agent for Windows for your application.`

Windows용 Kinesis 에이전트를 자동으로 설치하려면

```
msiexec /i AWSKinesisTap.1.1.216.4.msi /q
```

문제 해결을 위한 설치 메시지를***logfile.log***:

```
msiexec /i AWSKinesisTap.1.1.216.4.msi /q /L*V logfile.log
```

명령 프롬프트를 사용하여 Windows용 Kinesis 에이전트를 제거하려면

```
msiexec.exe /x {ADAB3982-68AA-4B45-AE09-7B9C03F3EBD3} /q
```

AWS Systems Manager 사용하여 Windows용 Kinesis 에이전트 설치

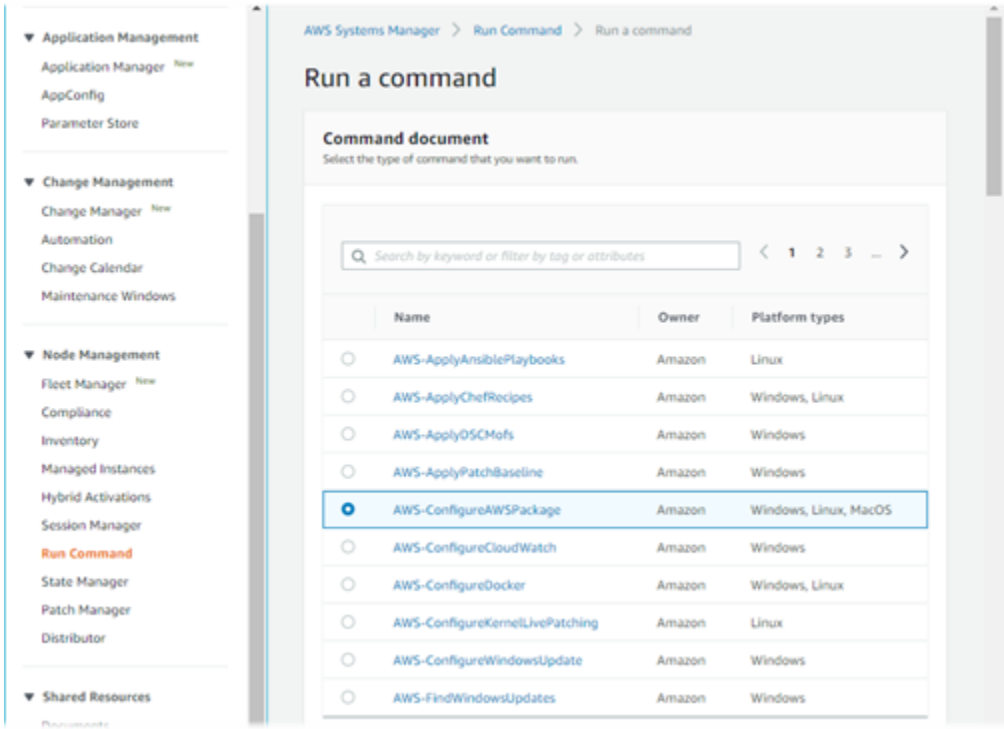
Systems Manager 실행 명령을 사용하여 Windows용 Kinesis 에이전트를 설치하려면 다음 단계를 수행하십시오. Run Command 단원을 참조하십시오.AWS Systems Manager 실행 명령의AWS Systems Manager 사용 설명서. Systems Manager 실행 명령을 사용하는 것 외에도 시스템 관리자를 사용할 수도 있습니다유지 관리 기간및상태 관리자를 사용하여 시간이 지남에 따라 Windows용 Kinesis 에이전트의 배포를 자동화할 수 있습니다.

> **Note**
>
> Windows용 Kinesis 에이전트용 Systems Manager 설치는AWS Systems Manager다음을 제외하고:
>
> - cn-north-1
> - cn-northwest-1

- 모든 AWS GovCloud 리전

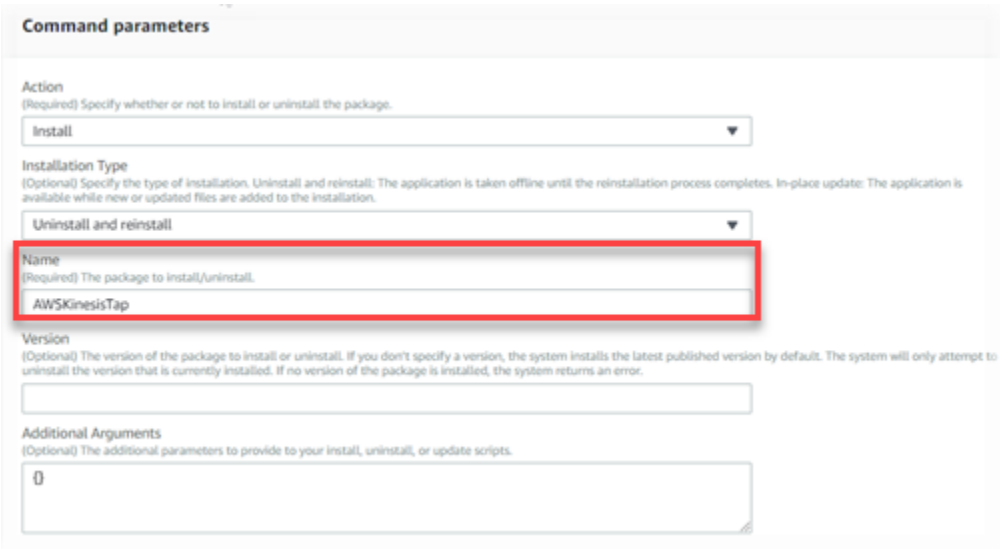
Systems Manager 사용하여 Windows용 Kinesis 에이전트를 설치하려면

1. Windows용 Kinesis 에이전트를 설치하려는 인스턴스에 SSM 에이전트 버전 2.2.58.0 이상이 설치되어 있는지 확인합니다. 자세한 내용은 단원을 참조하십시오.Windows 인스턴스에서 SSM 에이전트 설치 및 구성의AWS Systems Manager 사용 설명서.
2. AWS Systems Manager 콘솔을 엽니다.https://console.aws.amazon.com/systems-manager/.
3. 탐색 창의노드 관리를 선택하고명령 실행를 선택한 다음 를 선택합니다.명령 실행.
4. 에서명령 문서목록에서 [] 를 선택합니다.AWS-ConfigureAWSPackage문서를 참조하십시오.



5. 언더명령 파라미터에 대한이름를 입력합니다.AW스키네시스탭. 다른 설정은 기본값으로 그대로 둡니다.

> **Note**
>
> 나가기Version비워 두면 AWSkinesSpep 패키지의 최신 버전을 지정할 수 있습니다. 선택적으로 설치할 특정 버전을 입력할 수 있습니다.



6. 언더대상에서 명령을 실행할 인스턴스를 지정합니다. 인스턴스와 연결된 태그를 기반으로 인스턴스를 지정하도록 선택하거나, 인스턴스를 수동으로 선택하거나, 인스턴스를 포함하는 리소스 그룹을 지정할 수 있습니다.
7. 다른 모든 설정은 기본값으로 그대로 두고실행.

PowerShell 을 사용하여 Windows용 Kinesis 에이전트 설치

텍스트 편집기를 사용하여 다음 명령을 파일에 복사하고 PowerShell 스크립트로 저장합니다. 우리는`InstallKinesisAgent.ps1`다음 예제입니다.

```
Param(
    [ValidateSet("prod", "beta", "test")]
    [string] $environment = 'prod',
    [string] $version,
    [string] $baseurl
)

# Self-elevate the script if required.
if (-Not ([Security.Principal.WindowsPrincipal]
 [Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltI
 'Administrator')) {
    if ([int](Get-CimInstance -Class Win32_OperatingSystem | Select-Object -
ExpandProperty BuildNumber) -ge 6000) {
        $CommandLine = '-File "' + $MyInvocation.MyCommand.Path + '" ' +
 $MyInvocation.UnboundArguments
```

```
        Start-Process -FilePath PowerShell.exe -Verb Runas -ArgumentList $CommandLine
        Exit
    }
}

# Allows input to change base url. Useful for testing.
if ($baseurl) {
    if (!$baseUrl.EndsWith("/")) {
        throw "Invalid baseurl param value. Must end with a trailing forward slash
 ('/')"
    }

    $kinesistapBaseUrl = $baseurl
} else {
    $kinesistapBaseUrl = "https://s3-us-west-2.amazonaws.com/kinesis-agent-windows/
downloads/"
}

Write-Host "Using $kinesistapBaseUrl as base url"

$webClient = New-Object System.Net.WebClient

try {
    $packageJson = $webClient.DownloadString($kinesistapBaseUrl + 'packages.json' + '?
_t=' + [System.DateTime]::Now.Ticks) | ConvertFrom-Json
} catch {
    throw "Downloading package list failed."
}


if ($version) {
    $kinesistapPackage = $packageJson.packages | Where-Object { $_.packageName -eq
 "AWSKinesisTap.$version.nupkg" }

    if ($null -eq $kinesistapPackage) {
        throw "No package found matching input version $version"
    }
} else {
    $packageJson = $packageJson.packages | Where-Object { $_.packageName -match
 ".nupkg" }
    $kinesistapPackage = $packageJson[0]
}

$packageName = $kinesistapPackage.packageName
```

```
$checksum = $kinesistapPackage.checksum

#Create %TEMP%/kinesistap if not exists
$kinesistapTempDir = Join-Path $env:TEMP 'kinesistap'
if (![System.IO.Directory]::Exists($kinesistapTempDir)) {[void]
[System.IO.Directory]::CreateDirectory($kinesistapTempDir)}

#Download KinesisTap.x.x.x.x.nupkg package
$kinesistapNupkgPath = Join-Path $kinesistapTempDir $packageName
$webClient.DownloadFile($kinesistapBaseUrl + $packageName, $kinesistapNupkgPath)
$kinesistapUnzipPath = $kinesistapNupkgPath.Replace('.nupkg', '')

# Calculates hash of downloaded file. Downlevel compatible using .Net hashing on PS < 4
if ($PSVersionTable.PSVersion.Major -ge 4) {
    $calculatedHash = Get-FileHash $kinesistapNupkgPath -Algorithm SHA256
    $hashAsString = $calculatedHash.Hash.ToLower()
} else {
    $sha256 = New-Object System.Security.Cryptography.SHA256CryptoServiceProvider
    $calculatedHash =
 [System.BitConverter]::ToString($sha256.ComputeHash([System.IO.File]::ReadAllBytes($kinesistap
    $hashAsString = $calculatedHash.Replace("-", "").ToLower()
}

if ($checksum -eq $hashAsString) {
 Write-Host 'Local file hash matches checksum.' -ForegroundColor Green
} else {
 throw ("Get-FileHash does not match! Package may be corrupted.")
}

#Delete Unzip path if not empty
if ([System.IO.Directory]::Exists($kinesistapUnzipPath)) {Remove-Item –Path
 $kinesistapUnzipPath -Recurse -Force}

#Unzip KinesisTap.x.x.x.x.nupkg package
$null =
 [System.Reflection.Assembly]::LoadWithPartialName('System.IO.Compression.FileSystem')
[System.IO.Compression.ZipFile]::ExtractToDirectory($kinesistapNupkgPath,
 $kinesistapUnzipPath)

#Execute chocolaeyInstall.ps1 in the package and wait for completion.
$installScript = Join-Path $kinesistapUnzipPath '\tools\chocolateyInstall.ps1'
& $installScript

# Verify service installed.
```

```
$serviceName = 'AWSKinesisTap'
$service = Get-Service -Name $serviceName -ErrorAction Ignore
if ($null -eq $service) {
    throw ("Service not installed correctly.")
} else {
    Write-Host "Kinesis Tap Installed." -ForegroundColor Green
    Write-Host "After configuring run the following to start the service: Start-Service
 -Name $serviceName." -ForegroundColor Green
}
```

관리자 권한 명령 프롬프트 창을 엽니다. 파일이 다운로드된 디렉터리에서 다음 명령을 사용하여 스크립트를 실행합니다.

```
PowerShell.exe -File ".\InstallKinesisAgent.ps1"
```

특정 버전의 Windows용 Kinesis 에이전트를 설치하려면`-version`옵션을 선택합니다.

```
PowerShell.exe -File ".\InstallKinesisAgent.ps1" -version "version"
```

Replace*version*를 Windows용 유효한 Kinesis 에이전트 버전 번호로 교체하십시오. 버전 정보는 단원을 참조하십시오.GitHub 의 키네시스-에이전트 - 윈도우 저장소.

원격으로 PowerShell 스크립트를 실행할 수있는 많은 배포 도구가 있습니다. 여러 대의 서버 또는 데스크톱에서 Kinesis Windows용 에이전트를 자동으로 설치하는 데 사용할 수 있습니다.

Windows용 Kinesis 에이전트 구성 및 시작

Windows용 Kinesis 에이전트를 설치한 후에는 에이전트를 구성하고 시작해야 합니다. 그 후에는 더 이상의 작업 개입이 필요하지 않습니다.

Windows용 Kinesis 에이전트를 구성하고 시작하려면

1. Windows용 Kinesis 에이전트 구성 파일을 생성하고 배포합니다. 이 파일은 다른 전역 구성 항목과 함께 소스, 싱크 및 파이프를 구성합니다.

   Windows용 Kinesis 에이전트 구성에 대한 자세한 내용은 단원을 참조하십시오.마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성.

   사용자 지정하고 설치할 수 있는 전체 구성 파일 예제는Windows용 Kinesis 에이전트 구성 예.

2. 상승된 PowerShell 명령 프롬프트 창을 열고 다음 PowerShell 명령을 사용하여 Windows용 Kinesis 에이전트를 시작합니다.

```
Start-Service -Name AWSKinesisTap
```

마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성

Microsoft Windows용 Amazon Kinesis 에이전트를 시작하기 전에 구성 파일을 생성하여 배포해야 합니다. 구성 파일은 Windows 서버 및 데스크톱 컴퓨터의 데이터를 수집, 변환 및 다양한 AWS 서비스로 스트리밍하는 데 필요한 정보를 제공합니다. 구성 파일은 선택적 변환과 함께 소스를 싱크에 연결하는 소스, 싱크 및 파이프 세트를 정의합니다.

Windows용 Kinesis 에이전트 구성 파일의 이름은`appsettings.json`. 이 파일을`%PROGRAMFILES%\Amazon\AWSKinesisTap`.

주제

- 기본 구성 구조
- 소스 선언
- sink 선언
- 파이프 선언
- 자동 업데이트 구성
- Windows용 Kinesis 에이전트 구성 예
- 원격 분석 구성

기본 구성 구조

마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성 파일의 기본 구조는 다음 템플릿이 포함된 JSON 문서입니다.

```
{
     "Sources": [ ],
     "Sinks": [ ],
     "Pipes": [ ]
}
```

- 의 값`Sources`하나 이상의 입니다.소스 선언.
- 의 값`Sinks`하나 이상의 입니다.sink 선언.
- 의 값`Pipes`하나 이상의 입니다.파이프 선언.

Windows용 Kinesis 에이전트 소스, 파이프 및 싱크 개념에 대한 자세한 내용은마이크로소프트 윈도우용 Amazon Kinesis 에이전트 개념.

다음 예제는 완전한`appsettings.json`구성 파일을 사용하여 Kinesis 데이터 Firehose에 Windows 응용 프로그램 로그 이벤트를 스트리밍하도록 구성할 수 있습니다.

```
{
  "Sources": [
    {
      "LogName": "Application",
      "Id": "ApplicationLog",
      "SourceType": "WindowsEventLogSource"
    }
  ],
  "Sinks": [
    {
      "StreamName": "ApplicationLogFirehoseStream",
      "Region": "us-west-2",
      "Id": "MyKinesisFirehoseSink",
      "SinkType": "KinesisFirehose"
    }
  ],
  "Pipes": [
    {
      "Id": "ApplicationLogTotestKinesisFirehoseSink",
      "SourceRef": "ApplicationLog",
      "SinkRef": "MyKinesisFirehoseSink"
    }
  ]
}
```

각 선언에 대한 자세한 내용은 다음 단원을 참조하십시오.

- 소스 선언
- sink 선언
- 파이프 선언

구성 대소문자 구분

JSON 형식의 파일은 일반적으로 대소문자를 구분하므로 Windows용 Kinesis 에이전트 구성 파일의 모든 키와 값도 대/소문자를 구분한다고 가정해야 합니다. 일부 키와 값은`appsettings.json`구성 파일은 대소문자를 구분하지 않습니다. 예를 들면 다음과 같습니다.

- 의 값`Format`싱크에 대한 키-값 페어를 입력합니다. 자세한 내용은 sink 선언 섹션을 참조하세요.
- 의 값`SourceType`키-값 페어를 입력합니다.`SinkType`키-값 페어를 입력합니다.`Type`키-값 페어를 입력합니다.
- 의 값`RecordParser`에 대한 키-값 페어의`DirectorySource`소스에 대해 설명합니다. 자세한 내용은 디렉토리소스 구성 섹션을 참조하세요.
- 의 값`InitialPosition`키-값 페어를 입력합니다. 자세한 내용은 책갈피 구성 섹션을 참조하세요.
- 변수 대체를 위한 접두사입니다. 자세한 내용은 싱크 변수 대체 구성 섹션을 참조하세요.

소스 선언

마이크로소프트 윈도우용 Amazon Kinesis 에이전트에서소스 선언에는 로그, 이벤트 및 메트릭 데이터가 수집되어야 하는 위치와 정보가 나와 있습니다. 또한 선택적으로 변환 할 수 있도록 해당 데이터를 구문 분석하기위한 정보를 지정합니다. 다음 섹션에서는 Windows용 Kinesis 에이전트에서 사용할 수 있는 기본 제공 소스 유형의 구성에 대해 설명합니다. Windows용 Kinesis 에이전트는 확장할 수 있으므로 사용자 정의 소스 유형을 추가할 수 있습니다. 각 소스 유형에는 일반적으로 해당 소스 유형과 관련된 구성 객체에 특정 키 - 값 쌍이 필요합니다.

모든 소스 선언에는 최소한 다음 키-값 페어가 포함되어야 합니다.

`Id`

구성 파일에서 특정 소스 객체를 식별하는 고유한 문자열입니다.

`SourceType`

이 소스 객체의 소스 유형 이름입니다. 소스 유형은 이 소스 개체에 의해 수집 중인 로그, 이벤트 또는 메트릭 데이터의 출처를 지정합니다. 또한 소스의 다른 측면을 선언 할 수있는 제어합니다.

다양한 종류의 소스 선언을 사용하는 전체 구성 파일의 예는다양한 소스에서 Kinesis Data Streams 으로 스트리밍.

주제

- 디렉토리소스 구성
- Exchange 로그소스 구성
- W3SV클로그소스 구성
- UlsSource 구성
- 창 세븐로그소스 구성
- 창 세븐로그폴링소스 구성
- 창 집합소스 구성
- 창기능카운터 소스 구성
- Windows용 Kinesis 에이전트 기본 제공 메트릭 소스
- Windows용 Kinesis 에이전트 메트릭 목록
- 책갈피 구성

디렉토리소스 구성

Overview

이`DirectorySource`소스 유형은 지정된 디렉토리에 저장된 파일에서 로그를 수집합니다. 로그 파일은 여러 가지 형식으로 제공되므로`DirectorySource`선언을 사용하면 로그 파일의 데이터 형식을 지정할 수 있습니다. 그런 다음 다양한 AWS 서비스로 스트리밍하기 전에 로그 내용을 JSON 또는 XML과 같은 표준 형식으로 변환할 수 있습니다.

다음은 예제입니다.`DirectorySource`선언:

```
{
    "Id": "myLog",
    "SourceType": "DirectorySource",
    "Directory": "C:\\Program Data\\MyCompany\\MyService\\logs",
    "FileNameFilter": "*.log",
    "IncludeSubdirectories": true,
    "IncludeDirectoryFilter": "cpu\\cpu-1;cpu\\cpu-2;load;memory",
    "RecordParser": "Timestamp",
    "TimestampFormat": "yyyy-MM-dd HH:mm:ss.ffff",
    "Pattern": "\\d{4}-\\d{2}-\\d(2}",
    "ExtractionPattern": "",
    "TimeZoneKind": "UTC",
    "SkipLines": 0,
    "Encoding": "utf-16",
```

```
    "ExtractionRegexOptions": "Multiline"
}
```

모두DirectorySource선언은 다음 키-값 페어를 제공할 수 있습니다.

SourceType

리터럴 문자열이어야 합니다."DirectorySource"(필수).

Directory

로그 파일이 들어 있는 디렉토리의 경로입니다 (필수 사항).

FileNameFilter

선택적으로 와일드카드 파일 이름 지정 패턴에 따라 로그 데이터가 수집되는 디렉터리의 파일 집합을 제한합니다. 로그 파일 이름 패턴이 여러 개 있는 경우 이 기능을 사용하면 단일DirectorySource다음 예제에 표시된 대로 을 클릭합니다.

```
FileNameFilter: "*.log|*.txt"
```

시스템 관리자는 로그 파일을 보관하기 전에 압축하는 경우가 있습니다. 를 지정하는 경우"*.*"inFileNameFilter로 설정하면 알려진 압축 파일이 이제 제외됩니다. 이 기능은.zip,.gz, 및.bz2파일이 실수로 스트리밍되는 것을 방지할 수 있습니다. 이 키-값 쌍을 지정하지 않으면 디렉토리에 있는 모든 파일의 데이터가 기본적으로 수집됩니다.

IncludeSubdirectories

운영 체제에 의해 제한되는 임의의 깊이로 하위 디렉토리를 모니터링하도록 지정합니다. 이 기능은 여러 웹 사이트가 있는 웹 서버를 모니터링하는 데 유용합니다. 뿐만 아니라IncludeDirectoryFilter특성을 사용하여 필터에 지정된 특정 하위 디렉토리만 모니터링합니다.

RecordParser

방법을 지정합니다DirectorySource소스 유형은 지정된 디렉토리에있는 로그 파일을 구문 분석해야합니다. 이 키 - 값 쌍이 필요하며 유효한 값은 다음과 같습니다.

- SingleLine— 로그 파일의 각 행은 로그 레코드입니다.
- SingleLineJson— 로그 파일의 각 행은 JSON 형식의 로그 레코드입니다. 이 파서는 객체 장식을 사용하여 JSON에 추가 키 - 값 쌍을 추가 할 때 유용합니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요. 예를 들어,SingleLineJson레코드 파서에 대한 자세한 내용은자습서: 윈도우용 Kinesis 에이전트를 사용하여 JSON 로그 파일을 Amazon S3 로 스트리밍.

- `Timestamp`- 하나 이상의 줄에 로그 레코드가 포함될 수 있습니다. 로그 레코드는 타임스탬프로 시작합니다. 이 옵션을 사용하려면`TimestampFormat`키-값 페어를 선택합니다.
- `Regex`— 각 레코드는 특정 정규 표현식과 일치하는 텍스트로 시작합니다. 이 옵션을 사용하려면`Pattern`키-값 페어를 선택합니다.
- `SysLog`— 로그 파일이syslog표준 형식입니다. 로그 파일은 해당 사양에 따라 레코드로 구문 분석됩니다.
- `Delimited`— 로그 레코드의 데이터 항목이 일관된 구분 기호로 구분되는 Regex 레코드 파서의 간단한 버전입니다. 이 옵션은 사용하기 쉽고 정규식 파서보다 빠르게 실행되며이 옵션을 사용할 수있는 경우 선호됩니다. 이 옵션을 사용하는 경우`Delimiter`키-값 페어를 선택합니다.

`TimestampField`

레코드에 대한 타임스탬프를 포함하는 JSON 필드를 지정합니다. 이는 단지`SingleLineJson RecordParser`. 이 키-값 페어는 선택 사항입니다. 지정되지 않은 경우 Windows용 Kinesis 에이전트는 타임스탬프에 대해 레코드를 읽은 시간을 사용합니다. 이 키-값 쌍을 지정할 때의 한 가지 이점은 Windows용 Kinesis 에이전트에서 생성된 지연 시간 통계가 보다 정확하다는 것입니다.

`TimestampFormat`

레코드와 연관된 날짜 및 시간을 구문 분석하는 방법을 지정합니다. 문자열 중 하나입니다.`epoch`또는.NET 날짜/시간 형식 문자열을 사용하십시오. 값이`epoch`, 시간은 UNIX 에포크 시간을 기반으로 구문 분석됩니다. UNIX Epoch 시간에 대한 자세한 내용은 단원을 참조하십시오.Unix 시간. .NET 날짜/시간 형식 문자열에 대한 자세한 내용은사용자 정의 날짜 및 시간 형식 문자열를 참조하십시오). 이 키 - 값 쌍은 경우에만 필요합니다`Timestamp`레코드 파서가 지정되거나`SingleLineJson`레코드 파서와 함께 지정`TimestampField`키-값 페어를 선택합니다.

`Pattern`

잠재적으로 여러 줄 레코드의 첫 번째 줄과 일치해야 하는 정규 표현식을 지정합니다. 이 키 - 값 쌍은`Regex`레코드 구문 분석기.

`ExtractionPattern`

명명된 그룹을 사용해야 하는 정규 표현식을 지정합니다. 레코드는이 정규 표현식을 사용하여 구문 분석하고 명명 된 그룹은 구문 분석 된 레코드의 필드를 형성한다. 그런 다음 이러한 필드는 JSON 또는 XML 객체 또는 문서를 구성하기 위한 기반으로 사용되며, 이 객체는 싱크로 다양한 AWS 서비스로 스트리밍됩니다. 이 키 - 값 쌍은 선택 사항이며`Regex`레코드 파서 및 타임 스탬프 파서를 사용할 수 있습니다.

이`Timestamp`그룹 이름은 특별히 처리되며`Regex`각 로그 파일의 각 레코드에 대한 날짜 및 시간을 포함하는 파서.

`Delimiter`

각 로그 기록의 각 항목을 구분하는 문자 또는 문자열을 지정합니다. 이 키 - 값 쌍은`Delimited`레코드 구문 분석기. 두 문자 시퀀스 사용`\t`를 사용하여 탭 문자를 나타냅니다.

`HeaderPattern`

레코드에 대한 헤더 집합을 포함하는 로그 파일의 행을 일치시키기 위한 정규 표현식을 지정합니다. 로그 파일에 헤더 정보가 포함되어 있지 않으면`Headers`키-값 페어를 사용하여 암시 적 헤더를 지정합니다. 이`HeaderPattern`키 - 값 쌍은 선택 사항이며`Delimited`레코드 구문 분석기.

> **Note**
>
> 열에 대한 빈 (0 길이) 머리글 항목을 해당 열의 데이터를 최종 출력에서 필터링할 수 있는`DirectorySource`출력을 구문 분석합니다.

`Headers`

지정된 구분 기호를 사용하여 구문 분석된 데이터 열의 이름을 지정합니다. 이 키 - 값 쌍은 선택 사항이며`Delimited`레코드 구문 분석기.

> **Note**
>
> 열에 대한 빈 (0 길이) 머리글 항목을 해당 열의 데이터를 최종 출력에서 필터링할 수 있는`DirectorySource`출력을 구문 분석합니다.

`RecordPattern`

로그 파일에서 레코드 데이터가 포함된 행을 식별하는 정규 표현식을 지정합니다. 에 의해 식별되는 선택적 헤더 라인 이외의`HeaderPattern`, 지정된 값과 일치하지 않는 행`RecordPattern`는 레코드 처리 중에 무시됩니다. 이 키 - 값 쌍은 선택 사항이며`Delimited`레코드 구문 분석기. 이 옵션이 제공되지 않으면 기본값은 선택적`HeaderPattern`또는 선택 사항`CommentPattern`를 구문 분석 가능한 레코드 데이터가 포함 된 행으로 설정하십시오.

`CommentPattern`

로그 파일의 데이터를 구문 분석하기 전에 제외해야 하는 로그 파일의 행을 식별하는 정규식을 지정합니다. 이 키 - 값 쌍은 선택 사항이며`Delimited`레코드 구문 분석기. 이 옵션이 제공되지 않으

면 기본값은 선택적`HeaderPattern`를 구문 분석 가능한 레코드 데이터가 포함 된 행으로 설정하십시오.`RecordPattern`가 지정됩니다.

`TimeZoneKind`

로그 파일의 타임스탬프를 현지 표준 시간대 또는 UTC 표준 시간대에서 고려할지 여부를 지정합니다. 선택 사항으로, 기본값은 UTC입니다. 이 키 - 값 쌍에 대한 유일한 유효한 값은`Local`또는`UTC`. 다음과 같은 경우 타임 스탬프가 변경되지 않습니다.`TimeZoneKind`가 지정되지 않았거나 값이 UTC인 경우 타임 스탬프는 UTC로 변환 될 때`TimeZoneKind`값은`Local`이고 타임 스탬프를받는 싱크가 CloudWatch Logs 이거나 구문 분석 된 레코드가 다른 싱크로 전송됩니다. 메시지에 포함된 날짜와 시간은 변환되지 않습니다.

`SkipLines`

지정된 경우 레코드 구문 분석이 수행되기 전에 각 로그 파일의 시작 부분에서 무시되는 줄 수를 제어합니다. 선택 사항으로, 기본값은 0입니다.

인코딩

기본적으로 Windows용 Kinesis 에이전트는 바이트마크에서 인코딩을 자동으로 감지할 수 있습니다. 그러나 일부 이전 유니코드 형식에서는 자동 인코딩이 제대로 작동하지 않을 수 있습니다. 다음 예제에서는 Microsoft SQL Server 로그를 스트리밍하는 데 필요한 인코딩을 지정합니다.

```
"Encoding": "utf-16"
```

인코딩 이름 목록은 단원을 참조하십시오.인코딩 목록Microsoft .NET 설명서에 나와 있습니다.

추출정규 표현식 옵션

다음을 수행할 수 있습니다.`ExtractionRegexOptions`정규 표현식을 단순화할 수 있습니다. 이 키-값 페어는 선택 사항입니다. 기본값은 `"None"`입니다.

다음 예제에서는 지정 된`"."`식을 포함한 모든 문자와 일치`\r\n`.

```
"ExtractionRegexOptions" = "Multiline"
```

추출정규 표현식 옵션에 사용할 수 있는 필드 목록은열거형 정규식 옵션Microsoft .NET 설명서에 나와 있습니다.

Regex레코드 구문 분석기

구조화되지 않은 텍스트 로그를 구문 분석할 수 있는Regex레코드 파서와 함께TimestampFormat,Pattern, 및ExtractionPattern키-값 페어를 입력합니다. 예를 들면 로그 파일이 다음과 같다고 가정합니다.

```
[FATAL][2017/05/03 21:31:00.534][0x00003ca8][0000059c][][ActivationSubSystem]
[GetActivationForSystemID][0] 'ActivationException.File: EQCASLicensingSubSystem.cpp'
[FATAL][2017/05/03 21:31:00.535][0x00003ca8][0000059c][][ActivationSubSystem]
[GetActivationForSystemID][0] 'ActivationException.Line: 3999'
```

당신은에 대한 다음 정규 표현식을 지정할 수 있습니다Pattern키 - 값 쌍을 사용하여 로그 파일을 개별 로그 레코드로 나눌 수 있습니다.

```
^\[\w+\]\[(?<TimeStamp>\d{4}/\d{2}/\d{2} \d{2}:\d{2}:\d{2}\.\d{3})\]
```

이 정규 표현식은 다음 순서와 일치합니다.

1. 평가되는 문자열의 시작입니다.
2. 대괄호로 둘러싸인 하나 이상의 단어 문자
3. 대괄호로 둘러싸인 타임스탬프입니다. 타임스탬프는 다음 시퀀스와 일치합니다.
   a. 4자리 연도입니다.
   b. 슬래시
   c. 2자리 월입니다.
   d. 슬래시
   e. 2자리 날짜입니다.
   f. 공백 문자
   g. 2자리 시입니다.
   h. 콜론
   i. 2자리 분입니다.
   j. 콜론
   k. 2자리
   l. A 기간

m. 세 자리 밀리초

다음 형식을 지정할 수 있습니다.`TimestampFormat`키 - 값 쌍을 사용하여 텍스트 타임 스탬프를 날짜와 시간으로 변환 할 수 있습니다.

```
yyyy/MM/dd HH:mm:ss.fff
```

당신은을 통해 로그 레코드의 필드를 추출하기 위해 다음과 같은 정규 표현식을 사용할 수 있습니다`ExtractionPattern`키-값 페어를 입력합니다.

```
^\[(?<Severity>\w+)\]\[(?<TimeStamp>\d{4}/\d{2}/\d{2} \d{2}:\d{2}:\d{2}\.
\d{3})\]\[[^]]*\]\[[^]]*\]\[[^]]*\]\[(?<SubSystem>\w+)\]\[(?<Module>\w+)\]\[[^]]*\] '(?
<Message>.*)'$
```

이 정규 표현식은 다음 그룹과 순서대로 일치합니다.

1. `Severity`— 대괄호로 둘러싸인 하나 이상의 단어 문자입니다.
2. `TimeStamp`— 타임스탬프에 대한 이전 설명을 참조하십시오.
3. 0개 이상의 문자 시퀀스를 명명되지 않은 대괄호로 묶인 3개를 건너뜁니다.
4. `SubSystem`— 대괄호로 둘러싸인 하나 이상의 단어 문자입니다.
5. `Module`— 대괄호로 둘러싸인 하나 이상의 단어 문자입니다.
6. 0개 이상의 문자 시퀀스를 명명되지 않은 대괄호로 묶인 시퀀스를 건너뜁니다.
7. 이름이 지정되지 않은 공간 하나를 건너뜁니다.
8. `Message`— 작은따옴표로 둘러싸인 0개 이상의 문자입니다.

다음 소스 선언은 이러한 정규 표현식과 날짜 시간 형식을 결합하여 이러한 종류의 로그 파일을 구문 분석하는 데 필요한 전체 지침을 Kinesis Agent에 제공합니다.

```
{
    "Id": "PrintLog",
    "SourceType": "DirectorySource",
    "Directory": "C:\\temp\\PrintLogTest",
    "FileNameFilter": "*.log",
    "RecordParser": "Regex",
    "TimestampFormat": "yyyy/MM/dd HH:mm:ss.fff",
    "Pattern": "^\\[\\w+\\]\\[(?<TimeStamp>\\d{4}/\\d{2}/\\d{2} \\d{2}:\\d{2}:\\d{2}\\.
\\d{3})\\]",
```

```
    "ExtractionPattern": "^\\[(?<Severity>\\w+)\\]\\[(?<TimeStamp>\\d{4}/\\d{2}/\\d{2}
 \\d{2}:\\d{2}:\\d{2}\\.\\d{3})\\]\\[[^]]*\\]\\[[^]]*\\]\\[[^]]*\\]\\[(?<SubSystem>\\w
+)\\]\\[(?<Module>\\w+)\\]\\[[^]]*\\] '(?<Message>.*)'$",
    "TimeZoneKind": "UTC"
}
```

> **Note**
>
> JSON 형식 파일의 백 슬래시는 추가 백 슬래시를 사용하여 이스케이프해야합니다.

정규식에 대한 자세한 내용은 단원을 참조하십시오.정규 표현식 언어 - 빠른 참조에서 Microsoft .NET 설명서를 참조하십시오.

Delimited레코드 구문 분석기

다음을 수행할 수 있습니다.Delimited레코드 파서를 사용하여 각 데이터 행의 각 데이터 열을 분리하는 일관된 문자 시퀀스가있는 반구조화 된 로그 및 데이터 파일을 구문 분석합니다. 예를 들어 CSV 파일은 쉼표를 사용하여 각 데이터 열을 구분하고 TSV 파일은 탭을 사용합니다.

Microsoft를 구문 분석하려는 경우NPS 데이터베이스 형식네트워크 정책 서버에 의해 생성 된 로그 파일. 이러한 파일은 다음과 같습니다.

```
"NPS-
MASTER","IAS",03/22/2018,23:07:55,1,"user1","Domain1\user1",,,,,,,,0,"192.168.86.137","Nate
 - Test 1",,,,,,,1,,0,"311 1 192.168.0.213 03/15/2018 08:14:29
 1",,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,"Use Windows authentication for all users",1,,,,
"NPS-
MASTER","IAS",03/22/2018,23:07:55,3,,"Domain1\user1",,,,,,,,0,"192.168.86.137","Nate
 - Test 1",,,,,,,1,,16,"311 1 192.168.0.213 03/15/2018 08:14:29
 1",,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,"Use Windows authentication for all users",1,,,,
```

다음은 예제입니다.appsettings.json구성 파일에는DirectorySource선언을 사용하는Delimited레코드 파서를 사용하여이 텍스트를 객체 표현으로 구문 분석 할 수 있습니다. 그런 다음 JSON 형식의 데이터를 Kinesis Data Firehose 스로 스트리밍합니다.

```
{
    "Sources": [
        {
            "Id": "NPS",
            "SourceType": "DirectorySource",
```

```
            "Directory": "C:\\temp\\NPS",
            "FileNameFilter": "*.log",
            "RecordParser": "Delimited",
            "Delimiter": ",",
            "Headers": "ComputerName,ServiceName,Record-Date,Record-Time,Packet-
Type,User-Name,Fully-Qualified-Distinguished-Name,Called-Station-ID,Calling-Station-
ID,Callback-Number,Framed-IP-Address,NAS-Identifier,NAS-IP-Address,NAS-Port,Client-
Vendor,Client-IP-Address,Client-Friendly-Name,Event-Timestamp,Port-Limit,NAS-Port-
Type,Connect-Info,Framed-Protocol,Service-Type,Authentication-Type,Policy-Name,Reason-
Code,Class,Session-Timeout,Idle-Timeout,Termination-Action,EAP-Friendly-Name,Acct-
Status-Type,Acct-Delay-Time,Acct-Input-Octets,Acct-Output-Octets,Acct-Session-Id,Acct-
Authentic,Acct-Session-Time,Acct-Input-Packets,Acct-Output-Packets,Acct-Terminate-
Cause,Acct-Multi-Ssn-ID,Acct-Link-Count,Acct-Interim-Interval,Tunnel-Type,Tunnel-
Medium-Type,Tunnel-Client-Endpt,Tunnel-Server-Endpt,Acct-Tunnel-Conn,Tunnel-Pvt-
Group-ID,Tunnel-Assignment-ID,Tunnel-Preference,MS-Acct-Auth-Type,MS-Acct-EAP-Type,MS-
RAS-Version,MS-RAS-Vendor,MS-CHAP-Error,MS-CHAP-Domain,MS-MPPE-Encryption-Types,MS-
MPPE-Encryption-Policy,Proxy-Policy-Name,Provider-Type,Provider-Name,Remote-Server-
Address,MS-RAS-Client-Name,MS-RAS-Client-Version",
            "TimestampField": "{Record-Date} {Record-Time}",
            "TimestampFormat": "MM/dd/yyyy HH:mm:ss"
        }
    ],
    "Sinks": [
        {
            "Id": "npslogtest",
            "SinkType": "KinesisFirehose",
            "Region": "us-west-2",
            "StreamName": "npslogtest",
            "Format": "json"
        }
    ],
    "Pipes": [
        {
            "Id": "W3SVCLog1ToKinesisStream",
            "SourceRef": "NPS",
            "SinkRef": "npslogtest"
        }
    ]
}
```

Kinesis 데이터 파이어호스로 스트리밍되는 JSON 형식의 데이터는 다음과 같습니다.

```
{
```

```
    "ComputerName": "NPS-MASTER",
    "ServiceName": "IAS",
    "Record-Date": "03/22/2018",
    "Record-Time": "23:07:55",
    "Packet-Type": "1",
    "User-Name": "user1",
    "Fully-Qualified-Distinguished-Name": "Domain1\\user1",
    "Called-Station-ID": "",
    "Calling-Station-ID": "",
    "Callback-Number": "",
    "Framed-IP-Address": "",
    "NAS-Identifier": "",
    "NAS-IP-Address": "",
    "NAS-Port": "",
    "Client-Vendor": "0",
    "Client-IP-Address": "192.168.86.137",
    "Client-Friendly-Name": "Nate - Test 1",
    "Event-Timestamp": "",
    "Port-Limit": "",
    "NAS-Port-Type": "",
    "Connect-Info": "",
    "Framed-Protocol": "",
    "Service-Type": "",
    "Authentication-Type": "1",
    "Policy-Name": "",
    "Reason-Code": "0",
    "Class": "311 1 192.168.0.213 03/15/2018 08:14:29 1",
    "Session-Timeout": "",
    "Idle-Timeout": "",
    "Termination-Action": "",
    "EAP-Friendly-Name": "",
    "Acct-Status-Type": "",
    "Acct-Delay-Time": "",
    "Acct-Input-Octets": "",
    "Acct-Output-Octets": "",
    "Acct-Session-Id": "",
    "Acct-Authentic": "",
    "Acct-Session-Time": "",
    "Acct-Input-Packets": "",
    "Acct-Output-Packets": "",
    "Acct-Terminate-Cause": "",
    "Acct-Multi-Ssn-ID": "",
    "Acct-Link-Count": "",
    "Acct-Interim-Interval": "",
```

```
    "Tunnel-Type": "",
    "Tunnel-Medium-Type": "",
    "Tunnel-Client-Endpt": "",
    "Tunnel-Server-Endpt": "",
    "Acct-Tunnel-Conn": "",
    "Tunnel-Pvt-Group-ID": "",
    "Tunnel-Assignment-ID": "",
    "Tunnel-Preference": "",
    "MS-Acct-Auth-Type": "",
    "MS-Acct-EAP-Type": "",
    "MS-RAS-Version": "",
    "MS-RAS-Vendor": "",
    "MS-CHAP-Error": "",
    "MS-CHAP-Domain": "",
    "MS-MPPE-Encryption-Types": "",
    "MS-MPPE-Encryption-Policy": "",
    "Proxy-Policy-Name": "Use Windows authentication for all users",
    "Provider-Type": "1",
    "Provider-Name": "",
    "Remote-Server-Address": "",
    "MS-RAS-Client-Name": "",
    "MS-RAS-Client-Version": ""
}
```

SysLog레코드 구문 분석기

에 대 한SysLog레코드 파서를 사용하는 경우 원본의 구문 분석 된 출력에는 다음 정보가 포함됩니다.

속성	유형	설명
SysLogTimeStamp	문자열	syslog가 포맷한 로그 파일의 원래 날짜 및 시간입니다.
Hostname	문자열	syslog로 포맷된 로그 파일이 있는 컴퓨터의 이름입니다.
Program	문자열	로그 파일을 생성한 애플리케이션 또는 서비스의 이름입니다.
Message	문자열	응용 프로그램 또는 서비스에서 생성된 로그 메시지입니다.

속성	유형	설명
TimeStamp	문자열	ISO 8601 형식의 구문 분석 날짜 및 시간입니다.

다음은 JSON으로 변환된 SysLog 데이터의 예입니다.

```
{
    "SysLogTimeStamp": "Jun 18 01:34:56",
    "Hostname": "myhost1.example.mydomain.com",
    "Program": "mymailservice:",
    "Message": "Info: ICID 123456789 close",
    "TimeStamp": "2017-06-18T01:34.56.000"
}
```

Summary

다음에 사용할 수 있는 키-값 페어의 요약은DirectorySource소스 및RecordParser이 키-값 페어와 관련되어 있습니다.

키 이름	레코드파서	참고
SourceType	모든 사용자에게 필수	값이 있어야 합니다.Directory Source
Directory	모든 사용자에게 필수	
FileNameFilter	모든 항목에 대한 선택 사항	
RecordParser	모든 사용자에게 필수	
TimestampField	의 경우 선택 사항SingleLineJson	
TimestampFormat	의 필수 사항Timestamp 에 대한 필수SingleLineJson 다음과 같은 경	

키 이름	레코드파서	참고
	우TimestampField 가 지정됩니다.	
Pattern	의 필수 사항Regex	
ExtractionPattern	의 경우 선택 사항Regex	의 필수 사항Regex싱크가 지정하는 경우json또는xmlformat
Delimiter	의 필수 사항Delimited	
HeaderPattern	의 경우 선택 사항Delimited	
Headers	의 경우 선택 사항Delimited	
RecordPattern	의 경우 선택 사항Delimited	
CommentPattern	의 경우 선택 사항Delimited	
TimeZoneKind	의 경우 선택 사항Regex,Timestamp ,SysLog, 및SingleLineJson 타임 스탬프 필드가 식별 될 때	
SkipLines	모든 항목에 대한 선택 사항	

Exchange 로그소스 구성

이ExchangeLogSource유형은 마이크로소프트 익스체인지에서 로그를 수집하는 데 사용됩니다. Exchange는 여러 종류의 로그 형식으로 로그를 생성합니다. 이 소스 유형은 모두 구문 분석합니다. 그것을 사용하여 구문 분석 할 수는 있지만DirectorySource유형을Regex레코드 파서를 사용하는 것

이 훨씬 간단합니다`ExchangeLogSource`. 로그 파일의 정규식을 디자인하고 제공할 필요가 없기 때문입니다. 다음은 예제입니다.`ExchangeLogSource`선언:

```
{
   "Id": "MyExchangeLog",
   "SourceType": "ExchangeLogSource",
   "Directory": "C:\\temp\\ExchangeLogTest",
   "FileNameFilter": "*.log"
}
```

모든 교환 선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"ExchangeLogSource"`(필수).

`Directory`

로그 파일이 들어 있는 디렉토리의 경로입니다 (필수 사항).

`FileNameFilter`

선택적으로 와일드카드 파일 이름 지정 패턴에 따라 로그 데이터가 수집되는 디렉터리의 파일 집합을 제한합니다. 이 키-값 쌍을 지정하지 않으면 기본적으로 디렉토리의 모든 파일에서 로그 데이터가 수집됩니다.

`TimestampField`

레코드의 날짜 및 시간을 포함하는 열의 이름입니다. 이 키 - 값 쌍은 선택 사항이며 필드 이름이`date-time`또는`DateTime`. 그렇지 않으면 필요합니다.

W3SV클로그소스 구성

이`W3SVCLogSource`유형은 Windows용 IIS (인터넷 정보 서비스) 에서 로그를 수집하는 데 사용됩니다.

다음은 예제입니다.`W3SVCLogSource`선언:

```
{
   "Id": "MyW3SVCLog",
   "SourceType": "W3SVCLogSource",
```

```
    "Directory": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
    "FileNameFilter": "*.log"
}
```

모두`W3SVCLogSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"W3SVCLogSource"`(필수).

`Directory`

로그 파일이 들어 있는 디렉토리의 경로입니다 (필수 사항).

`FileNameFilter`

선택적으로 와일드카드 파일 이름 지정 패턴에 따라 로그 데이터가 수집되는 디렉터리의 파일 집합을 제한합니다. 이 키-값 쌍을 지정하지 않으면 기본적으로 디렉토리의 모든 파일에서 로그 데이터가 수집됩니다.

UlsSource 구성

이`UlsSource`유형은 마이크로 소프트 SharePoint 포인트에서 로그를 수집하는 데 사용됩니다. 다음은 예제입니다.`UlsSource`선언:

```
{
    "Id": "UlsSource",
    "SourceType": "UlsSource",
    "Directory": "C:\\temp\\uls",
    "FileNameFilter": "*.log"
}
```

모두`UlsSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"UlsSource"`(필수).

`Directory`

로그 파일이 들어 있는 디렉토리의 경로입니다 (필수 사항).

`FileNameFilter`

선택적으로 와일드카드 파일 이름 지정 패턴에 따라 로그 데이터가 수집되는 디렉터리의 파일 집합을 제한합니다. 이 키-값 쌍을 지정하지 않으면 기본적으로 디렉토리의 모든 파일에서 로그 데이터가 수집됩니다.

창 세븐로그소스 구성

이`WindowsEventLogSource`유형은 Windows 이벤트 로그 서비스에서 이벤트를 수집하는 데 사용됩니다. 다음은 예제입니다.`WindowsEventLogSource`선언:

```
{
    "Id": "mySecurityLog",
    "SourceType": "WindowsEventLogSource",
    "LogName": "Security"
}
```

모두`WindowsEventLogSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"WindowsEventLogSource"`(필수).

`LogName`

이벤트는 지정된 로그에서 수집됩니다. 일반적인 값은 다음과 같습니다.`Application`,`Security`, 및`System`를 사용할 수 있지만 유효한 Windows 이벤트 로그 이름을 지정할 수 있습니다. 이 키-값 페어가 필요합니다.

`Query`

선택적으로 어떤 이벤트가 출력되는지를 제한합니다`WindowsEventLogSource`. 이 키 - 값 쌍을 지정하지 않으면 기본적으로 모든 이벤트가 출력됩니다. 이 값의 구문에 대한 자세한 내용은 다음 () 을 참조하십시오.이벤트 쿼리 및 이벤트 XML에서 Windows 설명서를 참조하십시오. 로그 수준 정의에 대한 자세한 내용은 단원을 참조하십시오.이벤트 유형에서 Windows 설명서를 참조하십시오.

`IncludeEventData`

선택적으로 이 키-값 쌍의 값이`"true"`. 성공적으로 직렬화할 수 있는 이벤트 데이터만 포함됩니다. 이 키-값 쌍은 선택 사항이며 지정하지 않으면 공급자 관련 이벤트 데이터가 수집되지 않습니다.

> **Note**
> 이벤트 데이터를 포함하면 이 소스에서 스트리밍되는 데이터의 양이 크게 늘어날 수 있습니다. 이벤트의 최대 크기는 이벤트 데이터가 포함된 262,143바이트가 될 수 있습니다.

의 구문 분석 된 출력`WindowsEventLogSource`에는 다음 정보가 포함됩니다.

속성	유형	설명
`EventId`	정수	이벤트 유형의 식별자입니다.
`Description`	문자열	이벤트의 세부 정보를 설명하는 텍스트입니다.
`LevelDisplayName`	문자열	이벤트 범주 (오류, 경고, 정보, 성공 감사, 실패 감사 중 하나) 입니다.
`LogName`	문자열	이벤트가 기록된 위치 (일반적인 값은`Application` ,`Security`, 및`System`, 그러나 많은 가능성이있다).
`MachineName`	문자열	이벤트를 기록한 컴퓨터입니다.
`ProviderName`	문자열	이벤트를 기록한 응용 프로그램 또는 서비스
`TimeCreated`	문자열	ISO 8601 형식으로 이벤트가 발생한 경우
`Index`	정수	로그에있는 항목입니다.
`UserName`	문자열	알려진 경우 누가 항목을 만들었습니다.
`Keywords`	문자열	이벤트의 유형입니다. 표준 값은 다음과 같습니다.`AuditFailure` (실패한 보안 감사 이벤트),`AuditSucc`

속성	유형	설명
		ess (성공적인 보안 감사 이벤트),Classic와 함께 발생 이벤트 (RaiseEvent 함수)Correlation Hint (전송 이벤트),SQM(서비스 품질 메커니즘 이벤트),WDI Context(Windows 진단 인프라 컨텍스트 이벤트) 및WDI Diag(Windows 진단 인프라 진단 이벤트).
EventData	List object	로그 이벤트에 대한 공급자별 추가 데이터 (선택 사항) 에 대 한 값 경우에만 포함 됩니다IncludeEventData 키-값 페어는"true".

다음은 JSON으로 변환된 이벤트 예제입니다.

```
{[
    "EventId": 7036,
    "Description": "The Amazon SSM Agent service entered the stopped state.",
    "LevelDisplayName": "Informational",
    "LogName": "System",
    "MachineName": "mymachine.mycompany.com",
    "ProviderName": "Service Control Manager",
    "TimeCreated": "2017-10-04T16:42:53.8921205Z",
    "Index": 462335,
    "UserName": null,
    "Keywords": "Classic",
    "EventData": [
    "Amazon SSM Agent",
    "stopped",
    "rPctBAMZFhYubF8zVLcrBd3bTTcNzHvY5Jc2Br0aMrxxx=="
]}
```

창 세븐로그폴링소스 구성

WindowsEventLogPollingSource는 폴링 기반 메커니즘을 사용하여 구성된 매개 변수와 일치하는 이벤트 로그에서 모든 새 이벤트를 수집합니다. 폴링 간격은 마지막 폴링 중에 수집

된 이벤트 수에 따라 100ms에서 5000ms 사이에서 동적으로 업데이트됩니다. 다음은 예제입니다.`WindowsEventLogPollingSource`선언:

```
{
    "Id": "MySecurityLog",
    "SourceType": "WindowsEventLogPollingSource",
    "LogName": "Security",
    "IncludeEventData": "true",
    "Query": "",
    "CustomFilters": "ExcludeOwnSecurityEvents"
}
```

모두`WindowsEventLogPollingSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"WindowsEventLogPollingSource"`(필수).

`LogName`

로그를 지정합니다. 유효한 옵션은 다음과 같습니다.`Application`,`Security`,`System`또는 기타 유효한 로그가 있습니다.

`IncludeEventData`

선택 사항입니다. 일시`true`는 JSON 및 XML로 스트리밍될 때 추가 EventData 포함되도록 지정합니다. 기본값은 `false`.

`Query`

선택 사항입니다. Windows 이벤트 로그는 XPath 식을 사용하여 이벤트를 쿼리하는 기능을 지원합니다.`Query`. 자세한 내용은 단원을 참조하십시오.이벤트 쿼리 및 이벤트 XML에서 Microsoft 설명서를 참조하십시오.

`CustomFilters`

선택 사항입니다. 세미콜론으로 구분된 필터 목록 (`;`). 다음 필터를 지정할 수 있습니다.

`ExcludeOwnSecurityEvents`

Windows용 Kinesis 에이전트에서 생성된 보안 이벤트를 제외합니다.

창 집합소스 구성

이`WindowsETWEventSource`유형은 ETW (Windows용 이벤트 추적) 라는 기능을 사용하여 응용 프로그램 및 서비스 이벤트 추적을 수집하는 데 사용됩니다. 자세한 내용은 단원을 참조하십시오.이벤트 추적에서 Windows 설명서를 참조하십시오.

다음은 예제입니다.`WindowsETWEventSource`선언:

```
{
    "Id": "ClrETWEventSource",
    "SourceType": "WindowsETWEventSource",
    "ProviderName": "Microsoft-Windows-DotNETRuntime",
    "TraceLevel": "Verbose",
    "MatchAnyKeyword": 32768
}
```

모두`WindowsETWEventSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"WindowsETWEventSource"`(필수).

`ProviderName`

추적 이벤트를 수집하는 데 사용할 이벤트 공급자를 지정합니다. 이 이름은 설치된 공급자에 대해 유효한 ETW 제공자 이름이어야 합니다. 설치된 공급자를 확인하려면 Windows 명령 프롬프트 창에서 다음을 실행합니다.

```
logman query providers
```

`TraceLevel`

수집해야 하는 추적 이벤트 범주를 지정합니다. 허용되는 값은 다음과 같습니다.`Critical`,`Error`,`Warning`,`Informational`, 및`Verbose`. 정확한 의미는 선택한 ETW 공급자에 따라 다릅니다.

`MatchAnyKeyword`

이 값은 각 비트가 개별 키워드를 나타내는 64비트 숫자입니다. 각 키워드는 수집할 이벤트의 범주를 설명합니다. 지원되는 키워드 및 해당 값 및 해당 키워드와 관련된 방법`TraceLevel`에 대한 자세한 내용은 해당 공급자의 설명서를 참조하십시오. 예를 들어 CLR ETW 공급자에 대한 자세한 내용은CLR ETW 키워드 및 레벨Microsoft .NET Framework 설명서를 참조하십시오.

앞의 예제에서 32768 (0x00008000) 은`ExceptionKeyword`throw된 예외에 대한 정보를 수집하도록 공급자에게 지시하는 CLR ETW 공급자에 대해 설명합니다. JSON 은 기본적으로 16 진수 상수를 지원하지 않지만`MatchAnyKeyword`문자열에 배치하여. 여러 상수를 쉼표로 구분하여 지정할 수도 있습니다. 예를 들어, 다음을 사용하여`ExceptionKeyword`및`SecurityKeyword`(0x00000400):

```
{
   "Id": "MyClrETWEventSource",
   "SourceType": "WindowsETWEventSource",
   "ProviderName": "Microsoft-Windows-DotNETRuntime",
   "TraceLevel": "Verbose",
   "MatchAnyKeyword": "0x00008000, 0x00000400"
}
```

공급자에 대해 지정된 모든 키워드가 활성화되도록 하려면 OR을 사용하여 여러 키워드 값을 결합하고 해당 공급자에게 전달합니다.

의 출력`WindowsETWEventSource`에는 각 이벤트에 대해 다음 정보가 포함됩니다.

속성	유형	설명
`EventName`	문자열	어떤 이벤트가 발생했는지.
`ProviderName`	문자열	이벤트를 감지한 공급자입니다.
`FormattedMessage`	문자열	이벤트에 대한 텍스트 요약입니다.
`ProcessID`	정수	이벤트를 보고한 프로세스입니다.
`ExecutingThreadID`	정수	프로세스 내의 어떤 스레드가 이벤트를보고했습니다.
`MachineName`	문자열	이벤트를 보고하는 데스크톱 또는 서버의 이름입니다.
`Payload`	해시 테이블	문자열 키와 모든 종류의 객체를 값으로 사용하는 테이블입니다. 키는 페이로드 항목 이름이고 값

속성	유형	설명
		은 페이로드 항목의 값입니다. 페이로드는 공급자에 따라 다릅니다.

다음은 JSON으로 변환된 이벤트 예제입니다.

```
{
     "EventName": "Exception/Start",
     "ProviderName": "Microsoft-Windows-DotNETRuntime",
     "FormattedMessage": "ExceptionType=System.Exception;\r
\nExceptionMessage=Intentionally unhandled exception.;\r\nExceptionEIP=0x2ab0499;\r
\nExceptionHRESULT=-2,146,233,088;\r\nExceptionFlags=CLSCompliant;\r\nClrInstanceID=9
 ",
     "ProcessID": 3328,
     "ExecutingThreadID": 6172,
     "MachineName": "MyHost.MyCompany.com",
     "Payload":
      {
        "ExceptionType": "System.Exception",
        "ExceptionMessage": "Intentionally unhandled exception.",
        "ExceptionEIP": 44762265,
        "ExceptionHRESULT": -2146233088,
        "ExceptionFlags": 16,
        "ClrInstanceID": 9
      }
}
```

창기능카운터 소스 구성

이`WindowsPerformanceCounterSource`유형은 Windows에서 성능 카운터 메트릭을 수집합니다. 다음은 예제입니다.`WindowsPerformanceCounterSource`선언:

```
{
 "Id": "MyPerformanceCounter",
 "SourceType": "WindowsPerformanceCounterSource",
 "Categories": [{
   "Category": "Server",
   "Counters": ["Files Open", "Logon Total", "Logon/sec", "Pool Nonpaged Bytes"]
  },
```

```
   {
    "Category": "System",
    "Counters": ["Processes", "Processor Queue Length", "System Up Time"]
   },
   {
    "Category": "LogicalDisk",
    "Instances": "*",
    "Counters": [
     "% Free Space", "Avg. Disk Queue Length",
     {
      "Counter": "Disk Reads/sec",
      "Unit": "Count/Second"
     },
     "Disk Writes/sec"
    ]
   },
   {
    "Category": "Network Adapter",
    "Instances": "^Local Area Connection\* \d$",
    "Counters": ["Bytes Received/sec", "Bytes Sent/sec"]
   }
  ]
 }
```

모두`WindowsPerformanceCounterSource`선언은 다음 키-값 페어를 제공할 수 있습니다.

`SourceType`

리터럴 문자열이어야 합니다.`"WindowsPerformanceCounterSource"`(필수).

`Categories`

Windows에서 수집할 성능 카운터 메트릭 그룹 집합을 지정합니다. 각 지표 그룹에는 다음과 같은 키-값 페어가 저장됩니다.

`Category`

수집할 메트릭 카운터 집합을 지정합니다 (필수).

`Instances`

개체당 고유한 성능 카운터 집합이 있을 때 관심 있는 개체 집합을 지정합니다. 예를 들어 범주가`LogicalDisk`에는 디스크 드라이브당 일련의 성능 카운터가 있습니다. 이 키-값 페어는 선택 사항입니다. 와일드 카드를 사용할 수 있습니다.`*`및`?`를 사용하여 여러 인스턴스와 일치시킵니다. 모든 인스턴스에서 값을 집계하려면`_Total`.

다음을 사용할 수도 있습니다.`InstanceRegex`를 포함하는 정규 표현식을 받아들이는`*`와일드카드 문자를 인스턴스 이름의 일부로 사용합니다.

`Counters`

지정된 범주에 대해 수집할 메트릭을 지정합니다. 이 키-값 페어가 필요합니다. 와일드 카드를 사용할 수 있습니다.`*`및`?`를 사용하여 여러 카운터를 일치시킵니다. 다음을 지정할 수 있습니다.`Counters`이름만 사용하거나 이름과 단위를 사용하여. 카운터 단위를 지정하지 않으면 Windows용 Kinesis 에이전트는 이름에서 장치를 유추하려고 시도합니다. 이러한 추론이 올바르지 않으면 단위를 명시 적으로 지정할 수 있습니다. 변경할 수 있습니다.`Counter`이름을 지정할 수 있습니다. 카운터의 보다 복잡한 표현은 다음 키-값 페어를 사용하는 객체입니다.

`Counter`

카운터의 이름입니다. 이 키-값 페어가 필요합니다.

`Rename`

싱크대에 제시 할 카운터의 이름입니다. 이 키-값 페어는 선택 사항입니다.

`Unit`

카운터와 연관된 값의 의미입니다. 유효한 장치 이름의 전체 목록은MetricDatum의Amazon CloudWatch API 참조.

다음은 복잡한 카운터 사양을 보여주는 예제입니다.

```
{
   "Counter": "Disk Reads/sec,
   "Rename": "Disk Reads per second",
   "Unit": "Count/Second"
}
```

`WindowsPerformanceCounterSource`은 Amazon CloudWatch 싱크를 지정하는 파이프에서만 사용할 수 있습니다. Windows용 Kinesis 에이전트의 기본 제공 지표도 CloudWatch 로 스트리밍되는 경우 별도의 싱크를 사용합니다. 서비스 시작 후 Kinesis Agent for Windows용 로그를 검사하여 장치가`WindowsPerformanceCounterSource`선언을 참조하십시오. PowerShell 을 사용 하 여 범주, 인스턴스 및 카운터에 대 한 유효한 이름을 확인 합니다.

카운터 세트와 연결된 카운터를 포함하여 모든 범주에 대한 정보를 보려면 PowerShell 창에서 다음 명령을 실행합니다.

```
Get-Counter -ListSet * | Sort-Object
```

카운터 집합의 각 카운터에 사용할 수 있는 인스턴스를 확인하려면 PowerShell 창에서 다음 예제와 유사한 명령을 실행합니다.

```
Get-Counter -Counter "\Process(*)\% Processor Time"
```

의 값`Counter`매개 변수의 경로 중 하나여야 합니다.`PathsWithInstances`멤버 앞에 나열된`Get-Counter -ListSet`명령 호출로 이동합니다.

Windows용 Kinesis 에이전트 기본 제공 메트릭 소스

다음과 같은 일반적인 메트릭 소스 외에도`WindowsPerformanceCounterSource`유형 (창기능카운터 소스 구성) 에서 CloudWatch 싱크 유형은 Windows용 Kinesis 에이전트에 대한 메트릭을 수집하는 특수 소스에서 메트릭을 수신할 수 있습니다. 또한 Windows용 Kinesis 에이전트 측정치를 사용할 수도 있습니다.`KinesisTap`Windows 성능 카운터의 범주입니다.

이`MetricsFilter`키-값 쌍은 내장 Windows용 Kinesis 에이전트 메트릭 소스에서 CloudWatch로 스트리밍되는 메트릭을 지정합니다. 값은 세미콜론으로 구분된 하나 이상의 필터 표현식을 포함하는 문자열입니다. 예를 들면 다음과 같습니다.

`"MetricsFilter": "필터변환 표현식1;필터변환 표현식2"`

하나 이상의 필터 표현식과 일치하는 지표는 CloudWatch 로 스트리밍됩니다.

단일 인스턴스 메트릭은 본질적으로 전역이며 특정 소스 또는 싱크에 연결되지 않습니다. 여러 인스턴스 메트릭은 소스 또는 싱크 선언에 따라 차원적`Id`. 각 소스 또는 싱크 유형은 서로 다른 메트릭 집합을 가질 수 있습니다.

Windows용 Kinesis 에이전트 메트릭 이름의 기본 제공 목록은Windows용 Kinesis 에이전트 메트릭 목록.

단일 인스턴스 메트릭의 경우 필터 표현식은 메트릭의 이름입니다. 예를 들면 다음과 같습니다.

```
"MetricsFilter": "SourcesFailedToStart;SinksFailedToStart"
```

여러 인스턴스 메트릭의 경우 필터 표현식은 메트릭의 이름, 마침표 (`.`) 를 선택한 다음`Id`해당 메트릭을 생성 한 소스 또는 싱크 선언의 예를 들어, 싱크 선언이 있다고 가정하면`Id`의`MyFirehose`:

```
"MetricsFilter": "KinesisFirehoseRecordsFailedNonrecoverable.MyFirehose"
```

단일 인스턴스 메트릭과 다중 인스턴스 메트릭을 구분하도록 설계된 특수 와일드카드 패턴을 사용할 수 있습니다.

- 별표 (`*`) 는 마침표를 제외한 0 개 이상의 문자와 일치합니다 (`.`).
- 물음표 (`?`) 는 마침표를 제외한 한 문자와 일치합니다.
- 다른 문자는 자체 만 일치합니다.
- `_Total`는 차원에서 일치하는 모든 여러 인스턴스 값을 집계하는 특수 토큰입니다.

다음 예제에서는 모든 단일 인스턴스 지표를 찾습니다.

```
"MetricsFilter": "*"
```

별표가 기간 문자와 일치하지 않으므로 단일 인스턴스 메트릭만 포함됩니다.

다음 예제에서는 여러 인스턴스 지표를 모두 비교합니다.

```
"MetricsFilter": "*.*"
```

다음 예제에서는 모든 지표 (단일 및 다중) 와 일치합니다.

```
"MetricsFilter": "*;*.*"
```

다음 예에서는 모든 소스 및 싱크에서 여러 인스턴스 메트릭을 모두 집계합니다.

```
"MetricsFilter": "*._Total"
```

다음 예제에서는 모든 Kinesis Data Firehose 싱크에 대한 모든 Kinesis 데이터 파이어호스 지표를 집계합니다.

```
"MetricsFilter": "*Firehose*._Total"
```

다음 예제에서는 모든 단일 인스턴스 오류 메트릭과 다중 인스턴스 오류 메트릭을 일치시킵니다.

```
"MetricsFilter": "*Failed*;*Error*.*;*Failed*.*"
```

다음 예에서는 모든 소스 및 싱크에서 집계된 복구 불가능한 모든 오류 메트릭을 찾습니다.

```
"MetricsFilter": "*Nonrecoverable*._Total"
```

Windows용 Kinesis 에이전트 기본 제공 메트릭 소스를 사용하는 파이프를 지정하는 방법에 대한 자세한 내용은Windows 메트릭 파이프용 Kinesis 에이전트 구성.

Windows용 Kinesis 에이전트 메트릭 목록

다음은 Windows용 Kinesis 에이전트에서 사용할 수 있는 단일 인스턴스 및 여러 인스턴스 메트릭의 목록입니다.

단일 인스턴스 지표

다음과 같은 단일 인스턴스 지표를 사용할 수 있습니다.

`KinesisTapBuildNumber`

Windows용 Kinesis 에이전트의 버전 번호입니다.

`PipesConnected`

얼마나 많은 파이프가 소스를 싱크대에 성공적으로 연결했는지.

`PipesFailedToConnect`

얼마나 많은 파이프가 소스를 싱크대에 연결했습니까?

`SinkFactoriesFailedToLoad`

Windows용 Kinesis 에이전트에 로드되지 않은 싱크 유형이 몇 개입니까?

`SinkFactoriesLoaded`

Windows용 Kinesis 에이전트에 성공적으로 로드된 싱크 유형 수

`SinksFailedToStart`

얼마나 많은 싱크가 성공적으로 시작되지 않았습니까? 대개 잘못된 싱크 선언으로 인해.

`SinksStarted`

얼마나 많은 싱크가 성공적으로 시작되었는지.

`SourcesFailedToStart`

얼마나 많은 소스가 성공적으로 시작되지 않았습니까? 대개 잘못된 소스 선언으로 인해.

`SourcesStarted`

얼마나 많은 출처가 성공적으로 시작되었는지.

`SourceFactoriesFailedToLoad`

Windows용 Kinesis 에이전트에 로드되지 않은 소스 유형 수

`SourceFactoriesLoaded`

Windows용 Kinesis 에이전트에 성공적으로 로드된 소스 유형 수

다중 인스턴스 지표

다음과 같은 여러 인스턴스 지표를 사용할 수 있습니다.

디렉토리소스 지표

`DirectorySourceBytesRead`

이 간격 동안 읽은 바이트 수`DirectorySource`.

`DirectorySourceBytesToRead`

Windows용 Kinesis 에이전트에서 아직 읽지 않은 읽을 수 있는 알려진 바이트 수입니다.

`DirectorySourceFilesToProcess`

Windows용 Kinesis 에이전트에서 아직 검사하지 않은 알려진 파일 수

`DirectorySourceRecordsRead`

이 간격 동안 읽은 레코드 수`DirectorySource`.

창로그소스 지표

`EventLogSourceEventsError`

얼마나 많은 Windows 이벤트 로그 이벤트를 성공적으로 읽지 못했습니다.

`EventLogSourceEventsRead`

얼마나 많은 Windows 이벤트 로그 이벤트를 성공적으로 읽었습니다.

KinesisFirehose 싱크 메트릭

`KinesisFirehoseBytesAccepted`

간격 동안 허용되는 바이트 수입니다.

`KinesisFirehoseClientLatency`

레코드 생성과 레코드 스트리밍 사이에 Kinesis Data Firehose 서비스에 전달된 시간이 얼마나 걸렸습니다.

`KinesisFirehoseLatency`

Kinesis Data Firehose 서비스에 대한 레코드 스트리밍의 시작과 종료 사이에 경과된 시간

`KinesisFirehoseNonrecoverableServiceErrors`

재시도에도 불구하고 Kinesis Data Firehose 서비스에 오류 없이 레코드를 전송할 수 없는 횟수입니다.

`KinesisFirehoseRecordsAttempted`

Kinesis Data Firehose 서비스로 스트리밍하려고 시도한 레코드의 수입니다.

`KinesisFirehoseRecordsFailedNonrecoverable`

재시도에도 불구하고 Kinesis Data Firehose 서비스에 성공적으로 스트리밍되지 않은 레코드 수

`KinesisFirehoseRecordsFailedRecoverable`

얼마나 많은 레코드가 Kinesis Data Firehose 서비스로 성공적으로 스트리밍되었으나 재시도만 가능합니다.

`KinesisFirehoseRecordsSuccess`

재시도 없이 Kinesis Data Firehose 서비스로 성공적으로 스트리밍된 레코드 수

`KinesisFirehoseRecoverableServiceErrors`

얼마나 많은 레코드가 Kinesis Data Firehose 서비스로 성공적으로 전송될 수 있었지만 재시도만 가능합니다.

KinesisStream 지표

`KinesisStreamBytesAccepted`

간격 동안 허용되는 바이트 수입니다.

`KinesisStreamClientLatency`

레코드 생성과 레코드 스트리밍 사이에 Kinesis Data Streams 서비스로 전달된 시간

`KinesisStreamLatency`

Kinesis Data Streams 서비스에 대한 레코드 스트리밍의 시작과 끝 사이에 경과된 시간입니다.

`KinesisStreamNonrecoverableServiceErrors`

재시도에도 불구하고 Kinesis Data Streams 서비스에 오류 없이 레코드를 전송할 수 없는 횟수입니다.

`KinesisStreamRecordsAttempted`

Kinesis Data Streams 서비스로 스트리밍하려고 시도한 레코드 수입니다.

`KinesisStreamRecordsFailedNonrecoverable`

재시도에도 불구하고 Kinesis Data Streams 서비스에 성공적으로 스트리밍되지 않은 레코드 수입니다.

`KinesisStreamRecordsFailedRecoverable`

얼마나 많은 레코드가 Kinesis Data Streams 서비스로 성공적으로 스트리밍되었으나 재시도만 가능합니다.

`KinesisStreamRecordsSuccess`

재시도 없이 Kinesis Data Streams 서비스로 성공적으로 스트리밍된 레코드 수입니다.

`KinesisStreamRecoverableServiceErrors`

Kinesis Data Streams 서비스에 성공적으로 레코드가 전송될 수 있지만 재시도만 가능합니다.

CloudWatch Log 지표

`CloudWatchLogBytesAccepted`

간격 동안 허용되는 바이트 수입니다.

`CloudWatchLogClientLatency`

CloudWatch Logs 서비스에 대한 레코드 생성과 레코드 스트리밍 사이에 경과된 시간입니다.

`CloudWatchLogLatency`

CloudWatch Logs 서비스에 대한 레코드 스트리밍의 시작과 끝 사이에 경과된 시간입니다.

`CloudWatchLogNonrecoverableServiceErrors`

재시도에도 불구하고 CloudWatch Logs 서비스에 오류 없이 레코드를 전송할 수 없는 횟수입니다.

`CloudWatchLogRecordsAttempted`

CloudWatch Logs 서비스로 스트리밍하려고 시도한 레코드 수입니다.

`CloudWatchLogRecordsFailedNonrecoverable`

재시도에도 불구하고 CloudWatch Logs 서비스로 스트리밍되지 않은 레코드 수입니다.

`CloudWatchLogRecordsFailedRecoverable`

CloudWatch Logs 서비스로 성공적으로 스트리밍된 레코드 수 (재시도 횟수만 포함)

`CloudWatchLogRecordsSuccess`

다시 시도 없이 CloudWatch Logs 서비스로 성공적으로 스트리밍된 레코드 수입니다.

`CloudWatchLogRecoverableServiceErrors`

레코드를 CloudWatch Logs 서비스로 성공적으로 전송할 수 있지만 재시도만 할 수 있습니다.

CloudWatch 측정치

`CloudWatchLatency`

CloudWatch 서비스에 대한 지표 스트리밍의 시작과 끝 사이에 평균 시간이 경과된 시간입니다.

`CloudWatchNonrecoverableServiceErrors`

재시도에도 불구하고 CloudWatch 서비스에 오류 없이 지표를 전송할 수 없는 횟수입니다.

`CloudWatchRecoverableServiceErrors`

지표가 CloudWatch 서비스에 오류 없이 전송되었으나 재시도로만 전송된 횟수입니다.

`CloudWatchServiceSuccess`

재시도 없이 CloudWatch 서비스에 오류 없이 지표가 전송된 횟수입니다.

책갈피 구성

기본적으로 Windows용 Kinesis 에이전트는 에이전트가 시작된 후 생성된 싱크로 로그 레코드를 전송합니다. 때로는 자동 업데이트 중에 Kinesis Agent가 중지되는 기간 동안 생성된 로그 레코드와 같이 이전 로그 레코드를 전송하는 것이 유용합니다. 북마크 기능은 싱크로 전송 된 레코드를 추적합니다. Windows용 Kinesis 에이전트가 북마크 모드에 있고 시작되면 이후에 생성된 모든 로그 레코드와 함께 Windows용 Kinesis 에이전트가 중지된 후 생성된 모든 로그 레코드가 전송됩니다. 이 동작을 제어하기 위해 파일 기반 소스 선언에는 선택적으로 다음 키 - 값 쌍이 포함될 수 있습니다.

`InitialPosition`

책갈피의 초기 상황을 지정합니다. 가능한 값은 다음과 같습니다.

`EOS`

EOS (스트림의 끝) 를 지정합니다. 에이전트가 실행되는 동안 생성된 로그 레코드만 싱크로 전송됩니다.

`0`

사용 가능한 모든 로그 레코드와 이벤트가 처음에 전송됩니다. 그런 다음 북마크가 생성되어 북마크가 생성된 후 생성된 모든 새 로그 레코드와 이벤트가 결국 전송되도록 하기 위해 Kinesis Agent for Windows용 실행 여부와 상관없이 북마크가 생성됩니다.

`Bookmark`

북마크는 최신 로그 레코드 또는 이벤트 직후에 초기화됩니다. 그런 다음 북마크가 생성되어 북마크가 생성된 후 생성된 모든 새 로그 레코드와 이벤트가 결국 전송되도록 하기 위해 Kinesis Agent for Windows용 실행 여부와 상관없이 북마크가 생성됩니다.

책갈피는 기본적으로 활성화되어 있습니다. 파일은`%ProgramData%\Amazon\KinesisTap`디렉토리에 추가합니다.

`Timestamp`

로그 레코드 및 이벤트 후에 생성 된`InitialPositionTimestamp`값 (정의 다음) 이 전송됩니다. 그런 다음 책갈피가 생성되어 북마크가 생성된 후 생성된 모든 새 로그 레코드와 이벤트가 결국 Windows용 Kinesis 에이전트가 실행 중인지 여부에 관계없이 전송됩니다.

`InitialPositionTimestamp`

원하는 가장 빠른 로그 레코드 또는 이벤트 타임스탬프를 지정합니다. 이 키-값 페어를 지정하는 경우`InitialPosition`의 값은`Timestamp`.

`BookmarkOnBufferFlush`

이 설정은 북마크 가능한 소스에 추가할 수 있습니다. 로 설정된 경우`true`를 사용하면 싱크가 AWS 에 이벤트를 성공적으로 발송할 때만 책갈피 업데이트가 수행되도록 할 수 있습니다. 단일 싱크만 소스에 가입할 수 있습니다. 로그를 여러 대상으로 발송하는 경우 소스를 복제하여 데이터 손실과 관련된 잠재적인 문제를 방지하십시오.

Windows용 Kinesis 에이전트가 오랫동안 중지된 경우 북마크된 로그 레코드와 이벤트가 더 이상 존재하지 않을 수 있으므로 해당 책갈피를 삭제해야 할 수 있습니다. 주어진 북마크 파일소스 ID은 에 위치합니다.`%PROGRAMDATA%\Amazon\AWSKinesisTap\`*`source id`*`.bm`.

이름이 바뀌거나 잘린 파일에는 책갈피가 작동하지 않습니다. ETW 이벤트 및 성능 카운터의 특성으로 인해 북마크할 수 없습니다.

sink 선언

sink 선언는 다양한 AWS 서비스로 전송해야 하는 로그, 이벤트 및 메트릭을 지정할 수 있습니다. 다음 섹션에서는 Microsoft Windows용 Amazon Kinesis 에이전트에서 사용할 수 있는 기본 제공 싱크 유형의 구성에 대해 설명합니다. Windows용 Kinesis 에이전트는 확장할 수 있으므로 사용자 정의 싱크 유형을 추가할 수 있습니다. 각 싱크 유형은 일반적으로 해당 싱크 유형과 관련된 구성 선언에서 고유 한 키 - 값 쌍을 필요로합니다.

모든 싱크 선언은 다음 키-값 페어를 포함할 수 있습니다.

`Id`

구성 파일 (필수) 에서 특정 싱크를 식별하는 고유한 문자열입니다.

`SinkType`

이 싱크에 대한 싱크 유형의 이름입니다 (필수 사항). 싱크 유형은 이 싱크에서 스트리밍되는 로그, 이벤트 또는 메트릭 데이터의 대상을 지정합니다.

`AccessKey`

싱크 유형과 연결된 AWS 서비스에 대한 액세스 권한을 부여할 때 사용할 AWS 액세스 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 자세한 내용은 sink 보안 구성 섹션을 참조하세요.

`SecretKey`

싱크 유형과 연결된 AWS 서비스에 대한 액세스 권한을 부여할 때 사용할 AWS 보안 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 자세한 내용은 sink 보안 구성 섹션을 참조하세요.

`Region`

스트리밍을 위한 대상 리소스가 포함된 AWS 리전을 지정합니다. 이 키-값 페어는 선택 사항입니다.

`ProfileName`

인증에 사용할 AWS 프로필을 지정합니다. 이 키 - 값 쌍은 선택 사항이지만, 지정된 경우, 지정된 액세스 키와 비밀 키를 재정의합니다. 자세한 내용은 sink 보안 구성 섹션을 참조하세요.

`RoleARN`

싱크 유형과 연결된 AWS 서비스에 액세스할 때 사용할 IAM 역할을 지정합니다. 이 옵션은 Windows용 Kinesis 에이전트가 EC2 인스턴스에서 실행 중이지만 인스턴스 프로필에서 참조하는 역할보다 다른 역할이 더 적합할 때 유용합니다. 예를 들어 교차 계정 역할을 사용하여 EC2 인스턴스와 동일한 AWS 계정에 있지 않은 리소스를 타겟팅할 수 있습니다. 이 키-값 페어는 선택 사항입니다.

`Format`

스트리밍 전에 로그 및 이벤트 데이터에 적용되는 직렬화 종류를 지정합니다. 유효한 값은 `json` 및 `xml`입니다. 이 옵션은 데이터 파이프라인의 다운스트림 분석에서 특정 형식의 데이터를 요구하거나 선호하는 경우에 유용합니다. 이 키-값 쌍은 선택 사항이며, 지정하지 않으면 소스의 일반 텍스트가 싱크에서 싱크 유형과 연결된 AWS 서비스로 스트리밍됩니다.

`TextDecoration`

없는 경우`Format`가 지정되어 있는 경우`TextDecoration`는 로그 또는 이벤트 레코드를 스트리밍할 때 포함할 추가 텍스트를 지정합니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

`ObjectDecoration`

일시`Format`가 지정되어 있는 경우`ObjectDecoration`는 직렬화 및 스트리밍 전에 로그 또는 이벤트 레코드에 포함될 추가 데이터를 지정합니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

`BufferInterval`

싱크 유형과 연결된 AWS 서비스에 대한 API 호출을 최소화하기 위해 Windows용 Kinesis 에이전트는 스트리밍하기 전에 여러 로그, 이벤트 또는 메트릭 레코드를 버퍼링합니다. 이를 통해 API 호출당 청구되는 서비스에 대한 비용을 절감할 수 있습니다.`BufferInterval`는 AWS 서비스로 스트리밍하기 전에 레코드가 버퍼링되어야 하는 최대 시간 (초) 을 지정합니다. 이 키 - 값 쌍은 선택 사항이며, 지정된 경우 값을 나타내는 문자열을 사용합니다.

`BufferSize`

싱크 유형과 연결된 AWS 서비스에 대한 API 호출을 최소화하기 위해 Windows용 Kinesis 에이전트는 스트리밍하기 전에 여러 로그, 이벤트 또는 메트릭 레코드를 버퍼링합니다. 이를 통해 API 호출당 청구되는 서비스에 대한 비용을 절감할 수 있습니다.`BufferSize`는 AWS 서비스로 스트리밍하기 전에 버퍼링할 최대 레코드 수를 지정합니다. 이 키 - 값 쌍은 선택 사항이며, 지정된 경우 값을 나타내는 문자열을 사용합니다.

`MaxAttempts`

일관되게 스트리밍이 실패할 경우 Kinesis Agent가 일련의 로그, 이벤트 및 메트릭 레코드를 AWS 서비스로 스트리밍하려고 시도하는 최대 횟수를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 문자열을 사용하여 값을 나타냅니다. 기본값은"3".

다양한 종류의 싱크를 사용하는 전체 구성 파일의 예는Windows 응용 프로그램 이벤트 로그에서 싱크로 스트리밍.

주제

- KinesisStreamsink 구성
- KinesisFirehosesink 구성
- CloudWatch 싱크 구성
- CloudWatchLogssink 구성
- 로컬FileSystemsink 구성
- sink 보안 구성
- 구성ProfileRefreshingAWSCredentialProviderAWS 자격 증명 새로 고침
- 싱크 장식 구성
- 싱크 변수 대체 구성
- 싱크대 대기열 구성
- 싱크에 대한 프록시 구성
- 더 많은 싱크 속성에서 해석 변수 구성
- AWS 싱크에서 RoleARN 속성을 사용할 때 AWS STS 리전 엔드포인트 구성
- AWS 싱크용 VPC 엔드포인트 구성
- 대체 프록시 수단 구성

KinesisStreamsink 구성

이KinesisStream싱크 유형은 Kinesis Data Streams 서비스에 기록과 이벤트를 기록합니다. 일반적으로 Kinesis 데이터 스트림으로 스트리밍되는 데이터는 다양한 AWS 서비스를 사용하여 실행되는 하나 이상의 사용자 지정 애플리케이션에 의해 처리됩니다. 데이터는 Kinesis 데이터 스트림을 사용하여 구성된 명명된 스트림으로 스트리밍됩니다. 자세한 내용은 단원을 참조하십시오.Amazon Kinesis Data Streams 개발자 가이드.

다음은 Kinesis Data Streams 싱크 선언의 예입니다.

```
{
    "Id": "TestKinesisStreamSink",
    "SinkType": "KinesisStream",
    "StreamName": "MyTestStream",
    "Region": "us-west-2"
}
```

모두KinesisStream싱크 선언은 다음과 같은 추가 키-값 페어를 제공할 수 있습니다.

`SinkType`

을 지정해야 하며 값은 리터럴 문자열이어야 합니다.KinesisStream.

`StreamName`

에서 스트리밍된 데이터를 수신하는 Kinesis 데이터 스트림의 이름을 지정합니다.KinesisStream싱크 유형 (필수). 데이터를 스트리밍하기 전에 AWS 관리 콘솔, AWS CLI에서 또는 Kinesis 데이터 스트림 API를 사용하는 애플리케이션을 통해 스트림을 구성합니다.

`RecordsPerSecond`

초당 Kinesis Data Streams 으로 스트리밍되는 최대 레코드 수를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 정수를 사용하여 값을 나타냅니다. 기본 값은 1000 개 레코드입니다.

`BytesPerSecond`

Kinesis Data Streams 으로 스트리밍되는 초당 최대 바이트 수를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 정수를 사용하여 값을 나타냅니다. 기본값은 1MB입니다.

기본 설정BufferInterval이 싱크 유형의 경우 1 초이고 기본BufferSize은 500개의 레코드입니다.

`KinesisFirehose`sink 구성

이`KinesisFirehose`싱크 유형은 Kinesis Data Firehose 서비스에 로그 레코드와 이벤트를 스트리밍합니다. Kinesis Data Firehose 는 스트리밍된 데이터를 스토리지용으로 다른 서비스로 제공합니다. 일반적으로 저장된 데이터는 데이터 파이프라인의 후속 단계에서 분석됩니다. 데이터는 Kinesis Data Firehose 를 사용하여 구성된 명명된 전송 스트림으로 스트리밍됩니다. 자세한 내용은 단원을 참조하십시오.Amazon Kinesis Data Firehose se를 개발자 가이드.

다음은 Kinesis Data Firehose 싱크 선언의 예입니다.

```
{
   "Id": "TestKinesisFirehoseSink",
   "SinkType": "KinesisFirehose",
   "StreamName": "MyTestFirehoseDeliveryStream",
   "Region": "us-east-1",
   "CombineRecords": "true"
}
```

모두`KinesisFirehose`싱크 선언은 다음과 같은 추가 키-값 페어를 제공할 수 있습니다.

`SinkType`

을 지정해야 하며 값은 리터럴 문자열이어야 합니다.`KinesisFirehose`.

`StreamName`

에서 스트리밍된 데이터를 수신하는 Kinesis Data Firehose 전송 스트림의 이름을 지정합니다.`KinesisStream`싱크 유형 (필수). 데이터를 스트리밍하기 전에 AWS 관리 콘솔, AWS CLI를 사용하거나 Kinesis Data Firehose API를 사용하는 애플리케이션을 통해 전송 스트림을 구성합니다.

`CombineRecords`

로 설정된 경우`true`는 여러 개의 작은 레코드를 최대 크기가 5KB인 큰 레코드로 결합하도록 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 함수를 사용하여 결합 된 레코드는\n. AWS Lambda 를 사용하여 Kinesis Data Firehose 레코드를 변환하는 경우 Lambda 함수에서 구분 문자를 고려해야 합니다.

`RecordsPerSecond`

초당 Kinesis Data Streams 으로 스트리밍되는 최대 레코드 수를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 정수를 사용하여 값을 나타냅니다. 기본값은 5000개 레코드입니다.

`BytesPerSecond`

초당 Kinesis Data Streams 으로 스트리밍되는 최대 바이트 수를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 정수를 사용하여 값을 나타냅니다. 기본값은 5MB입니다.

기본 설정`BufferInterval`이 싱크 유형의 경우 1 초이고 기본`BufferSize`은 500개의 레코드입니다.

CloudWatch 싱크 구성

이`CloudWatch`싱크 유형은 메트릭을 CloudWatch 서비스로 스트리밍합니다. AWS Management Console에서 지표를 볼 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오.

다음은 예제입니다.`CloudWatch`sink 선언:

```
{
   "Id": "CloudWatchSink",
   "SinkType": "CloudWatch"
}
```

모두`CloudWatch`싱크 선언은 다음과 같은 추가 키-값 페어를 제공할 수 있습니다.

`SinkType`

을 지정해야 하며 값은 리터럴 문자열이어야 합니다.`CloudWatch`.

`Interval`

Windows용 Kinesis 에이전트가 메트릭을 CloudWatch 서비스에 보고하는 빈도 (초) 를 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정된 경우 정수를 사용하여 값을 나타냅니다. 기본 값은 60초입니다. 고해상도 CloudWatch 메트릭을 원할 경우 1초를 지정합니다.

`Namespace`

지표 데이터가 보고되는 CloudWatch 네임스페이스를 지정합니다. CloudWatch 네임스페이스는 지표 집합을 함께 그룹화합니다. 이 키-값 페어는 선택 사항입니다. 기본 값은 `KinesisTap`입니다.

`Dimensions`

네임스페이스 내에서 메트릭 집합을 격리하는 데 사용되는 CloudWatch 차원을 지정합니다. 예를 들어 각 데스크톱 또는 서버에 대해 별도의 메트릭 데이터 세

트를 제공하는 데 유용할 수 있습니다. 이 키-값 쌍은 선택 사항이며, 지정된 경우 값은 다음 형식을 준수해야 합니다."키1=Value1;키 2=Value2...". 기본 값은 `"ComputerName={computername};InstanceId={instance_id}"`입니다. 이 값은 싱크 변수 대체를 지원합니다. 자세한 내용은 싱크 변수 대체 구성 섹션을 참조하세요.

`MetricsFilter`

내장된 Windows용 Kinesis 에이전트 메트릭 소스에서 CloudWatch 로 스트리밍되는 메트릭을 지정합니다. 이 키-값 쌍의 값 구문 세부 정보를 포함하여 기본 제공 Kinesis Windows용 에이전트 메트릭 소스에 대한 자세한 내용은Windows용 Kinesis 에이전트 기본 제공 메트릭 소스.

`CloudWatchLogs`sink 구성

이`CloudWatchLogs`싱크 유형은 Amazon CloudWatch Logs Logs로 기록과 이벤트를 스트림합니다. AWS Management Console에서 로그를 보거나 데이터 파이프라인의 추가 단계를 통해 로그를 처리할 수 있습니다. 데이터는 CloudWatch Logs 로그에 구성된 명명된 로그 스트림으로 스트리밍됩니다. 로그 스트림은 명명된 로그 그룹으로 구성됩니다. 자세한 내용은 단원을 참조하십시오.Amazon CloudWatch Logs.

다음은 CloudWatch Logs 싱크 선언의 예입니다.

```
{
   "Id": "MyCloudWatchLogsSink",
   "SinkType": "CloudWatchLogs",
   "BufferInterval": "60",
   "BufferSize": "100",
   "Region": "us-west-2",
   "LogGroup": "MyTestLogGroup",
   "LogStream": "MyTestStream"
}
```

모두`CloudWatchLogs`싱크 선언은 다음과 같은 추가 키-값 페어를 제공해야 합니다.

`SinkType`

리터럴 문자열이어야 합니다.`CloudWatchLogs`.

`LogGroup`

에서 스트리밍되는 로그 스트림과 이벤트 레코드를 수신하는 로그 스트림이 들어 있는 CloudWatch Logs 그룹의 이름을 지정합니다.`CloudWatchLogs`sink 타입입니다. 지정된 로그 그룹이 존재하지 않는 경우 Windows용 Kinesis 에이전트가 해당 그룹을 생성하려고 합니다.

`LogStream`

CloudWatch Logs 스트림의 이름을 지정합니다.`CloudWatchLogs`sink 타입입니다. 이 값은 싱크 변수 대체를 지원합니다. 자세한 내용은 싱크 변수 대체 구성 섹션을 참조하세요. 지정된 로그 스트림이 존재하지 않는 경우 Windows용 Kinesis 에이전트가 해당 스트림을 생성하려고 합니다.

기본 설정`BufferInterval`이 싱크 유형의 경우 1 초이고 기본`BufferSize`은 500개의 레코드입니다. 최대 버퍼 크기는 10,000개 레코드입니다.

로컬`FileSystem`sink 구성

싱크 타입`FileSystem`는 로그 및 이벤트 레코드를 AWS 서비스로 스트리밍하지 않고 로컬 파일 시스템의 파일에 저장합니다.`FileSystem`싱크는 테스트 및 진단에 유용합니다. 예를 들어, 이 싱크 유형을 사용하여 레코드를 AWS 로 보내기 전에 레코드를 검사할 수 있습니다.

다음으로 바꿉니다.`FileSystem`싱크의 경우 구성 파라미터를 사용하여 일괄 처리, 제한 및 오류 발생 시 재시도를 시뮬레이션하여 실제 AWS 싱크의 동작을 모방할 수도 있습니다.

에 연결된 모든 소스의 모든 레코드`FileSystem`싱크는 다음과 같이 지정된 단일 파일에 저장됩니다.`FilePath`. 다음의 경우,`FilePath`지정하지 않으면 레코드가 라는 파일에 저장됩니다.*`SinkId`*`.txt`의`%TEMP%`디렉토리로, 일반적으로`C:\Users\`*`UserName`*`\AppData\Local\Temp`를 참조하십시오.*`SinkId`*은 싱크의 고유 식별자입니다.*`UserName`*은 활성 사용자의 Windows 사용자 이름입니다.

이 싱크 유형은 텍스트 장식 속성을 지원합니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요.

예제`FileSystem`싱크 유형 구성은 다음 예시와 같이 표시됩니다.

```
{
    "Id": "LocalFileSink",
    "SinkType": "FileSystem",
    "FilePath": "C:\\ProgramData\\Amazon\\local_sink.txt",
    "Format": "json",
    "TextDecoration": "",
    "ObjectDecoration": ""
}
```

이`FileSystem`구성은 다음과 같은 키-값 페어입니다.

`SinkType`

리터럴 문자열이어야 합니다.`FileSystem`.

`FilePath`

레코드가 저장되는 경로와 파일을 지정합니다. 이 키-값 페어는 선택 사항입니다. 지정하지 않은 경우 기본값은 입니다.*`TempPath`*`\\`*`SinkId`*`.txt`를 참조하십시오.*`TempPath`*에 저장된 폴더입니다.`%TEMP%`및 변수*`SinkId`*은 싱크의 고유 식별자입니다.

`Format`

이벤트의 형식이 될 지정합니다.`json`또는`xml`. 이 키 값 쌍은 선택 사항이며 대/소문자를 구분하지 않습니다. 생략하면 이벤트가 일반 텍스트로 파일에 기록됩니다.

`TextDecoration`

일반 텍스트로 작성된 이벤트에만 적용됩니다. 이 키-값 페어는 선택 사항입니다.

`ObjectDecoration`

이벤트에만 적용됩니다.`Format`다음의 경우 이 로 설정됩니다.`json`. 이 키-값 페어는 선택 사항입니다.

고급 사용 — 스로틀링 및 실패 시뮬레이션 기록

`FileSystem`는 레코드 제한을 시뮬레이션하여 AWS 싱크의 동작을 모방할 수 있습니다. 다음 키-값 페어를 사용하여 레코드 제한 및 실패 시뮬레이션 속성을 지정할 수 있습니다.

대상 파일에 대한 잠금을 획득하고 쓰기를 방지하여`FileSystem`싱크를 사용하여 네트워크 실패 시 AWS 싱크의 동작을 시뮬레이션하고 검사합니다.

다음 예제는`FileSystem`구성을 시뮬레이션 속성과 함께 사용합니다.

```
{
    "Id": "LocalFileSink",
    "SinkType": "FileSystem",
    "FilePath": "C:\\ProgramData\\Amazon\\local_sink.txt",
    "TextDecoration": "",
    "RequestsPerSecond": "100",
    "BufferSize": "10",
    "MaxBatchSize": "1024"
}
```

`RequestsPerSecond`

선택 사항이며 문자열 유형으로 지정됩니다. 생략할 경우 기본값은 입니다."5". 레코드 수가 아니라 싱크가 처리하는 요청 속도 (즉, 파일에 쓰기) 를 제어합니다. Windows용 Kinesis 에이전트는 AWS 엔드포인트에 일괄 요청을 하므로 요청에 여러 레코드가 포함될 수 있습니다.

`BufferSize`

선택 사항이며 문자열 유형으로 지정됩니다. 싱크가 파일에 저장하기 전에 배치되는 최대 이벤트 레코드 수를 지정합니다.

`MaxBatchSize`

선택 사항이며 문자열 유형으로 지정됩니다. 싱크가 파일에 저장하기 전에 배치되는 최대 이벤트 레코드 데이터 양 (바이트) 을 지정합니다.

최대 레코드 속도 제한은`BufferSize`은 요청 당 최대 레코드 수를 결정하고`RequestsPerSecond`. 다음 공식을 사용하여 초당 레코드 속도 제한을 계산할 수 있습니다.

레코드레이트=`BufferSize*RequestsPerSecond`

위의 예에서 구성 값을 감안할 때 초당 1000 레코드의 최대 레코드 속도가 있습니다.

sink 보안 구성

인증 구성

Windows용 Kinesis 에이전트가 로그, 이벤트 및 메트릭을 AWS 서비스에 스트리밍하려면 액세스를 인증해야 합니다. Windows용 Kinesis 에이전트에 인증을 제공하는 데에는 다음과 같은 몇 가지 방법이 있습니다. 실행 방법은 Windows용 Kinesis 에이전트가 실행 중인 상황과 특정 조직에 대한 특정 보안 요구 사항에 따라 다릅니다.

- Windows용 Kinesis 에이전트가 Amazon EC2 호스트에서 실행 중인 경우 인증을 제공하는 가장 안전하고 간단한 방법은 필수 AWS 서비스에 필요한 작업에 충분한 액세스 권한을 가진 IAM 역할과 해당 역할을 참조하는 EC2 인스턴스 프로필을 생성하는 것입니다. 인스턴스 프로필 생성에 대한 자세한 내용은 단원을 참조하십시오.인스턴스 프로파일 사용. IAM 역할에 추가해야 하는 정책에 대한 자세한 내용은 단원을 참조하십시오. 권한 부여 구성.

  인스턴스 프로필을 생성한 후 Windows용 Kinesis 에이전트를 사용하는 EC2 인스턴스와 연결할 수 있습니다. 인스턴스에 이미 연결된 인스턴스 프로필이 있는 경우 해당 인스턴스 프로필과 연결된 역할에 적절한 정책을 연결할 수 있습니다.

- Windows용 Kinesis 에이전트가 한 계정의 EC2 호스트에서 실행되지만 싱크의 대상인 리소스가 다른 계정에 상주하는 경우 교차 계정 액세스를 위한 IAM 역할을 생성할 수 있습니다. 자세한 내용은 단원을 참조하십시오.자습서: IAM 역할을 사용하여 AWS 계정 간 액세스 권한 위임. 교차 계정 역할을 생성한 후 교차 계정 역할의 Amazon 리소스 이름 (ARN) 을`RoleARN`싱크 선언에서 키-값 페어입니다. 그런 다음 Windows용 Kinesis 에이전트는 해당 싱크의 싱크 유형과 연결된 AWS 리소스에 액세스할 때 지정된 교차 계정 역할을 수임하려고 시도합니다.
- Windows용 Kinesis 에이전트가 Amazon EC2 외부에서 실행 중인 경우 (예: 온프레미스) 다음과 같은 몇 가지 옵션이 있습니다.
  - 온프레미스 서버 또는 데스크톱 시스템을 Amazon EC2 Systems Manager 관리 인스턴스로 등록할 수 있는 경우 다음 프로세스를 사용하여 인증을 구성합니다.
    1. 에 설명된 프로세스를 사용하십시오.하이브리드 환경에서 AWS Systems Manager 설정를 사용하여 서비스 역할을 만들고, 관리되는 인스턴스에 대한 활성화를 만들고, SSM 에이전트를 설치합니다.
    2. 적절한 정책을 서비스 역할에 연결하여 Kinesis Agent가 구성된 싱크에서 데이터를 스트리밍하는 데 필요한 리소스에 액세스할 수 있도록 합니다. IAM 역할에 추가해야 하는 정책에 대한 자세한 내용은 단원을 참조하십시오. 권한 부여 구성.
    3. 에 설명된 프로세스를 사용하십시오.구성`ProfileRefreshingAWSCredentialProvider`AWS 자격 증명 새로 고침AWS 자격 증명을 새로 고칩니다.

    SSM 및 AWS 에서 자격 증명을 안전하게 관리하므로 비 EC2 인스턴스에 권장되는 접근 방식입니다.
  - 기본 시스템 계정 대신 특정 사용자로 Kinesis Windows용 에이전트용 AWSkinesistap 서비스를 실행할 수 있는 경우 다음 프로세스를 사용합니다.
    1. AWS 서비스를 사용할 AWS 계정에서 IAM 사용자를 생성합니다. 생성 프로세스 중에 이 사용자의 액세스 키와 보안 키를 캡처합니다. 이 프로세스의 이후 단계를 위해 이 정보가 필요합니다.
    2. 필요한 서비스에 필요한 작업에 대한 액세스 권한을 부여하는 정책을 IAM 사용자에게 연결합니다. IAM 사용자에게 연결할 정책에 대한 자세한 내용은 권한 부여 구성.
    3. 각 데스크톱 또는 서버에서 AWSkinesistap 서비스를 변경하여 기본 시스템 계정이 아닌 특정 사용자에서 실행되도록 합니다.
    4. 앞에서 기록한 액세스 키와 보안 키를 사용하여 SDK 저장소에 프로필을 생성합니다. 자세한 내용은 AWS 자격 증명 구성을 참조하십시오.

5. 업데이트`AWSKinesisTap.exe.config`파일의`%PROGRAMFILES%\Amazon\AWSKinesisTap`디렉토리를 클릭하여 이전 단계에서 생성된 프로파일의 이름을 지정합니다. 자세한 내용은 AWS 자격 증명 구성을 참조하십시오.

이는 특정 호스트 및 특정 사용자에 대해 자격 증명이 암호화되므로 관리할 수 없는 비 EC2 호스트에 권장되는 방법입니다.

- 기본 시스템 계정에서 Windows용 Kinesis 에이전트용 AWSkinesisTap 서비스를 실행해야 하는 경우 공유 자격 증명 파일을 사용해야 합니다. 이는 시스템 계정에 SDK 저장소를 사용하도록 설정하기 위한 Windows 사용자 프로필이 없기 때문입니다. 공유 자격 증명 파일은 암호화되지 않으므로 이 방법을 사용하지 않는 것이 좋습니다. 공유 구성 파일을 사용하는 방법에 대한 자세한 내용은AWS 자격 증명 구성의.NET용 AWS SDK. 이 방법을 사용하는 경우 NTFS 암호화 및 공유 구성 파일에 대한 제한된 파일 액세스를 사용하는 것이 좋습니다. 키는 관리 플랫폼에 의해 교체되어야 하며, 키 순환이 발생할 때 공유 구성 파일을 업데이트해야 합니다.

싱크 선언에 액세스 키와 비밀 키를 직접 제공 할 수는 있지만 선언이 암호화되지 않기 때문에이 방법은 권장되지 않습니다.

권한 부여 구성

다음 정책을 따르는 적절한 정책을 Windows용 Kinesis 에이전트가 AWS 서비스로 데이터를 스트리밍하는 데 사용할 IAM 사용자 또는 역할에 연결합니다.

Kinesis Data Streams

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": "arn:aws:kinesis:*:*:stream/*"
        }
    ]
}
```

특정 리전, 계정 또는 스트림 이름으로 권한을 제한하려면 ARN 의 적절한 별표를 특정 값으로 바꿉니다. 자세한 내용은 의 “Kinesis Data Streams 의 Amazon 리소스 이름 (ARN) 을 참조하십시오.IAM을 사용하여 Amazon Kinesis Data Streams 리소스에 대한 액세스 제어.

Kinesis Data Firehose

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch"
            ],
            "Resource": "arn:aws:firehose:*:*:deliverystream/*"
        }
    ]
}
```

특정 리전, 계정 또는 전송 스트림 이름으로 권한을 제한하려면 ARN 의 적절한 별표를 특정 값으로 바꿉니다. 자세한 내용은 단원을 참조하십시오.Amazon Kinesis Data Firehose 를 통한 액세스 제어의Amazon Kinesis Data Firehose se를 개발자 가이드.

CloudWatch

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        }
    ]
}
```

자세한 내용은 단원을 참조하십시오.CloudWatch 리소스에 대한 액세스 권한 관리 개요의Amazon CloudWatch Logs.

기존 로그 그룹 및 로그 스트림이 있는 CloudWatch 로그

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor3",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:*"
        },
        {
            "Sid": "VisualEditor4",
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:*:*:log-group:*:*:*"
        }
    ]
}
```

특정 리전, 계정, 로그 그룹 또는 로그 스트림에 대한 액세스를 제한하려면 ARN의 적절한 별표를 적절한 값으로 바꿉니다. 자세한 내용은 단원을 참조하십시오.CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요의Amazon CloudWatch Logs.

Kinesis 에이전트에서 로그 그룹 및 로그 스트림을 생성할 수 있는 추가 권한이 있는 CloudWatch 로그

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor5",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
```

```
            "Resource": "arn:aws:logs:*:*:log-group:*"
        },
        {
            "Sid": "VisualEditor6",
            "Effect": "Allow",
            "Action": "logs:PutLogEvents",
            "Resource": "arn:aws:logs:*:*:log-group:*:*:*"
        },
        {
            "Sid": "VisualEditor7",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

특정 리전, 계정, 로그 그룹 또는 로그 스트림에 대한 액세스를 제한하려면 ARN의 적절한 별표를 적절한 값으로 바꿉니다. 자세한 내용은 단원을 참조하십시오.CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요의Amazon CloudWatch Logs.

EC2 태그 변수 확장에 필요한 권한

함께 변수 확장을 사용하여`ec2tag`변수 접두사에는`ec2:Describe*`권한을 부여합니다.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "VisualEditor8",
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    }
   ]
}
```

> **Note**
>
> 여러 명령문을 단일 정책으로 결합할 수 있습니다.`Sid`는 해당 정책 내에서 고유합니다. 정책 생성에 대한 자세한 내용은 단원을 참조하십시오.IAM 정책 생성의IAM 사용 설명서.

구성ProfileRefreshingAWSCredentialProviderAWS 자격 증명 새로 고침

하이브리드 환경에 AWS Systems Manager 사용하여 AWS 자격 증명을 관리하는 경우 시스템 관리자는`c:\Windows\System32\config\systemprofile\.aws\credentials`. 하이브리드 환경을 위한 Systems Manager 에 대한 자세한 내용은하이브리드 환경을 위한 AWS Systems Manager 설정의AWS Systems Manager 사용자 가이드.

AWS .net SDK는 새 자격 증명을 자동으로 선택하지 않으므로`ProfileRefreshingAWSCredentialProvider`플러그인을 사용하여 자격 증명을 새로 고칩니다.

다음을 수행할 수 있습니다.`CredentialRef`속성을 참조할 수 있는 모든 AWS 동기화 구성의`Credentials`정의를 참조하십시오.`CredentialType`속성이 로 설정되어 있습니다.`ProfileRefreshingAWSCredentialProvider`다음 예제에 표시된 대로 를 사용합니다.

```
{
    "Sinks": [{
        "Id": "myCloudWatchLogsSink",
        "SinkType": "CloudWatchLogs",
        "CredentialRef": "ssmcred",
        "Region": "us-west-2",
        "LogGroup": "myLogGroup",
        "LogStream": "myLogStream"
    }],
    "Credentials": [{
        "Id": "ssmcred",
        "CredentialType": "ProfileRefreshingAWSCredentialProvider",
        "Profile": "default",
        "FilePath": "%USERPROFILE%//.aws//credentials",
        "RefreshingInterval": 300
    }]
}
```

자격 증명 정의는 키-값 페어로 다음과 같은 특성으로 구성됩니다.

`Id`

싱크 정의를 사용하여 지정할 수있는 문자열을 정의`CredentialRef`을 클릭하여 이 자격 증명 구성을 참조합니다.

`CredentialType`

리터럴 문자열로 설정`ProfileRefreshingAWSCredentialProvider`.

`Profile`

선택 사항입니다. 기본값은 `default`입니다.

`FilePath`

선택 사항입니다. AWS 자격 증명 파일 경로를 지정합니다. 지정하지 않을 경우, `%USERPROFILE%/.aws/credentials`이 기본값입니다.

`RefreshingInterval`

선택 사항입니다. 자격 증명이 새로 고쳐지는 빈도 (초) 입니다. 지정하지 않을 경우, `300`이 기본값입니다.

싱크 장식 구성

싱크 선언에는 소스에서 수집한 레코드를 향상시키기 위해 다양한 AWS 서비스로 스트리밍할 추가 데이터를 지정하는 키-값 쌍이 선택적으로 포함될 수 있습니다.

`TextDecoration`

이 키-값 페어를 사용합니다.`Format`은 싱크 선언에 지정됩니다. 이 값은 변수 대체가 발생하는 특수 형식 문자열입니다. 예를 들어,`TextDecoration`의`"{ComputerName}:::{timestamp:yyyy-MM-dd HH:mm:ss}:::{_record}"`싱크대 용으로 제공됩니다. 원본이 텍스트를 포함하는 로그 레코드를 내보내는 경우`The system has resumed from sleep.`이고 해당 소스가 파이프를 통해 싱크에 연결되면`MyComputer1:::2017-10-26 06:14:22:::The system has resumed from sleep.`는 싱크 유형과 연결된 AWS 서비스로 스트리밍됩니다. 이`{_record}`변수는 원본에서 전달한 원본 텍스트 레코드를 참조합니다.

`ObjectDecoration`

이 키-값 페어를 사용합니다.`Format`레코드 직렬화 전에 추가 데이터를 추가하기 위해 싱크 선언에 지정됩니다. 예를 들어,`ObjectDecoration`의`"ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd HH:mm:ss}"`JSON을 지정하는 싱크에 대해 제공됩니다.`Format`. 싱크 유형과 연결된 AWS 서비스로 스트리밍된 결과 JSON에는 원본의 원본 데이터 외에 다음 키-값 쌍이 포함됩니다.

```
{
```

```
    ComputerName: "MyComputer2",
    DT: "2017-10-17 21:09:04"
}
```

ObjectDecoration 사용 예는 자습서: 윈도우용 Kinesis 에이전트를 사용하여 JSON 로그 파일을 Amazon S3 로 스트리밍 단원을 참조하십시오.

ObjectDecorationEx

에 비해 더 유연한 데이터 추출 및 형식을 허용하는 식을 지정합니다ObjectDecoration. 이 필드는 싱크의 형식이 될 때 사용할 수 있습니다json. 표현식 구문은 다음과 같습니다.

```
"ObjectDecorationEx":
 "attribute1={expression1};attribute2={expression2};attribute3={expression3}(;...)"
```

예를 들어 다음과 같습니다.ObjectDecorationEx속성을 사용합니다.

```
"ObjectDecorationEx":
 "host={env:ComputerName};message={upper(_record)};time={format(_timestamp,
 'yyyyMMdd')}"
```

리터럴 레코드를 변환합니다.

System log message

표현식에 의해 반환 된 값과 함께 다음과 같이 JSON 객체로:

```
{
    "host": "EC2AMAZ-1234",
    "message": "SYSTEM LOG MESSAGE",
    "time": "20210201"
}
```

표현식 수식에 대한 자세한 내용은 단원을 참조하십시오.표현식 쓰기 팁. 대부분의 경우ObjectDecoration선언은 타임 스탬프 변수를 제외하고 새로운 구문을 사용하여 작동해야 합니다. A{timestamp:yyyyMMdd}의 필드를 입력합니다.ObjectDecoration로 표현됩니다.{format(_timestamp,'yyyyMMdd')}inObjectDecorationEx.

TextDecorationEx

에 비해 더 유연한 데이터 추출 및 형식을 허용하는 식을 지정합니다TextDecoration다음 예제에 표시된 대로 을 사용합니다.

```
"TextDecorationEx": "Message '{lower(_record)}' at {format(_timestamp, 'yyyy-MM-
dd')}"
```

다음을 수행할 수 있습니다.`TextDecorationEx`를 사용하여 JSON 객체를 작성할 수 있습니다. 다음 예제와 같이 '@ {'를 사용하여 열린 중괄호를 이스케이프합니다.

```
"TextDecorationEx": "@{ \"var\": \"{upper($myvar1)}\" }"
```

싱크에 연결된 소스의 소스 유형이`DirectorySource`이면 싱크는 세 가지 추가 변수를 사용할 수 있습니다.

`_FilePath`

로그 파일의 전체 경로입니다.

`_FileName`

파일의 파일 이름 및 파일 이름 확장명입니다.

`_Position`

로그 파일에서 레코드가 있는 위치를 나타내는 정수입니다.

이러한 변수는 모든 레코드를 단일 스트림으로 스트리밍하는 싱크에 연결된 여러 파일에서 로그 레코드를 수집하는 원본을 사용할 때 유용합니다. 이러한 변수의 값을 스트리밍 레코드에 주입하면 데이터 파이프라인의 다운스트림 분석이 파일 및 각 파일 내의 위치별로 레코드를 정렬할 수 있습니다.

표현식 쓰기 팁

표현식은 다음 중 하나일 수 있습니다.

- 변수 표현식입니다.
- 상수 표현식 (예:`'hello'`,`1`,`1.21`,`null`,`true`,`false`.
- 다음 예제와 같이 함수를 호출하는 호출 표현식입니다.

```
regexp_extract('Info: MID 118667291 ICID 197973259 RID 0 To: <jd@acme.com>', 'To: (\\
\\S+)', 1)
```

특수 문자

특수 문자를 이스케이프하려면 두 개의 백 슬래시가 필요합니다.

Nesting

다음 예제와 같이 함수 호출은 중첩될 수 있습니다.

```
format(date(2018, 11, 28), 'MMddyyyy')
```

Variables

로컬, 메타, 글로벌: 변수의 세 가지 유형이 있습니다.

- 로컬 변수로 시작`$`와 같은`$message`. 이벤트 객체의 속성, 이벤트가 사전인 경우 항목 또는 이벤트가 JSON 객체 인 경우 속성을 해결하는 데 사용됩니다. 지역 변수에 공백이나 특수 문자가 포함 된 경우 다음과 같은 인용 된 지역 변수를 사용`$'date created'`.
- 메타 변수밑줄로 시작합니다 (`_`) 로 설정되며 이벤트의 메타 데이터를 확인하는 데 사용됩니다. 모든 이벤트 유형은 다음 메타 변수를 지원합니다.

  `_timestamp`

  이벤트의 타임스탬프입니다.

  `_record`

  이벤트의 원시 텍스트 표현입니다.

  로그 이벤트는 다음과 같은 추가 메타 변수를 지원합니다.

  `_filepath`

  `_filename`

  `_position`

  `_linenumber`

- 전역 변수는 환경 변수, EC2 인스턴스 메타데이터 또는 EC2Tag로 해석됩니다. 성능을 높이려면 접두사를 사용하여 검색 범위를 제한하는 것이 좋습니다.`{env:ComputerName}`,`{ec2:InstanceId}`, 및`{ec2tag:Name}`.

기본 제공 함수 함수

Windows용 Kinesis 에이전트는 다음과 같은 기본 제공 기능을 지원합니다. 인수 중 하나라도NULL이 함수는 처리하도록 설계되지 않았습니다.NULL,NULL객체가 반환됩니다.

```
//string functions
int length(string input)
string lower(string input)
string lpad(string input, int size, string padstring)
string ltrim(string input)
string rpad(string input, int size, string padstring)
string rtrim(string input)
string substr(string input, int start)
string substr(string input, int start, int length)
string trim(string input)
string upper(string str)

//regular expression functions
string regexp_extract(string input, string pattern)
string regexp_extract(string input, string pattern, int group)

//date functions
DateTime date(int year, int month, int day)
DateTime date(int year, int month, int day, int hour, int minute, int second)
DateTime date(int year, int month, int day, int hour, int minute, int second, int
 millisecond)

//conversion functions
int? parse_int(string input)
decimal? parse_decimal(string input)
DateTime? parse_date(string input, string format)
string format(object o, string format)

//coalesce functions
object coalesce(object obj1, object obj2)
object coalesce(object obj1, object obj2, object obj3)
object coalesce(object obj1, object obj2, object obj3, object obj4)
object coalesce(object obj1, object obj2, object obj3, object obj4, object obj5)
object coalesce(object obj1, object obj2, object obj3, object obj4, object obj5, object
 obj6)
```

싱크 변수 대체 구성

이KinesisStream,KinesisFirehose, 및CloudWatchLogs싱크 선언에는LogStream또는StreamName키-값 페어를 입력합니다. 이러한 키-값의 값에는 Windows용 Kinesis 에이전트에 의해 자동으로 해석되는 변수 참조가 포함될 수 있습니다. 용CloudWatchLogs,LogGroup키-값 쌍도 필요하며 Windows용 Kinesis 에이전트에 의해 자동으로 해석되는 변수 참조를 포함할 수 있습니다. 변수는 템플릿을 사용하여 지정됩니다{*prefix*:*variablename*}여기서 각 항목은 다음과 같습니다.*prefix*:는 선택 사항입니다. 지원되는 접두사는 다음과 같습니다.

- env— 변수 참조가 같은 이름의 환경 변수 값으로 해석됩니다.
- ec2— 변수 참조가 같은 이름의 EC2 인스턴스 메타데이터로 확인됩니다.
- ec2tag— 변수 참조가 같은 이름의 EC2 인스턴스 태그 값으로 확인됩니다. 이ec2:Describe*권한은 인스턴스 태그에 액세스하는 데 필요합니다. 자세한 내용은 EC2 태그 변수 확장에 필요한 권한 섹션을 참조하세요.

접두사를 지정하지 않으면 다음과 같은 이름의 환경 변수가있는 경우variablename이면 변수 참조가 환경 변수 값으로 해석됩니다. 그렇지 않을 경우 입니다.variablename확장하는 데instance_id또는hostname로 설정하면 변수 참조가 동일한 이름의 EC2 메타데이터 값으로 확인됩니다. 그렇지 않으면 변수 참조가 확인되지 않습니다.

다음은 변수 참조를 사용하는 유효한 키-값 페어의 예입니다.

```
"LogStream": "LogStream_{instance_id}"
"LogStream": "LogStream_{hostname}"
"LogStream": "LogStream_{ec2:local-hostname}"
"LogStream": "LogStream_{computername}"
"LogStream": "LogStream_{env:computername}"
```

이CloudWatchLogs싱크 선언은 원본의 원래 로그 또는 이벤트 레코드의 타임 스탬프가 로그 스트림의 이름을 변경할 수 있도록 허용하는 특수 형식 타임 스탬프 변수를 지원합니다. 형식은 {timestamp:timeformat}입니다. 다음 예를 참조하십시오.

```
"LogStream": "LogStream_{timestamp:yyyyMMdd}"
```

로그 또는 이벤트 레코드가 2017년 6월 5일에 생성된 경우LogStream이전 예제에서 키-값 페어는"LogStream_20170605".

권한이 있는 경우`CloudWatchLogs`싱크 유형은 생성 된 이름을 기반으로 필요한 경우 자동으로 새 로그 스트림을 만들 수 있습니다. 다른 싱크 유형에서는 스트림 이름 이외의 추가 구성이 필요하기 때문에 이 작업을 수행할 수 없습니다.

텍스트 및 개체 장식에서 발생하는 특수 변수 대체가 있습니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요.

싱크대 대기열 구성

이`KinesisStream`,`KinesisFirehose`, 및`CloudWatchLogs`싱크 선언은 일시적인 연결 문제로 인해 이러한 싱크 유형과 연결된 AWS 서비스로 스트리밍하지 못한 레코드의 대기열을 선택적으로 활성화할 수 있습니다. 연결이 복원될 때 큐 및 자동 스트리밍 재시도를 활성화하려면 싱크 선언에서 다음 키-값 쌍을 사용합니다.

`QueueType`

사용할 대기열 메커니즘의 종류를 지정합니다. 지원되는 유일한 값은`file`는 레코드가 파일에서 대기열에 있어야 함을 나타냅니다. 이 키-값 쌍은 Windows용 Kinesis 에이전트의 큐 기능을 활성화하기 위해 필요합니다. 지정되지 않은 경우 기본 동작은 메모리에만 대기하고 메모리 대기열 제한에 도달하면 스트리밍에 실패합니다.

`QueuePath`

대기 중인 레코드의 파일이 들어 있는 폴더 경로를 지정합니다. 이 키-값 페어는 선택 사항입니다. 기본값은 입니다.`%PROGRAMDATA%\KinesisTap\Queue\`신키드여기서 각 항목은 다음과 같습니다.신키드의 값으로 할당 된 식별자입니다`Id`싱크 선언에 대한.

`QueueMaxBatches`

스트리밍을 위해 레코드를 대기열에 넣을 때 Kinesis Windows용 에이전트가 사용할 수 있는 총 공간을 제한합니다. 공간의 양은 이 키-값 쌍의 값에 일괄 처리당 최대 바이트 수를 곱한 값으로 제한됩니다. 에 대 한 일괄 처리 당 최대 바이트`KinesisStream`,`KinesisFirehose`, 및`CloudWatchLogs`싱크 유형은 각각 5MB, 4MB 및 1MB입니다. 이 제한에 도달하면 스트리밍 실패가 대기열에 추가되지 않고 복구할 수 없는 실패로 보고됩니다. 이 키-값 페어는 선택 사항입니다. 기본값은 10,000개의 배치입니다.

싱크에 대한 프록시 구성

AWS 서비스에 액세스하는 모든 Windows용 Kinesis 에이전트 싱크 유형에 대한 프록시를 구성하려면`%Program Files%\Amazon\KinesisTap\AWSKinesisTap.exe.config`. 지침은 다음 () 을

참조하십시오.proxy의 섹션을 참조하십시오..NET용 AWS SDK에 대한 구성 파일 참조의.NET용 AWS SDK 개발자 안내서.

더 많은 싱크 속성에서 해석 변수 구성

다음 예제에서는 사용하는 싱크 구성을 보여 줍니다.Region환경 변수의 값에 대한Region속성 키-값 페어를. 용RoleARN, 그것은 EC2 태그 키를 지정MyRoleARN은 해당 키와 연관된 값으로 평가됩니다.

```
"Id": "myCloudWatchLogsSink",
"SinkType": "CloudWatchLogs",
"LogGroup": "EC2Logs",
"LogStream": "logs-{instance_id}"
"Region": "{env:Region}"
"RoleARN": "{ec2tag:MyRoleARN}"
```

AWS 싱크에서 RoleARN 속성을 사용할 때 AWS STS 리전 엔드포인트 구성

이 기능은 Amazon EC2 KinesisPap을 사용하고RoleARN속성을 사용하여 대상 AWS 서비스를 인증할 외부 IAM 역할을 가정할 수 있습니다.

설정함으로써UseSTSRegionalEndpointstotrue에서 에이전트가 지역 끝점을 사용하도록 지정할 수 있습니다 (예:https://sts.us-east-1.amazonaws.com) 대신 전역 끝점 (예:https://sts.amazonaws.com). 지역 STS 끝점을 사용하면 작업에 대한 왕복 대기 시간이 줄어들고 글로벌 끝점 서비스에서 실패의 영향이 제한됩니다.

AWS 싱크용 VPC 엔드포인트 구성

싱크 구성에서 VPC 엔드포인트를 지정할 수 있습니다.CloudWatchLogs,CloudWatch,KinesisStreams, 및KinesisFirehosesink 유형. VPC 엔드포인트를 이용하면 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 AWS Direct Connect 연결 없이도 AWS PrivateLink로 지원하는 지원되는 AWS 서비스 및 VPC 엔드포인트 서비스에 비공개로 연결할 수 있습니다. VPC의 인스턴스는 서비스의 리소스와 통신하는 데 퍼블릭 IP 주소를 필요로 하지 않습니다. VPC와 기타 서비스 간의 트래픽은 Amazon 네트워크를 벗어나지 않습니다. 자세한 내용은 단원을 참조하십시오.VPC 엔드포인트의Amazon VPC 사용 설명서.

VPC 엔드포인트는ServiceURL속성의 다음 예제에 표시된 대로CloudWatchLogssink 구성을 선택합니다. 값을 설정합니다.ServiceURL에 표시된 값으로VPC 엔드포인트 정보탭에서 Amazon VPC 콘솔을 사용합니다.

```
{
```

```
    "Id": "myCloudWatchLogsSink",
    "SinkType": "CloudWatchLogs",
    "LogGroup": "EC2Logs",
    "LogStream": "logs-{instance_id}",
    "ServiceURL":"https://vpce-ab1c234de56-ab7cdefg.logs.us-east-1.vpce.amazonaws.com"
}
```

대체 프록시 수단 구성

이 기능을 사용하면 .NET 대신 AWS SDK에 내장된 프록시 지원을 사용하여 싱크 구성으로 프록시 서버를 구성할 수 있습니다. 이전에는 프록시를 사용하도록 에이전트를 구성하는 유일한 방법은.NET의 기본 기능을 사용하는 것이었습니다. 이 기능은 프록시 파일에 정의된 프록시를 통해 모든 HTTP/S 요청을 자동으로 라우팅했습니다.

현재 프록시 서버에서 에이전트를 사용 중인 경우 이 방법을 사용하기 위해 오버를 변경할 필요가 없습니다.

다음을 수행할 수 있습니다.`ProxyHost`및`ProxyPort`속성을 사용하여 다음 예제와 같이 대체 프록시를 구성합니다.

```
{
    "Id": "myCloudWatchLogsSink",
    "SinkType": "CloudWatchLogs",
    "LogGroup": "EC2Logs",
    "LogStream": "logs-{instance_id}",
    "Region": "us-west-2",
    "ProxyHost": "myproxy.mydnsdomain.com",
    "ProxyPort": "8080"
}
```

파이프 선언

사용파이프 선언를 사용하여 소스를 연결합니다 (소스 선언) 를 싱크 (sink 선언) Microsoft Windows에 대한 Amazon Kinesis 에이전트에 로그인합니다. 파이프 선언은 JSON 객체로 표현됩니다. Windows용 Kinesis 에이전트가 시작되면 지정된 파이프에 대한 소스에서 로그, 이벤트 또는 메트릭이 수집됩니다. 그런 다음 해당 파이프와 연결된 싱크를 사용하여 다양한 AWS 서비스로 스트리밍됩니다.

다음은 파이프 선언의 예입니다.

```
{
```

```
    "Id": "MyAppLogToCloudWatchLogs",
    "SourceRef": "MyAppLog",
    "SinkRef": "MyCloudWatchLogsSink"
}
```

주제

- 파이프 구성
- Windows 메트릭 파이프용 Kinesis 에이전트 구성

파이프 구성

모든 파이프 선언에는 다음 키-값 페어가 포함될 수 있습니다.

`Id`

파이프의 이름을 지정합니다 (필수). 구성 파일 내에서 고유해야 합니다.

`Type`

로그 데이터가 소스에서 싱크로 전송될 때 파이프에 의해 적용되는 변환 유형 (있는 경우) 을 지정합니다. 지원되는 유일한 값은 `RegexFilterPipe`입니다. 이 값을 사용하면 로그 레코드의 기본 텍스트 표현에 대한 정규 표현식 필터링을 사용할 수 있습니다. 필터링을 사용하면 관련 로그 레코드만 데이터 파이프라인으로 전송하여 전송 및 저장 비용을 줄일 수 있습니다. 이 키-값 페어는 선택 사항입니다. 기본 값은 변환을 제공하지 않는 것입니다.

`FilterPattern`

에 대한 정규 표현식을 지정합니다`RegexFilterPipe`싱크로 전송되기 전에 소스에서 수집한 로그 레코드를 필터링하는 데 사용되는 파이프라인 로그 레코드는`RegexFilterPipe`유형 파이프는 정규 표현식이 레코드의 기본 텍스트 표현과 일치 할 때 사용됩니다. 생성된 구조화된 로그 레코드 (예:`ExtractionPattern`키-값 페어를 입력합니다.`DirectorySource`선언을 사용하여 필터링 할 수 있습니다`RegexFilterPipe`메커니즘에 저장됩니다. 이 메커니즘은 구문 분석하기 전에 원래 텍스트 표현에 대해 작동하기 때문입니다. 이 키 - 값 쌍은 선택 사항이지만 파이프가`RegexFilterPipe`유형의 값이 포함됩니다.

다음은 예제입니다.`RegexFilterPipe`파이프 선언:

```
{
 "Id": "MyAppLog2ToFirehose",
 "Type": "RegexFilterPipe",
```

```
 "SourceRef": "MyAppLog2",
 "SinkRef": "MyFirehose",
 "FilterPattern": "^(10|11),.*",
 "IgnoreCase": false,
 "Negate": false
}
```

`SourceRef`

이름 (의 값을 지정`Id`키-값 쌍) 을 통해 파이프에 대한 로그, 이벤트 및 메트릭 데이터를 수집하는 소스를 정의하는 소스 선언을 정의합니다 (필수).

`SinkRef`

이름 (의 값을 지정`Id`키 - 값 쌍) 파이프에 대한 로그, 이벤트 및 메트릭 데이터를 수신하는 싱크를 정의하는 싱크 선언의 (필수).

`IgnoreCase`

선택 사항입니다. 다음 값을 허용합니다.`true`또는`false`. 로 설정된 경우`true`로 설정하면 정규식은 대소 문자를 구분하지 않는 방식으로 레코드와 일치합니다.

`Negate`

선택 사항입니다. 다음 값을 허용합니다.`true`또는`false`. 로 설정된 경우`true`, 파이프는 레코드를 전달합니다안 함정규식 와 일치해야 합니다.

전체 구성 파일의 예는`RegexFilterPipe`파이프 유형에 대한 자세한 내용은파이프 사용.

Windows 메트릭 파이프용 Kinesis 에이전트 구성

라는 기본 제공 메트릭 소스가 있습니다.`_KinesisTapMetricsSource`에서 Windows용 Kinesis 에이전트에 대한 메트릭을 생성합니다. 있는 경우`CloudWatch`와 싱크 선언`Id`의`MyCloudWatchSink`의 경우 다음 예제 파이프라인 선언은 Windows용 Kinesis 에이전트가 생성한 메트릭을 해당 싱크로 전송합니다.

```
{
   "Id": "KinesisAgentMetricsToCloudWatch",
   "SourceRef": "_KinesisTapMetricsSource",
   "SinkRef": "MyCloudWatchSink"
}
```

Windows용 Kinesis 에이전트 기본 제공 메트릭 소스에 대한 자세한 내용은Windows용 Kinesis 에이전트 기본 제공 메트릭 소스.

구성 파일에서 Windows 성능 카운터 메트릭도 스트리밍하는 경우 Windows용 Kinesis 에이전트 메트릭 및 Windows 성능 카운터 메트릭 모두에 동일한 싱크를 사용하는 대신 별도의 파이프와 싱크를 사용하는 것이 좋습니다.

자동 업데이트 구성

사용`appsettings.json`구성 파일을 사용하여 Microsoft Windows용 Amazon Kinesis 에이전트와 Windows용 Kinesis 에이전트의 구성 파일을 자동으로 업데이트할 수 있습니다. 업데이트 동작을 제어하려면`Plugins`키 - 값 쌍은 다음과 같은 구성 파일의 동일한 수준에있다.`Sources`,`Sinks`, 및`Pipes`.

이`Plugins`키 - 값 쌍은 소스, 싱크 및 파이프의 범주에 특별히 해당하지 않는 사용할 추가 일반 기능을 지정합니다. 예를 들어 Windows용 Kinesis 에이전트를 업데이트하기 위한 플러그인이 있으며`appsettings.json`구성 파일을 입력합니다. 플러그인은 JSON 객체로 표현되며 항상`Type`키-값 페어를 입력합니다. 이`Type`플러그인에 대해 다른 키-값 페어를 지정할 수 있는지를 결정합니다. 지원되는 플러그인 유형은 다음과 같습니다.

`PackageUpdate`

Windows용 Kinesis 에이전트가 패키지 버전 구성 파일을 주기적으로 확인하도록 지정합니다. 패키지 버전 파일에 다른 버전의 Windows용 Kinesis 에이전트가 설치되어야 한다고 표시되면 Windows용 Kinesis 에이전트가 해당 버전을 다운로드하여 설치합니다. 이`PackageUpdate`플러그인 키-값 페어는 다음과 같습니다.

`Type`

값은 문자열 이어야 합니다.`PackageUpdate`이며 필수입니다.

`Interval`

패키지 버전 파일에서 문자열로 표시되는 변경 사항을 분 단위로 확인하는 빈도를 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 값을 지정하지 않으면 기본값은 60분입니다. 값이 1보다 작으면 업데이트 확인이 수행되지 않습니다.

`PackageVersion`

패키지 버전 JSON 파일의 위치를 지정합니다. 파일은 파일 공유에 있을 수 있습니다 (`file://`), 웹 사이트 (`http://`) 또는 Amazon S3 (`s3://`). 예를 들어 값`s3://mycompany/config/agent-package-version.json`는 Windows용 Kinesis 에이전트가`config/`

`agent-package-version.json`파일의`mycompany`Amazon S3 버킷입니다. 해당 파일의 내용을 기반으로 업데이트를 수행해야합니다.

> **Note**
>
> 의 값이`PackageVersion`키-값 쌍은 Amazon S3 경우 대/소문자를 구분합니다.

다음은 패키지 버전 파일의 내용을 보여주는 예제입니다.

```
{
    "Name": "AWSKinesisTap",
    "Version": "1.0.0.106",
    "PackageUrl": "https://s3-us-west-2.amazonaws.com/kinesis-agent-windows/
downloads/AWSKinesisTap.{Version}.nupkg"
}
```

이`Version`키-값 쌍은 아직 설치되지 않은 경우 설치할 Kinesis 에이전트의 버전을 지정합니다. 이`{Version}`변수 참조의`PackageUrl`는 사용자가 지정한 값을 해결합니다.`Version`키-값 페어를 입력합니다. 이 예제에서 변수는 문자열`1.0.0.106`. 이 변수 해상도는 패키지 버전 파일에 원하는 특정 버전이 저장되는 단일 위치가있을 수 있도록 제공됩니다. 여러 패키지 버전 파일을 사용하여 더 큰 배포 전에 새 버전의 유효성을 검사하기 위해 Windows용 Kinesis Agent의 새 버전을 롤아웃하는 속도를 제어할 수 있습니다. Windows용 Kinesis 에이전트 배포를 롤백하려면 하나 이상의 패키지 버전 파일을 변경하여 사용자 환경에서 작동하는 것으로 알려진 Windows용 Kinesis 에이전트의 이전 버전을 지정합니다.

의 값이`PackageVersion`키 - 값 쌍은 다른 패키지 버전 파일의 자동 선택을 용이하게하기 위해 변수 대체의 영향을받습니다. 변수 대체에 대한 자세한 내용은 단원을 참조하세요.싱크 변수 대체 구성.

`AccessKey`

Amazon S3 패키지 버전 파일에 대한 액세스를 인증할 때 사용할 액세스 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 키-값 페어를 사용하는 것은 권장하지 않습니다. 권장되는 대체 인증 방법은 인증 구성.

`SecretKey`

Amazon S3 패키지 버전 파일에 대한 액세스를 인증할 때 사용할 비밀 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 키-값 페어를 사용하는 것은 권장하지 않습니다. 권장되는 대체 인증 방법은 인증 구성.

`Region`

Amazon S3 에서 패키지 버전 파일에 액세스할 때 사용할 리전 엔드포인트를 지정합니다. 이 키-값 페어는 선택 사항입니다.

`ProfileName`

Amazon S3 패키지 버전 파일에 대한 액세스를 인증할 때 사용할 보안 프로필을 지정합니다. 자세한 내용은 인증 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

`RoleARN`

교차 계정 시나리오에서 Amazon S3 패키지 버전 파일에 대한 액세스를 인증할 때 위임할 역할을 지정합니다. 자세한 내용은 인증 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

그렇지 않은 경우`PackageUpdate`플러그인이 지정되면 업데이트가 필요한지 확인하기 위해 패키지 버전 파일이 검사되지 않습니다.

`ConfigUpdate`

Windows용 Kinesis 에이전트가 업데이트된`appsettings.json`구성 파일을 파일 공유, 웹 사이트 또는 Amazon S3. 업데이트된 구성 파일이 있는 경우 Windows용 Kinesis 에이전트에서 다운로드하여 설치합니다.`ConfigUpdate`키-값 페어는 다음과 같습니다.

`Type`

값은 문자열 이어야 합니다.`ConfigUpdate`이며 필수입니다.

`Interval`

새 구성 파일을 확인하는 빈도를 문자열로 표시되는 분 단위로 지정합니다. 이 키 - 값 쌍은 선택 사항이며 지정되지 않은 경우 기본값은 5 분입니다. 값이 1보다 작으면 구성 파일 업데이트가 확인되지 않습니다.

`Source`

업데이트된 구성 파일을 찾을 위치를 지정합니다. 파일은 파일 공유에 있을 수 있습니다 (`file://`), 웹 사이트 (`http://`) 또는 Amazon S3 (`s3://`). 예를 들어 값`s3://mycompany/config/appsettings.json`는 Windows용 Kinesis 에이전트가`config/appsettings.json`파일의`mycompany`Amazon S3 버킷입니다.

> **Note**
>
> 의 값이`Source`키-값 쌍은 Amazon S3 경우 대/소문자를 구분합니다.

의 값이Source키 - 값 쌍은 다른 구성 파일의 자동 선택을 용이하게하기 위해 변수 대체에 의해 영향을받습니다. 변수 대체에 대한 자세한 내용은 단원을 참조하세요.싱크 변수 대체 구성.

`Destination`

로컬 시스템에서 구성 파일을 저장할 위치를 지정합니다. 상대 경로, 절대 경로 또는 다음과 같은 환경 변수 참조가 포함 된 경로 일 수 있습니다.`%PROGRAMDATA%`. 경로가 상대 경로인 경우 Windows용 Kinesis 에이전트가 설치된 위치에 상대적입니다. 일반적으로 값은 다음과 같아야 합니다.`..\appsettings.json`. 이 키-값 페어가 필수입니다.

`AccessKey`

Amazon S3 구성 파일에 대한 액세스를 인증할 때 사용할 액세스 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 키-값 페어를 사용하는 것은 권장하지 않습니다. 권장되는 대체 인증 방법은 인증 구성.

`SecretKey`

Amazon S3 구성 파일에 대한 액세스를 인증할 때 사용할 비밀 키를 지정합니다. 이 키-값 페어는 선택 사항입니다. 이 키-값 페어를 사용하는 것은 권장하지 않습니다. 권장되는 대체 인증 방법은 인증 구성.

`Region`

Amazon S3 에서 구성 파일에 액세스할 때 사용할 리전 엔드포인트를 지정합니다. 이 키-값 페어는 선택 사항입니다.

`ProfileName`

Amazon S3 구성 파일에 대한 액세스를 인증할 때 사용할 보안 프로필을 지정합니다. 자세한 내용은 인증 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

`RoleARN`

교차 계정 시나리오에서 Amazon S3 구성 파일에 대한 액세스를 인증할 때 위임할 역할을 지정합니다. 자세한 내용은 인증 구성 섹션을 참조하세요. 이 키-값 페어는 선택 사항입니다.

그렇지 않은 경우`ConfigUpdate`플러그인이 지정되면 구성 파일 업데이트가 필요한지 여부를 확인하기 위해 구성 파일이 검사되지 않습니다.

다음은 예제입니다.`appsettings.json`사용 방법을 보여 주는 구성 파일`PackageUpdate`및`ConfigUpdate`플러그인입니다. 이 예제에서는 패키지 버전 파일이 있는`mycompany`이라는 Amazon S3 버킷의 이름이`config/agent-package-version.json`. 이 파

일은 약 2시간마다 변경 사항을 확인합니다. 다른 버전의 Windows용 Kinesis 에이전트가 패키지 버전 파일에 지정된 경우 지정된 에이전트 버전이 패키지 버전 파일의 지정된 위치에서 설치됩니다.

또한, 이`appsettings.json`구성 파일에 저장된`mycompany`이라는 Amazon S3 버킷의 이름 이`config/appsettings.json`. 약 30분마다 해당 파일이 현재 구성 파일과 비교됩니다. 서로 다른 경우 업데이트된 구성 파일이 Amazon S3 에서 다운로드되고`appsettings.json`구성 파일을 입력합니다.

```
{
  "Sources": [
    {
      "Id": "ApplicationLogSource",
      "SourceType": "DirectorySource",
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "RecordParser": "SingleLine"
    }
  ],
  "Sinks": [
    {
       "Id": "ApplicationLogKinesisFirehoseSink",
       "SinkType": "KinesisFirehose",
       "StreamName": "ApplicationLogFirehoseDeliveryStream",
       "Region": "us-east-1"
    }
    ],
  "Pipes": [
    {
      "Id": "ApplicationLogSourceToApplicationLogKinesisFirehoseSink",
      "SourceRef": "ApplicationLogSource",
      "SinkRef": "ApplicationLogKinesisFirehoseSink"
    }
  ],
  "Plugins": [
    {
      "Type": "PackageUpdate"
      "Interval": "120",
      "PackageVersion": "s3://mycompany/config/agent-package-version.json"
    },
    {
      "Type": "ConfigUpdate",
      "Interval": "30",
      "Source": "s3://mycompany/config/appsettings.json",
```

```
      "Destination": ".\appSettings.json"
    }
  ]
}
```

Windows용 Kinesis 에이전트 구성 예

이`appsettings.json`구성 파일은 Microsoft Windows용 Amazon Kinesis 에이전트가 로그, 이벤트 및 메트릭을 수집하는 방법을 제어하는 JSON 문서입니다. 또한 Windows용 Kinesis 에이전트가 해당 데이터를 변환하고 이를 다양한 AWS 서비스로 스트리밍하는 방법을 제어합니다. 구성 파일의 소스, 싱크 및 파이프 선언에 대한 자세한 내용은소스 선언,sink 선언, 및파이프 선언.

다음 섹션에는 여러 가지 시나리오에 대한 구성 파일의 예가 포함되어 있습니다.

주제

- 다양한 소스에서 Kinesis Data Streams 으로 스트리밍
- Windows 응용 프로그램 이벤트 로그에서 싱크로 스트리밍
- 파이프 사용
- 여러 소스 및 파이프 사용

다양한 소스에서 Kinesis Data Streams 으로 스트리밍

다음 예제는`appsettings.json`구성 파일은 다양한 소스에서 Kinesis Data Streams 으로, Windows 성능 카운터에서 Amazon CloudWatch 메트릭에 이르는 스트리밍 로그 및 이벤트를 보여 줍니다.

`DirectorySource`,**`SysLog`**레코드 구문 분석기

다음 파일은 모든 파일에서 syslog 형식 로그 레코드를`.log`파일 확장명입니다.`C:\LogSource\`디렉터리를`SyslogKinesisDataStream`us-east-1 리전에서 Kinesis 데이터 스트림 스트림입니다. 에이전트를 종료했다가 나중에 다시 시작하는 경우에도 로그 파일의 모든 데이터가 전송되도록 책갈피가 설정됩니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는`SyslogKinesisDataStream`Streams.

```
{
  "Sources": [
    {
      "Id": "SyslogDirectorySource",
      "SourceType": "DirectorySource",
```

```
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "RecordParser": "SysLog",
      "TimeZoneKind": "UTC",
      "InitialPosition": "Bookmark"
    }
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
      "SinkType": "KinesisStream",
      "StreamName": "SyslogKinesisDataStream",
      "Region": "us-east-1"
    }
  ],
  "Pipes": [
    {
      "Id": "SyslogDS2KSSink",
      "SourceRef": "SyslogDirectorySource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

`DirectorySource`,`SingleLineJson`레코드 구문 분석기

다음 파일은 모든 파일에서 JSON 형식의 로그 레코드를`.log`파일 확장명입니다.`C:\LogSource\`디렉터리를`JsonKinesisDataStream`us-east-1 리전에서 Kinesis 데이터 스트림 스트림입니다. 의 키-값 페어를 스트리밍하기 전에`ComputerName`및`DT`키는 컴퓨터 이름 및 레코드가 처리되는 날짜 및 시간과 함께 각 JSON 객체에 추가됩니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는`JsonKinesisDataStream`Streams.

```
{
  "Sources": [
    {
      "Id": "JsonLogSource",
      "SourceType": "DirectorySource",
      "RecordParser": "SingleLineJson",
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "InitialPosition": 0
    }
```

```
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
      "SinkType": "KinesisStream",
      "StreamName": "JsonKinesisDataStream",
      "Region": "us-east-1",
      "Format": "json",
      "ObjectDecoration": "ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd
 HH:mm:ss}"
    }
  ],
  "Pipes": [
    {
      "Id": "JsonLogSourceToKinesisStreamSink",
      "SourceRef": "JsonLogSource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

`ExchangeLogSource`

다음 파일은 Microsoft Exchange에 의해 생성되고`.log`확장명입니다.`C:\temp\ExchangeLog\`디렉터리를`ExchangeKinesisDataStream`JSON 형식의 us-east-1 리전에서 Kinesis 데이터 스트림입니다. Exchange 로그는 JSON 형식이 아니지만 Windows용 Kinesis 에이전트는 로그를 구문 분석하여 JSON으로 변환할 수 있습니다. 의 키-값 페어를 스트리밍하기 전에`ComputerName`및`DT`키는 컴퓨터 이름 및 레코드가 처리되는 날짜 및 시간 값을 포함하는 각 JSON 객체에 추가됩니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는`ExchangeKinesisDataStream`Streams.

```
{
  "Sources": [
    {
       "Id": "ExchangeSource",
       "SourceType": "ExchangeLogSource",
       "Directory": "C:\\temp\\ExchangeLog\",
       "FileNameFilter": "*.log"
    }
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
```

```
      "SinkType": "KinesisStream",
      "StreamName": "ExchangeKinesisDataStream",
      "Region": "us-east-1",
      "Format": "json",
      "ObjectDecoration": "ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd
 HH:mm:ss}"
    }
  ],
  "Pipes": [
    {
      "Id": "ExchangeSourceToKinesisStreamSink",
      "SourceRef": "ExchangeSource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

W3SVCLogSource

다음 파일은 해당 파일의 표준 위치에 저장된 Windows 로그 레코드에 대한 IIS (인터넷 정보 서비스)를`IISKinesisDataStreamus-east-1` 리전에서 Kinesis 데이터 스트림 스트림입니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는`IISKinesisDataStreamStreams`. IIS는 윈도우 용 웹 서버입니다.

```
{
  "Sources": [
    {
       "Id": "IISLogSource",
       "SourceType": "W3SVCLogSource",
       "Directory": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
       "FileNameFilter": "*.log"
    }
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
      "SinkType": "KinesisStream",
      "StreamName": "IISKinesisDataStream",
      "Region": "us-east-1"
    }
  ],
```

```
  "Pipes": [
    {
      "Id": "IISLogSourceToKinesisStreamSink",
      "SourceRef": "IISLogSource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

`WindowsEventLogSource`쿼리 포함

다음 파일 스트림 로그 이벤트 수준 Windows 시스템 이벤트 로그에서`Critical`또는`Error`(2 보다 작거나 같은) 을`SystemKinesisDataStream`JSON 형식의 us-east-1 리전에서 Kinesis 데이터 스트림입니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는`SystemKinesisDataStream`Streams.

```
{
  "Sources": [
    {
         "Id": "SystemLogSource",
         "SourceType": "WindowsEventLogSource",
         "LogName": "System",
         "Query": "*[System/Level<=2]"
    }
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
      "SinkType": "KinesisStream",
      "StreamName": "SystemKinesisDataStream",
      "Region": "us-east-1",
      "Format": "json"
    }
  ],
  "Pipes": [
    {
      "Id": "SLSourceToKSSink",
      "SourceRef": "SystemLogSource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

WindowsETWEventSource

다음 파일은 Microsoft CLR (공용 언어 런타임) 예외 및 보안 이벤트를ClrKinesisDataStreamJSON 형식의 us-east-1 리전에서 Kinesis 데이터 스트림입니다. 사용자 지정 응용 프로그램에서 레코드를 읽고 처리할 수 있는ClrKinesisDataStreamStreams.

```
{
  "Sources": [
    {
       "Id": "ClrETWEventSource",
       "SourceType": "WindowsETWEventSource",
       "ProviderName": "Microsoft-Windows-DotNETRuntime",
       "TraceLevel": "Verbose",
       "MatchAnyKeyword": "0x00008000, 0x00000400"
    }
  ],
  "Sinks": [
    {
      "Id": "KinesisStreamSink",
      "SinkType": "KinesisStream",
      "StreamName": "ClrKinesisDataStream",
      "Region": "us-east-1",
      "Format": "json"
    }
  ],
  "Pipes": [
    {
      "Id": "ETWSourceToKSSink",
      "SourceRef": "ClrETWEventSource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

WindowsPerformanceCounterSource

다음 파일은 열려있는 총 파일 수, 재부팅 이후 총 로그인 시도 횟수, 초당 디스크 읽기 수, us-east-1 리전의 CloudWatch 지표에 대한 사용 가능한 디스크 공간 비율에 대한 성능 카운터를 스트리밍합니다. CloudWatch 에서 이러한 지표를 그래프로 만들고, 그래프에서 대시보드를 만들고, 임계값이 초과되면 알림을 보내는 경보를 설정할 수 있습니다.

```
{
```

```
  "Sources": [
    {
      "Id": "PerformanceCounter",
      "SourceType": "WindowsPerformanceCounterSource",
      "Categories": [
        {
          "Category": "Server",
          "Counters": [
            "Files Open",
            "Logon Total"
          ]
        },
        {
          "Category": "LogicalDisk",
          "Instances": "*",
          "Counters": [
            "% Free Space",
            {
              "Counter": "Disk Reads/sec",
              "Unit": "Count/Second"
            }
          ]
        }
      ],
    }
  ],
  "Sinks": [
    {
      "Namespace": "MyServiceMetrics",
      "Region": "us-east-1",
      "Id": "CloudWatchSink",
      "SinkType": "CloudWatch"
    }
  ],
  "Pipes": [
    {
      "Id": "PerformanceCounterToCloudWatch",
      "SourceRef": "PerformanceCounter",
      "SinkRef": "CloudWatchSink"
    }
  ]
}
```

Windows 응용 프로그램 이벤트 로그에서 싱크로 스트리밍

다음 예제는`appsettings.json`구성 파일은 Windows 애플리케이션 이벤트 로그를 Microsoft Windows용 Amazon Kinesis 에이전트의 다양한 싱크로 스트리밍하는 것을 보여 줍니다. 사용의 예는`KinesisStream`및`CloudWatch`싱크 유형에 대한 자세한 내용은다양한 소스에서 Kinesis Data Streams 으로 스트리밍.

`KinesisFirehose`

다음 파일 스트림`Critical`또는`Error`Windows 응용 프로그램 로그 이벤트를`WindowsLogFirehoseDeliveryStream`us-east-1 리전에서 Kinesis Data Firehose 전송 스트림. Kinesis Data Firehose 에 대한 연결이 중단되면 이벤트는 먼저 메모리에 대기됩니다. 그런 다음 필요한 경우 연결이 복원될 때까지 디스크의 파일에 대기합니다. 그런 다음 이벤트는 대기열에 없어져 새 이벤트가 전송됩니다.

데이터 파이프라인 요구 사항에 따라 스트리밍된 데이터를 여러 종류의 스토리지 및 분석 서비스에 저장하도록 Kinesis Data Firehose를 구성할 수 있습니다.

```
{
  "Sources": [
    {
          "Id": "ApplicationLogSource",
          "SourceType": "WindowsEventLogSource",
          "LogName": "Application",
          "Query": "*[System/Level<=2]"
    }
  ],
  "Sinks": [
    {
       "Id": "WindowsLogKinesisFirehoseSink",
       "SinkType": "KinesisFirehose",
       "StreamName": "WindowsLogFirehoseDeliveryStream",
       "Region": "us-east-1",
       "QueueType": "file"
    }
    ],
  "Pipes": [
    {
      "Id": "ALSource2ALKFSink",
      "SourceRef": "ApplicationLogSource",
      "SinkRef": "WindowsLogKinesisFirehoseSink"
```

```
    }
  ]
}
```

CloudWatchLogs

다음 파일 스트림Critical또는Error클라우드 워치 로그 스트림에 대한 Windows 애플리케이션 로그 이벤트MyServiceApplicationLog-Group로그 그룹을 생성합니다. 각 스트림의 이름은Stream-. 스트림이 생성된 4자리 연도, 2자리 월 및 2자리 날짜로 끝납니다. (예:Stream-20180501는 2018 년 5 월 1 일에 생성 된 스트림입니다).

```
{
  "Sources": [
    {
         "Id": "ApplicationLogSource",
         "SourceType": "WindowsEventLogSource",
         "LogName": "Application",
         "Query": "*[System/Level<=2]"
    }
  ],
  "Sinks": [
    {
      "Id": "CloudWatchLogsSink",
      "SinkType": "CloudWatchLogs",
      "LogGroup": "MyServiceApplicationLog-Group",
      "LogStream": "Stream-{timestamp:yyyyMMdd}",
      "Region": "us-east-1",
      "Format": "json"
    }
  ],
  "Pipes": [
    {
      "Id": "ALSource2CWLSink",
      "SourceRef": "ApplicationLogSource",
      "SinkRef": "CloudWatchLogsSink"
    }
  ]
}
```

파이프 사용

다음 예제는appsettings.json구성 파일은 파이프 관련 기능을 사용하는 방법을 보여 줍니다.

이 예제에서는 로그 항목을`c:\LogSource\`에`ApplicationLogFirehoseDeliveryStream`Kinesis Data Firehose 전송 스트림. 그것은에 의해 지정된 정규 표현식과 일치하는 줄 만 포함`FilterPattern`키-값 페어의 형태로 표시됩니다. 특히 로그 파일의 줄만`10`또는`11`는 Kinesis Data Firehose 로 스트리밍됩니다.

```
{
  "Sources": [
    {
      "Id": "ApplicationLogSource",
      "SourceType": "DirectorySource",
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "RecordParser": "SingleLine"
    }
  ],
  "Sinks": [
    {
       "Id": "ApplicationLogKinesisFirehoseSink",
       "SinkType": "KinesisFirehose",
       "StreamName": "ApplicationLogFirehoseDeliveryStream",
       "Region": "us-east-1"
    }
    ],
  "Pipes": [
    {
      "Id": "ALSourceToALKFSink",
      "Type": "RegexFilterPipe",
      "SourceRef": "ApplicationLogSource",
      "SinkRef": "ApplicationLogKinesisFirehoseSink",
      "FilterPattern": "^(10|11),.*"
    }
  ]
}
```

여러 소스 및 파이프 사용

다음 예제는`appsettings.json`구성 파일은 여러 소스 및 파이프 사용을 보여줍니다.

이 예제에서는 응용 프로그램, 보안 및 시스템 Windows 이벤트 로그를`EventLogStream`Kinesis Data Firehose 전송 스트림은 세 개의 소스, 세 개의 파이프 및 단일 싱크를 사용합니다.

```
{
```

```
    "Sources": [
  {
    "Id": "ApplicationLog",
    "SourceType": "WindowsEventLogSource",
    "LogName": "Application"
  },
  {
    "Id": "SecurityLog",
    "SourceType": "WindowsEventLogSource",
    "LogName": "Security"
  },
  {
    "Id": "SystemLog",
    "SourceType": "WindowsEventLogSource",
    "LogName": "System"
  }
    ],
    "Sinks": [
  {
    "Id": "EventLogSink",
    "SinkType": "KinesisFirehose",
    "StreamName": "EventLogStream",
    "Format": "json"
  },
    ],
    "Pipes": [
  {
    "Id": "ApplicationLogToFirehose",
    "SourceRef": "ApplicationLog",
    "SinkRef": "EventLogSink"
  },
  {
    "Id": "SecurityLogToFirehose",
    "SourceRef": "SecurityLog",
    "SinkRef": "EventLogSink"
  },
  {
    "Id": "SystemLogToFirehose",
    "SourceRef": "SystemLog",
    "SinkRef": "EventLogSink"
  }
    ]
}
```

원격 분석 구성

더 나은 지원을 활성화하기 위해 기본적으로 Microsoft Windows용 Amazon Kinesis 에이전트는 에이전트 작동에 대한 통계를 수집하여 AWS 로 전송합니다. 이 정보에는 개인 식별 정보가 포함되어 있지 않습니다. AWS 서비스로 수집하거나 스트리밍하는 데이터는 포함되지 않습니다. 60분마다 약 1~2KB의 이 메트릭 데이터를 수집합니다.

이러한 통계 수집 및 전송을 옵트아웃할 수 있습니다. 이렇게 하려면 다음 키-값 페어를`appsettings.json`구성 파일을 소스, 싱크 및 파이프와 동일한 수준으로 설정합니다.

```
"Telemetrics":
    { "off": "true" }
```

예를 들어 다음 구성 파일은 소스, 싱크 및 파이프를 구성하고 원격 측정도 비활성화합니다.

```
{
  "Sources": [
    {
      "Id": "ApplicationLogSource",
      "SourceType": "DirectorySource",
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "RecordParser": "SingleLine"
    }
  ],
  "Sinks": [
    {
       "Id": "ApplicationLogKinesisFirehoseSink",
       "SinkType": "KinesisFirehose",
       "StreamName": "ApplicationLogFirehoseDeliveryStream",
       "Region": "us-east-1"
    }
    ],
  "Pipes": [
    {
      "Id": "ApplicationLogSourceToApplicationLogKinesisFirehoseSink",
      "SourceRef": "ApplicationLogSource",
      "SinkRef": "ApplicationLogKinesisFirehoseSink"
    }
  ],
```

```
  "Telemetrics":
    {
      "off": "true"
    }
}
```

원격 분석이 활성화된 경우 다음 메트릭을 수집합니다.

`ClientId`

소프트웨어가 설치될 때 자동으로 할당된 고유 ID입니다.

`ClientTimestamp`

원격 분석이 수집된 날짜 및 시간입니다.

`OSDescription`

운영 체제 설명.

`DotnetFramework`

현재 닷넷 프레임워크 버전입니다.

`MemoryUsage`

Windows용 Kinesis 에이전트에서 사용하는 메모리 양 (MB) 입니다.

`CPUUsage`

Windows용 Kinesis 에이전트 CPU 사용률의 양 (십진수) 입니다. 예를 들어, 0.01은 1% 를 의미합니다.

`InstanceId`

Windows용 Kinesis 에이전트가 Amazon EC2 인스턴스에서 실행 중인 경우 Amazon EC2 인스턴스 ID입니다.

`InstanceType (string)`

Windows용 Kinesis 에이전트가 Amazon EC2 인스턴스에서 실행 중인 경우 Amazon EC2 인스턴스 유형입니다.

또한, 우리는에 나열된 메트릭을 수집Windows용 Kinesis 에이전트 메트릭 목록.

자습서: 윈도우용 Kinesis 에이전트를 사용하여 JSON 로그 파일을 Amazon S3 로 스트리밍

이 자습서에서는 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 (Windows용 Kinesis 에이전트)를 사용하여 데이터 파이프라인을 설정하는 방법에 대한 자세한 단계를 제공합니다.

이 자습서에는 다음 단계가 포함됩니다.

- Windows용 Kinesis 에이전트를 사용하여 JSON 형식의 로그 파일을Amazon Simple Storage Service(Amazon S3)를 통해Amazon Kinesis Data Firehose. Windows용 Kinesis 에이전트에 대한 자세한 내용은마이크로소프트 윈도우용 Amazon Kinesis 에이전트란 무엇입니까?.
- 객체 장식을 사용하여 스트리밍하기 전에 로그 데이터를 향상시킵니다. 자세한 내용은 싱크 장식 구성 섹션을 참조하세요.
- 사용Amazon Athena를 사용하여 특정 종류의 로그 레코드를 검색합니다.

Prerequisites

AWS 계정이 아직 없는 경우AWS 계정 설정하나를 얻을 수 있습니다.

주제

- 단계 1: AWS 서비스 구성
- 단계 2: Windows용 Kinesis 에이전트 설치, 구성 및 실행
- 단계 3: Amazon S3 에서 로그 데이터 쿼리
- 다음 단계

단계 1: AWS 서비스 구성

다음 단계에 따라 Microsoft Windows용 Amazon Kinesis 에이전트를 사용하여 로그 데이터를 Simple Storage Service (Amazon S3) 로 스트리밍하기 위한 환경을 준비합니다. 자세한 내용과 사전 조건에 대한 자세한 내용은자습서: Amazon S3 로 JSON 로그 파일 스트리밍.

AWS 관리 콘솔을 사용하여 AWS Identity and Access Management (IAM), Amazon S3, Kinesis Data Firehose 및 Amazon Elastic Compute Cloud (Amazon EC2) 를 구성하여 EC2 인스턴스에서 Amazon S3 3로 로그 데이터를 스트리밍할 준비를 합니다.

주제

- IAM 정책 및 역할 구성
- Amazon S3 버킷을 생성합니다.
- Kinesis Data Firehose 전송 스트림 생성
- Windows용 Kinesis 에이전트를 실행하기 위한 Amazon EC2 인스턴스 생성
- 다음 단계

IAM 정책 및 역할 구성

Windows용 Kinesis 에이전트가 특정 KinKinesis Data Firehose 전송 스트림에 레코드를 스트리밍할 수 있도록 권한을 부여하는 다음 정책을 생성합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "firehose:PutRecord",
                "firehose:PutRecordBatch"
            ],
            "Resource": "arn:aws:firehose:region:account-id:deliverystream/log-
delivery-stream"
        }
    ]
}
```

Replace*region*을 Kinesis Data Firehose 전송 스트림이 생성될 AWS 리전의 이름으로 바꿉니다 (`us-east-1`(예:). Replace*account-id*을 전송 스트림이 생성될 AWS 계정에 대한 12자리 계정 ID로 바꿉니다.

탐색 모음에서 지원을 선택한 후 지원 센터를 선택합니다. 현재 로그인한 12자리 계정 번호 (ID) 는지원 센터탐색 창에서 를 선택합니다.

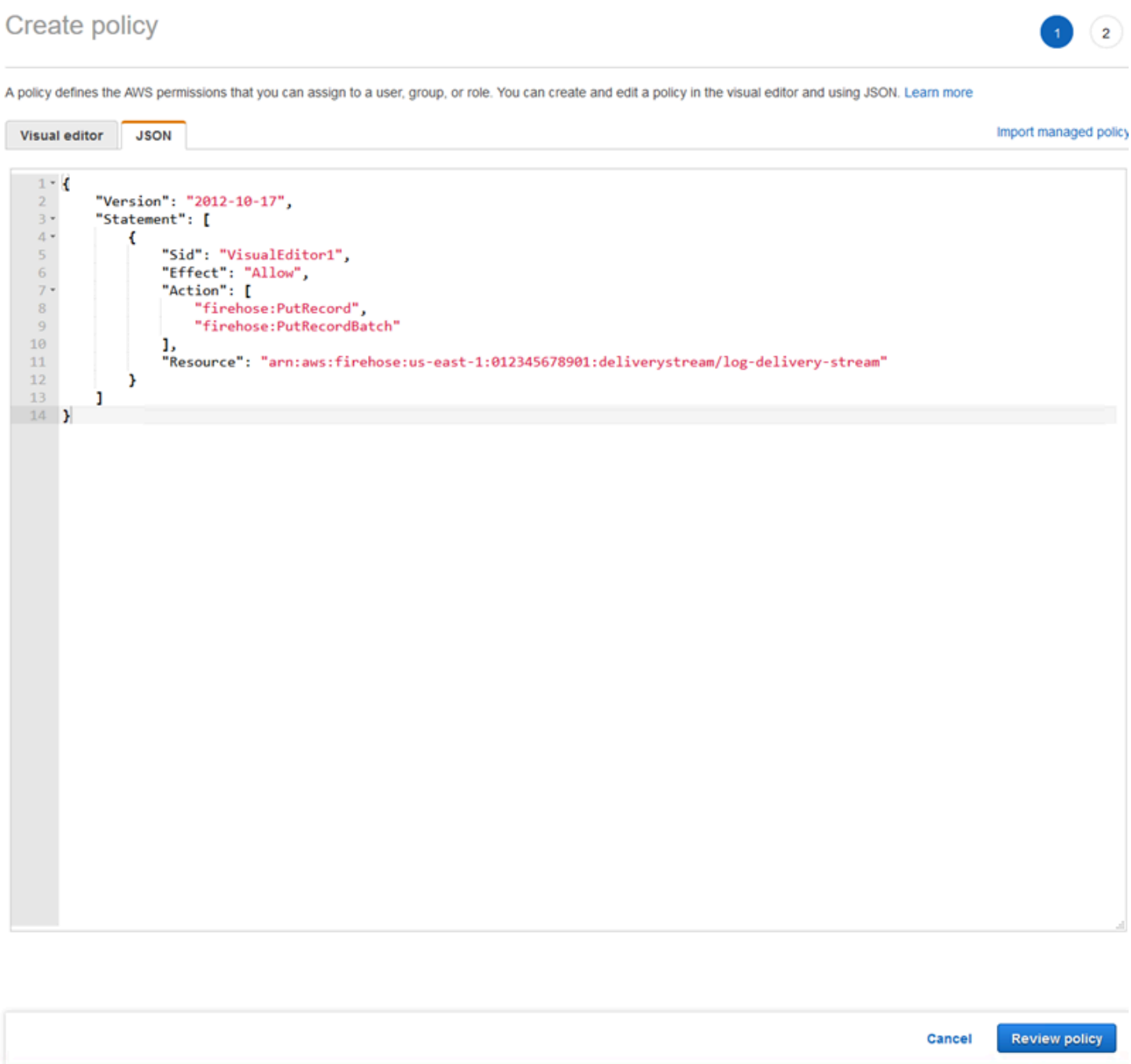
다음 절차를 사용하여 정책을 생성합니다. 정책 이름을 지정합니다.`log-delivery-stream-access-policy`.

JSON 정책 편집기를 사용하여 정책을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 왼쪽의 탐색 창에서정책.

   정책를 처음으로 선택하는 경우 Welcome to Managed Policies 페이지가 나타납니다. [Get Started]를 선택합니다.
3. 페이지 상단에서 정책 생성을 선택합니다.
4. JSON 탭을 선택합니다.
5. JSON 정책 문서를 입력합니다. IAM 정책 언어에 대한 자세한 내용은 섹션을 참조하십시오.IAM JSON 정책 참조의IAM 사용 설명서.
6. 작업이 완료되면 [Review policy]를 선택합니다. 정책 검사기가 모든 구문 오류를 보고합니다.

   > **Note**
   >
   > 언제든지 Visual editor(시각적 편집기) 및 JSON 탭을 전환할 수 있습니다. 그러나 변경 작업을 수행하거나정책 검토의Visual editor(시각적 편집기)탭에서 IAM은 시각적 편집기에 최적화되도록 정책을 재구성할 수 있습니다. 자세한 내용은 단원을 참조하십시오.정책 재구성의IAM 사용 설명서.
7. 에정책 검토페이지에서이름및설명(선택 사항) 을 선택합니다. 정책 요약을 검토하여 정책이 부여한 권한을 확인합니다. 그런 다음 [Create policy]를 선택하여 작업을 저장합니다.



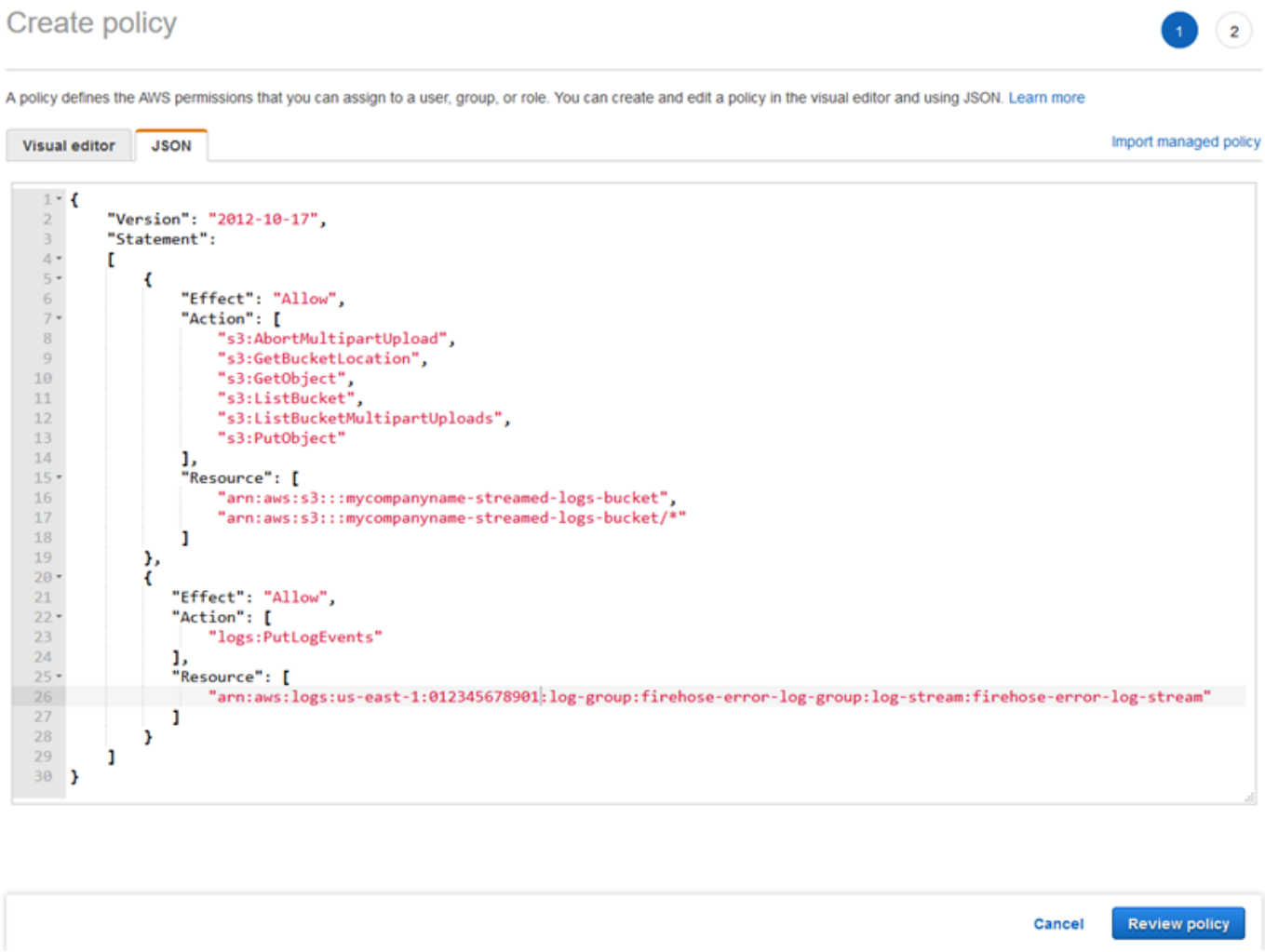
Kinesis Data Firehose 가 S3 버킷에 대한 액세스 권한을 부여하는 역할을 생성하려면 다음과 같이 하십시오.

1. 이전 절차를 사용하여`firehose-s3-access-policy`다음 JSON을 사용하여 정의 된:

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

```
    "Statement":
    [
        {
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
           "Effect": "Allow",
           "Action": [
               "logs:PutLogEvents"
           ],
           "Resource": [
               "arn:aws:logs:region:account-id:log-group:firehose-error-log-
group:log-stream:firehose-error-log-stream"
           ]
        }
    ]
}
```

Replace*bucket-name*를 로그가 저장되는 고유 버킷 이름으로 바꿉니다. Replace*region*를 CloudWatch 로그 로그 그룹 및 로그 스트림이 생성될 AWS 리전으로 연결합니다. 이는 Kinesis Data Firehose 를 통해 Amazon S3 로 데이터를 스트리밍하는 동안 발생하는 오류를 기록하기 위한 것입니다. Replace*account-id*를 로그 그룹 및 로그 스트림이 생성될 계정의 12자리 계정 ID 로 바꿉니다.



2. IAM 콘솔의 탐색 창에서 역할을 선택하고 역할 만들기를 선택합니다.
3. 선택을 선택합니다.AWS 서비스역할 유형을 선택한 후Kinesis서비스를 선택합니다.
4. 선택Kinesis Data Firehose사용 사례에 대해 선택한 후다음: 권한.
5. 검색 상자에 를 입력합니다.**`firehose-s3-access-policy`**를 선택하고 해당 정책을 선택한 다음다음: 검토.
6. 에서역할 이름상자에**`firehose-s3-access-role`**.
7. 역할 생성을 선택합니다.

Windows용 Kinesis 에이전트를 실행할 EC2 인스턴스의 인스턴스 프로필과 연결할 역할을 생성하려면

1. IAM 콘솔의 탐색 창에서 역할을 선택하고 역할 만들기를 선택합니다.
2. 선택을 선택합니다.AWS 서비스역할 유형을 선택한 후EC2.
3. 선택다음: 권한.

4. 검색 상자에 **log-delivery-stream-access-policy**를 입력합니다.
5. 정책을 선택한 후다음: 검토.
6. 에서역할 이름상자에**kinesis-agent-instance-role**.
7. 역할 생성을 선택합니다.

Amazon S3 버킷을 생성합니다.

Kinesis Data Firehose 에서 로그를 스트리밍하는 S3 버킷을 생성합니다.

로그 스토리지용 S3 버킷을 생성하려면

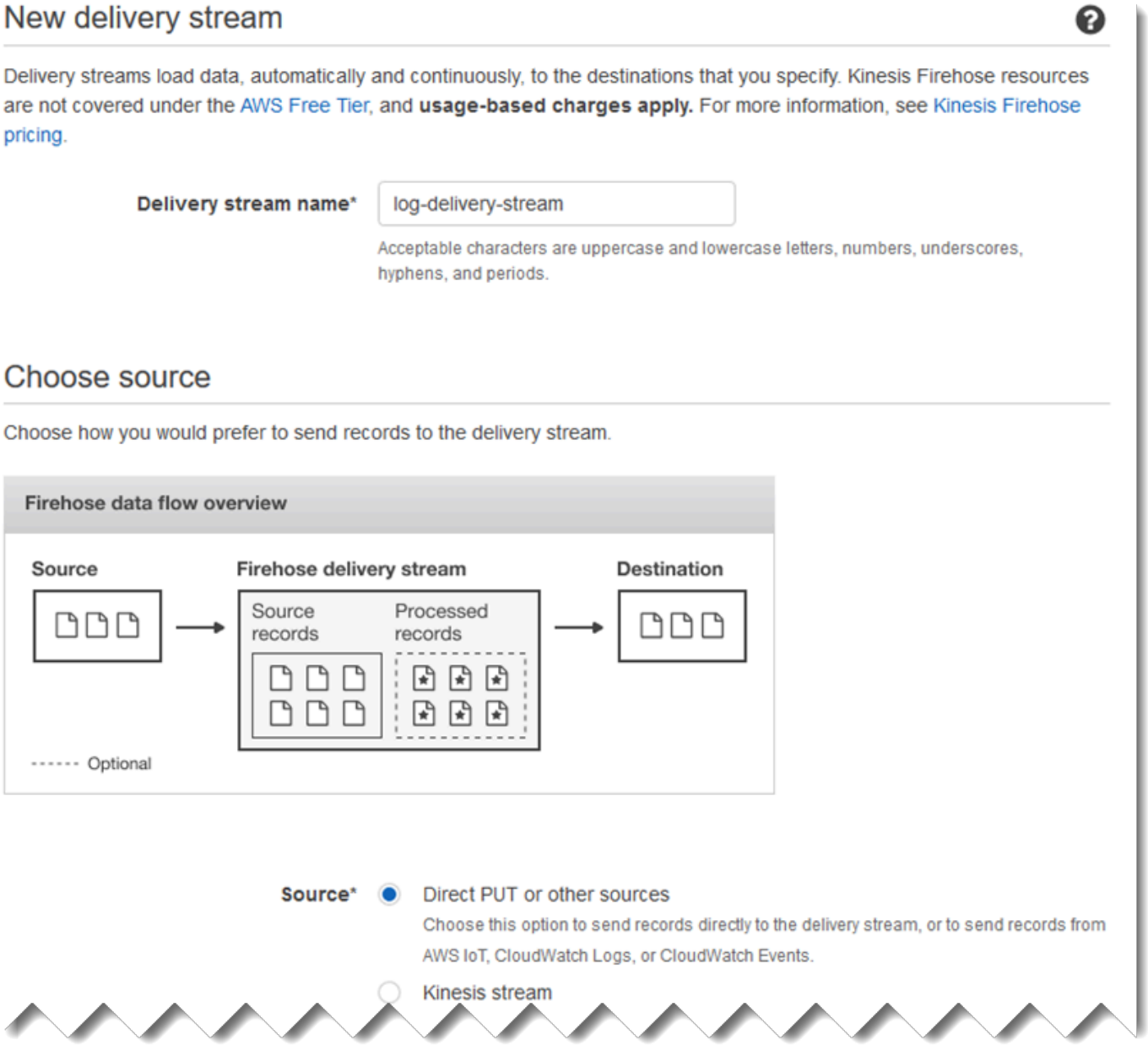
1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷 만들기를 선택합니다.
3. 에서Bucket name상자에 선택한 고유한 S3 버킷 이름을 입력합니다.IAM 정책 및 역할 구성.
4. 버킷을 생성할 리전을 선택합니다. 이것은 일반적으로 Kinesis Data Firehose 전송 스트림 및 Amazon EC2 인스턴스를 만들려는 리전과 동일합니다.
5. Create를 선택합니다.

Kinesis Data Firehose 전송 스트림 생성

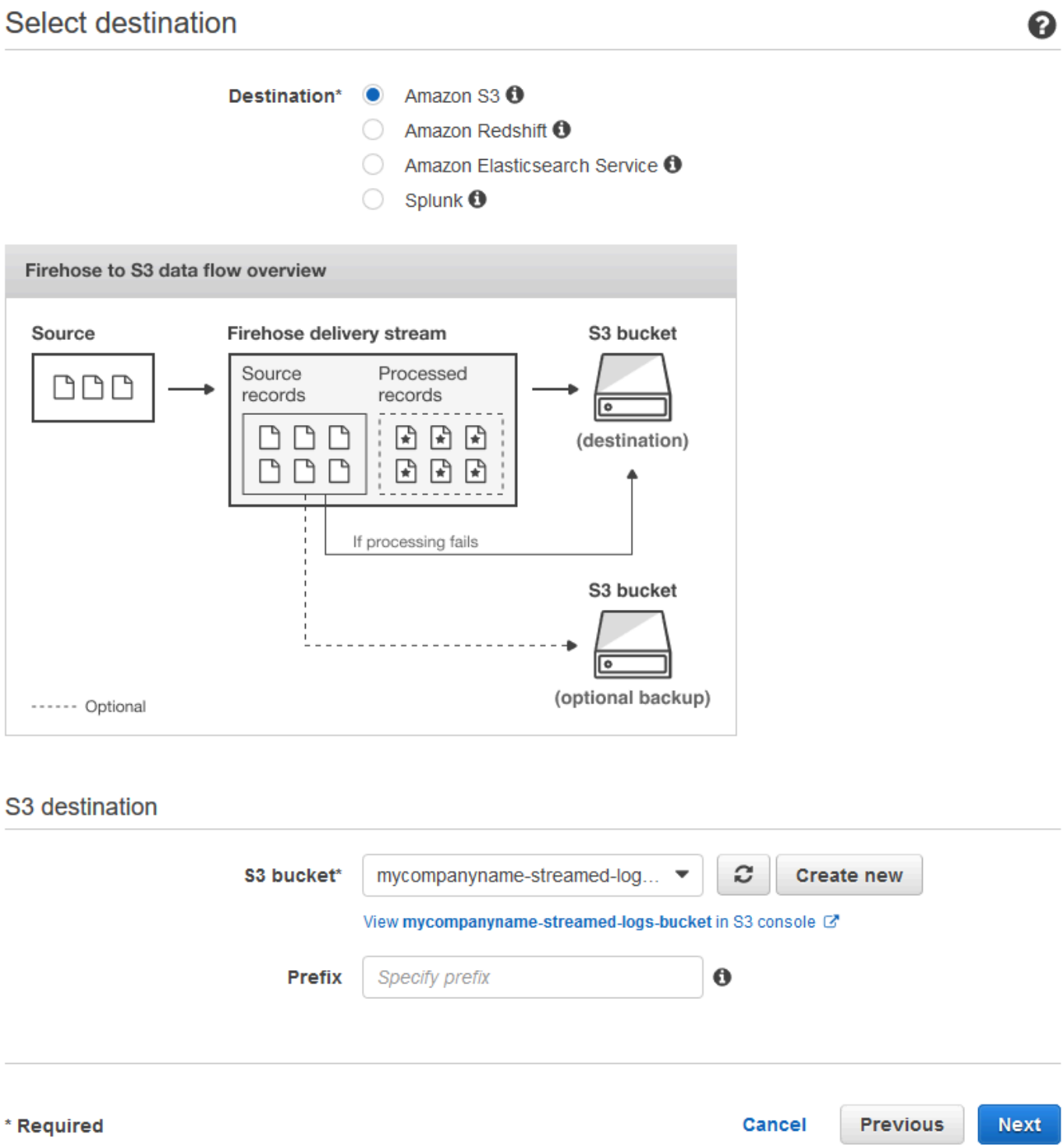
Amazon S3 스트리밍된 레코드를 저장할 Kinesis Data Firehose 전송 스트림을 생성합니다.

Kinesis Data Firehose 전송 스트림을 생성하려면

1. Kinesis Data Firehose 콘솔을https://console.aws.amazon.com/firehose/.
2. Create Delivery Stream을 선택합니다.
3. 에서Delivery stream name상자에**log-delivery-stream**.
4. 에 대 한소스를 선택하고직접 PUT 또는 기타 소스.



5. [Next]를 선택합니다.
6. 선택다음를 다시 선택합니다.
7. 목적지에 대해Amazon S3.
8. 에 대 한S3 버킷에서 생성한 버킷의 이름을 선택합니다.Amazon S3 버킷을 생성합니다..



9. [Next]를 선택합니다.
10. 에서버퍼 간격상자에**60**.
11. 언더IAM 역할를 선택하고새로 생성하거나.
12. 용IAM 역할를 선택하고`firehose-s3-access-role`.

13. [Allow]를 선택합니다.

Configure settings

Configure buffer, compression, logging, and IAM role settings for your delivery stream.

S3 buffer conditions

Firehose buffers incoming records before delivering them to your S3 bucket. Record delivery will be triggered once either of these conditions has been satisfied. Learn more

Buffer size* 5 MB

Specify a buffer size between 1-128 MB

Buffer interval* 60 seconds

Specify a buffer interval between 60-900 seconds

S3 compression and encryption

Firehose can compress records before delivering them to your S3 bucket. Compressed records can also be encrypted in the S3 bucket using a KMS master key. Learn more

S3 compression*
- (●) Disabled
- ( ) GZIP
- ( ) Snappy
- ( ) Zip

S3 encryption*
- (●) Disabled
- ( ) Enabled

Error logging

Firehose can log record delivery errors to CloudWatch Logs. If enabled, a CloudWatch log group and corresponding log streams are created on your behalf. Learn more

Error logging*
- ( ) Disabled
- (●) Enabled

IAM role

Firehose uses an IAM role to access your specified resources, such as the S3 bucket and KMS key. Learn more

IAM role* firehose-s3-access-role

Create new or choose

14. [Next]를 선택합니다.
15. Create delivery stream(전송 스트림 생성)을 선택합니다.

Windows용 Kinesis 에이전트를 실행하기 위한 Amazon EC2 인스턴스 생성

Kinesis 데이터 파이어호스를 통해 로그 레코드를 스트리밍하기 위해 Windows용 Kinesis 에이전트를 사용하는 EC2 인스턴스를 생성합니다.

EC2 인스턴스를 생성하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 의 지침을 따르십시오.Amazon EC2 Windows 인스턴스 시작하기다음 추가 단계를 사용하십시오.
   - 에 대 한IAM 역할를 선택하고`kinesis-agent-instance-role`.
   - 퍼블릭 인터넷에 연결된 VPC (Virtual Private Cloud) 가 없는 경우Amazon EC2 사용한 설정의Windows 인스턴스용 Amazon EC2 사용 설명서.
   - 컴퓨터 또는 조직의 컴퓨터에서만 인스턴스에 대한 액세스를 제한하는 보안 그룹을 만들거나 사용합니다. 자세한 내용은 단원을 참조하십시오.Amazon EC2 사용한 설정의Windows 인스턴스용 Amazon EC2 사용 설명서.
   - 기존 key pair 지정하는 경우 키 페어의 개인 키에 액세스할 수 있어야 합니다. 또는 새 key pair 생성하고 프라이빗 키를 안전한 곳에 저장합니다.
   - 계속하기 전에 인스턴스가 실행 중이고 두 개의 상태 확인 중 두 개가 완료될 때까지 기다립니다.
   - 인스턴스에는 퍼블릭 IP 주소가 필요합니다. 할당되지 않은 경우탄력적 IP 주소의Windows 인스턴스용 Amazon EC2 사용 설명서.

다음 단계

단계 2: Windows용 Kinesis 에이전트 설치, 구성 및 실행

단계 2: Windows용 Kinesis 에이전트 설치, 구성 및 실행

이 단계에서는 AWS 관리 콘솔을 사용하여Windows용 Kinesis 에이전트를 실행하기 위한 Amazon EC2 인스턴스 생성. 그런 다음 인스턴스에 마이크로소프트 윈도우용 Amazon Kinesis 에이전트를 설치하고, Windows용 Kinesis 에이전트용 구성 파일을 생성 및 배포한 다음AW스키네시스탭서비스를 선택합니다.

1. Remore 데스크톱 프로토콜 (RDP) 을 통해 인스턴스에 원격으로 연결합니다.단계 2: 인스턴스에 연결의Windows 인스턴스용 Amazon EC2 사용 설명서.
2. 인스턴스에서 Windows 서버 관리자를 사용하여 사용자 및 관리자에 대한 Microsoft Internet Explorer 보안 강화 구성을 사용하지 않도록 설정합니다. 자세한 내용은 단원을 참조하십시오.Internet Explorer 보안 강화 구성을 해제하는 방법마이크로소프트 TechNet 웹 사이트를 참조하십시오.
3. 인스턴스에서 Windows용 Kinesis 에이전트를 설치하고 구성합니다. 자세한 내용은 윈도우용 Kinesis 에이전트 설치 섹션을 참조하세요.
4. 인스턴스에서 메모장을 사용하여 Windows용 Kinesis 에이전트 구성 파일을 만듭니다. 파일을 다음 위치에 저장합니다.`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`. 다음 콘텐츠를 구성 파일에 추가합니다.

```
{
  "Sources": [
    {
      "Id": "JsonLogSource",
      "SourceType": "DirectorySource",
      "RecordParser": "SingleLineJson",
      "Directory": "C:\\LogSource\\",
      "FileNameFilter": "*.log",
      "InitialPosition": 0
    }
  ],
  "Sinks": [
    {
      "Id": "FirehoseLogStream",
      "SinkType": "KinesisFirehose",
      "StreamName": "log-delivery-stream",
      "Region": "us-east-1",
      "Format": "json",
      "ObjectDecoration": "ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd
 HH:mm:ss}"
    }
  ],
  "Pipes": [
    {
      "Id": "JsonLogSourceToFirehoseLogStream",
      "SourceRef": "JsonLogSource",
      "SinkRef": "FirehoseLogStream"
    }
  ]
```

```
}
```

이 파일은 Windows용 Kinesis 에이전트가`c:\logsource\`디렉토리 (source) 을 Kinesis Data Firehose 전송 스트림에 추가합니다.`log-delivery-stream`(sink). 각 로그 레코드가 Kinesis Data Firehose 로 스트리밍되기 전에 컴퓨터 이름과 타임스탬프가 포함된 두 개의 추가 키-값 쌍으로 향상되었습니다.

5. 생성`c:\LogSource\`디렉터리를 만들고 메모장을 사용하여`test.log`파일을 해당 디렉터리에 추가합니다.

```
{ "Message": "Copasetic message 1", "Severity": "Information" }
{ "Message": "Copasetic message 2", "Severity": "Information" }
{ "Message": "Problem message 2", "Severity": "Error" }
{ "Message": "Copasetic message 3", "Severity": "Information" }
```

6. 상승된 PowerShell 세션에서 다음 명령을 사용하여AW스키네시스탭서비스:

```
Start-Service -ServiceName AWSKinesisTap
```

7. 파일 탐색기를 사용하여`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`디렉토리에 로그인합니다. 최신 로그 파일을 엽니다. 로그 파일은 다음과 같아야 합니다.

```
2018-09-28 23:51:02.2472 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.AWS.AWSEventSinkFactory.
2018-09-28 23:51:02.2784 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Windows.PerformanceCounterSinkFactory.
2018-09-28 23:51:02.5753 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Core.DirectorySourceFactory.
2018-09-28 23:51:02.5909 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.ExchangeSource.ExchangeSourceFactory.
2018-09-28 23:51:02.5909 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Uls.UlsSourceFactory.
2018-09-28 23:51:02.5909 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Windows.WindowsSourceFactory.
2018-09-28 23:51:02.9347 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Core.Pipes.PipeFactory.
2018-09-28 23:51:03.5128 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.AutoUpdate.AutoUpdateFactory.
2018-09-28 23:51:03.5440 Amazon.KinesisTap.Hosting.LogManager INFO Performance
 counter sink  started.
```

```
2018-09-28 23:51:03.7628 Amazon.KinesisTap.Hosting.LogManager INFO
 KinesisFirehoseSink id FirehoseLogStream for StreamName log-delivery-stream
 started.
2018-09-28 23:51:03.7784 Amazon.KinesisTap.Hosting.LogManager INFO Connected source
 JsonLogSource to sink FirehoseLogStream
2018-09-28 23:51:03.7940 Amazon.KinesisTap.Hosting.LogManager INFO DirectorySource
 id JsonLogSource watching directory C:\LogSource\ with filter *.log started.
```

이 로그 파일은 서비스가 시작되었으며 로그 레코드가 현재`c:\LogSource\`디렉토리에 로그인합니다. 각 행은 단일 JSON 객체로 구문 분석됩니다. 컴퓨터 이름 및 타임스탬프에 대한 키-값 쌍이 각 개체에 추가됩니다. 그런 다음 Kinesis Data Firehose 로 스트리밍됩니다.

8. 1 ~ 2 분 안에 생성한 Amazon S3 버킷으로 이동합니다.Amazon S3 버킷을 생성합니다.AWS Management Console을 사용합니다. 본체에서 올바른 리전을 선택했는지 확인합니다.

   이 버킷에는 현재 연도의 폴더가 있습니다. 해당 폴더를 열어 당월의 폴더를 표시합니다. 해당 폴더를 열어 현재 날짜의 폴더를 표시합니다. 해당 폴더를 열어 현재 시간 (UTC) 에 대한 폴더를 표시합니다. 해당 폴더를 열어 이름으로 시작하는 하나 이상의 항목을 표시합니다.`log-delivery-stream`.

9. 최신 항목의 내용을 열어 로그 레코드가 Amazon S3 성공적으로 저장되었는지 확인하고 원하는 향상된 기능을 제공합니다. 모든 것이 올바르게 구성되면 콘텐츠는 다음과 비슷합니다.

```
{"Message":"Copasetic message 1","Severity":"Information","ComputerName":"EC2AMAZ-
ABCDEFGH","DT":"2018-09-28 23:51:04"}
{"Message":"Copasetic message 2","Severity":"Information","ComputerName":"EC2AMAZ-
ABCDEFGH","DT":"2018-09-28 23:51:04"}
{"Message":"Problem message 2","Severity":"Error","ComputerName":"EC2AMAZ-
ABCDEFGH","DT":"2018-09-28 23:51:04"}
{"Message":"Copasetic message 3","Severity":"Information","ComputerName":"EC2AMAZ-
ABCDEFGH","DT":"2018-09-28 23:51:04"}
```

10. 다음 문제 중 하나를 해결하는 방법에 대한 자세한 내용은마이크로소프트 윈도우용 Amazon Kinesis 에이전트 문제 해결:
    - Windows용 Kinesis 에이전트 로그 파일에 오류가 있습니다.
    - Amazon S3 필요한 폴더 또는 항목이 없습니다.
    - Amazon S3 항목의 내용이 올바르지 않습니다.

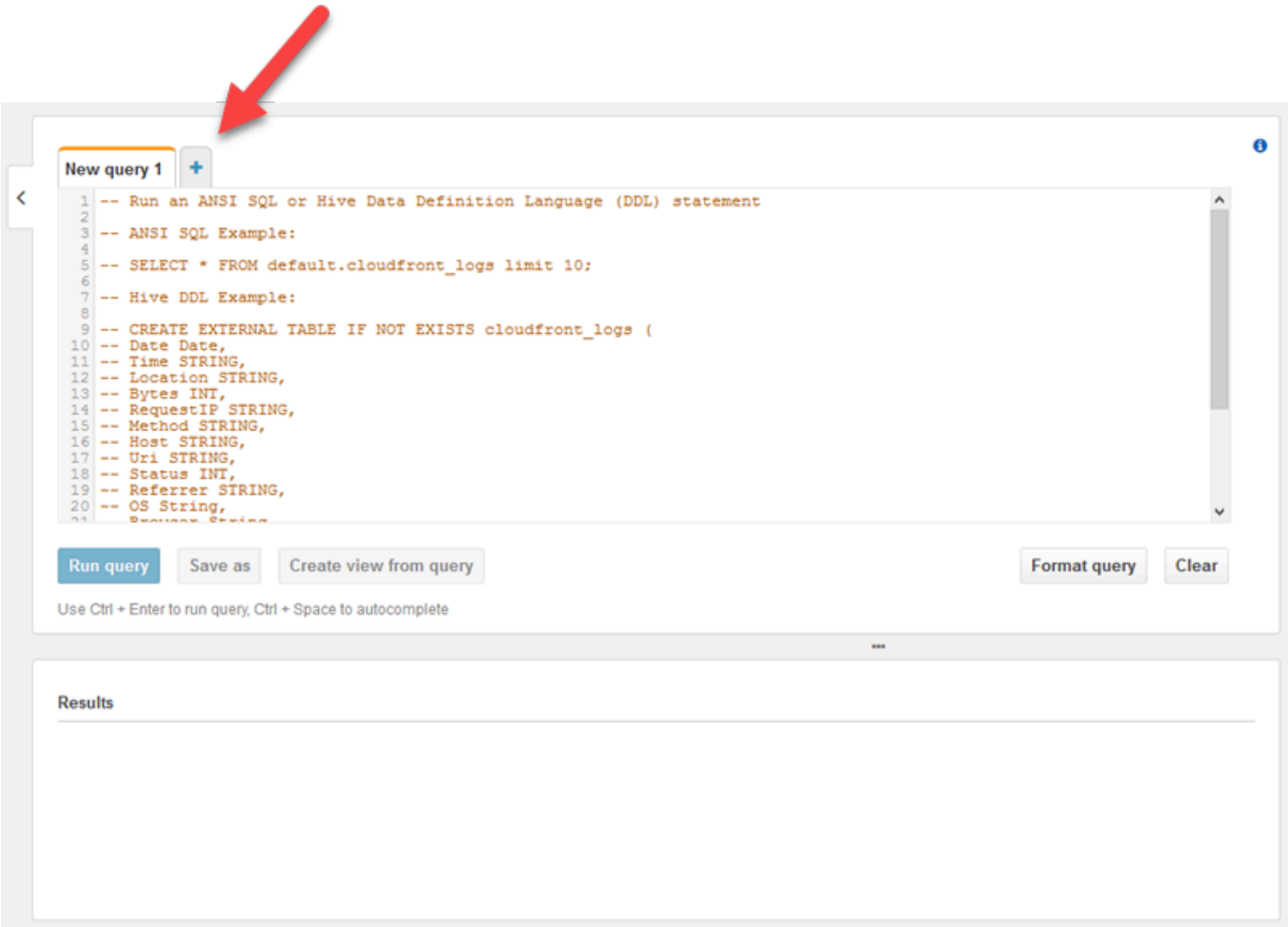
다음 단계

단계 3: Amazon S3 에서 로그 데이터 쿼리

단계 3: Amazon S3 에서 로그 데이터 쿼리

마이크로소프트 윈도우용 Amazon Kinesis 에이전트의 마지막 단계에서자습서Amazon Athena를 사용하여 Amazon Simple Storage Service (Amazon S3) 에 저장된 로그 데이터를 쿼리합니다.

1. https://console.aws.amazon.com/athena/에서 Athena 콘솔을 엽니다.
2. 더하기 기호를 선택합니다 (+) 를 사용하여 새 쿼리 창을 만들 수 있습니다.

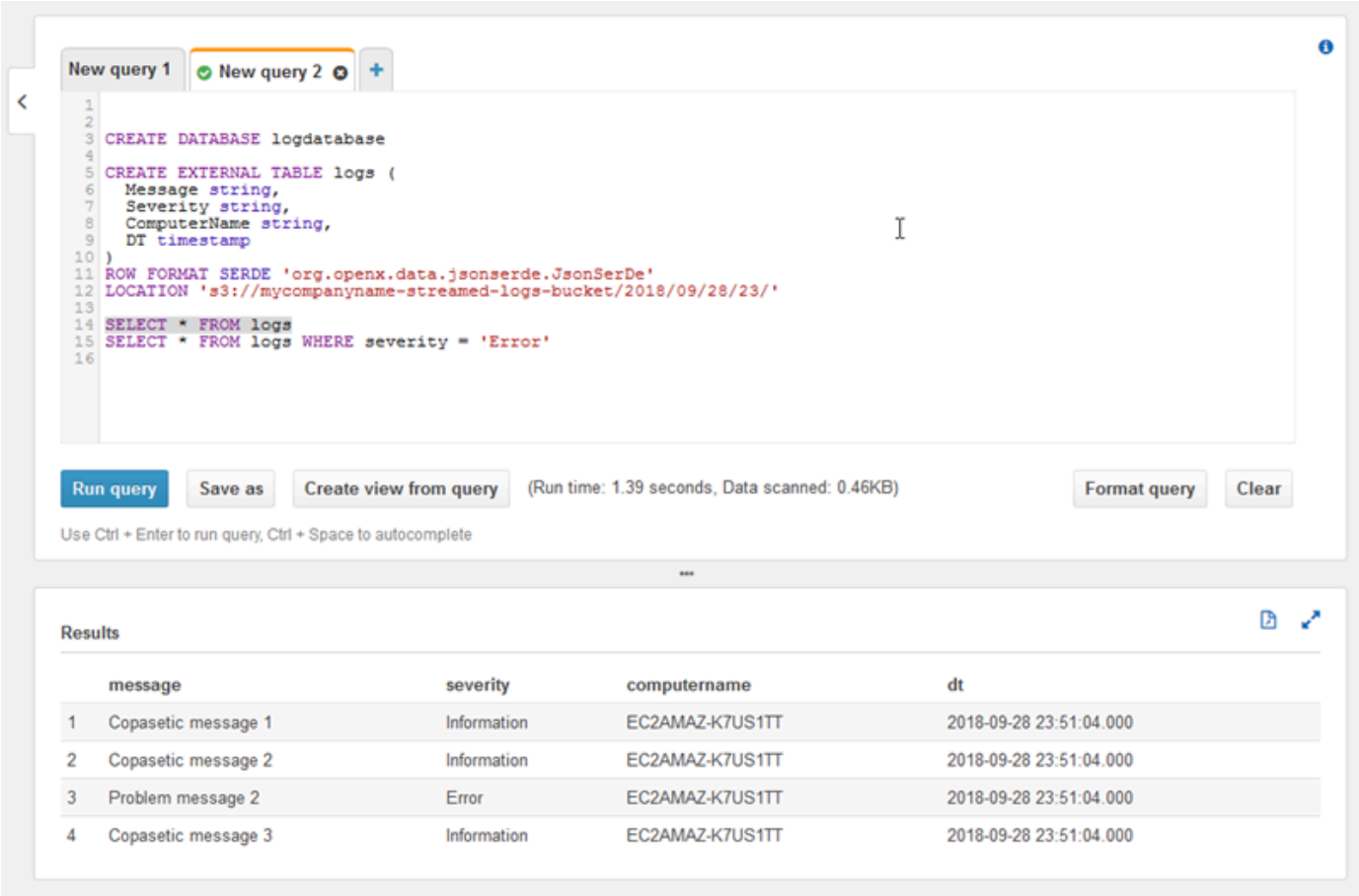


3. 다음 텍스트를 질의 창에 입력합니다.

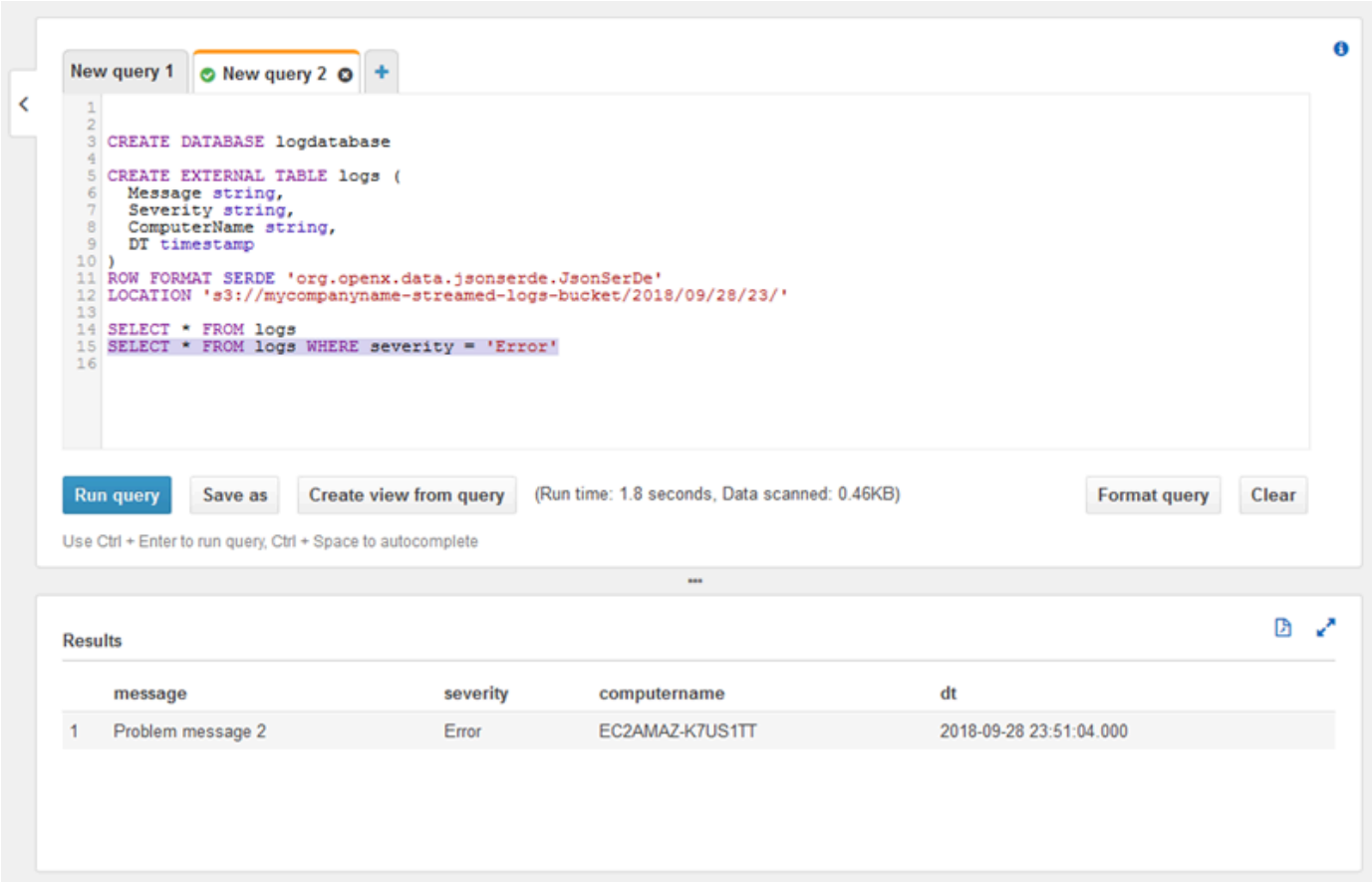
```
CREATE DATABASE logdatabase

CREATE EXTERNAL TABLE logs (
  Message string,
  Severity string,
  ComputerName string,
  DT timestamp
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://bucket/year/month/day/hour/'

SELECT * FROM logs
SELECT * FROM logs WHERE severity = 'Error'
```

Replace*bucket*에서 만든 버킷의 이름으로 바꿉니다.Amazon S3 버킷을 생성합니다..
Replace*year*,*month*,*day*및*hour*에 Amazon S3 로그 파일이 UTC로 생성된 연도, 월, 일 및 시간을 입력합니다.

4. 에 대 한 텍스트를 선택 합니다.`CREATE DATABASE`문을 마우스 오른쪽 단추로 클릭한 다음쿼리 실행. 이것은 Athena 나에 로그 데이터베이스를 만듭니다.
5. 에 대 한 텍스트를 선택 합니다.`CREATE EXTERNAL TABLE`문을 마우스 오른쪽 단추로 클릭한 다음쿼리 실행. 이렇게 하면 로그 데이터와 함께 S3 버킷을 참조하는 Athena 테이블이 생성되어 JSON의 스키마를 Athena 테이블의 스키마에 매핑합니다.
6. 첫 번째 텍스트 선택`SELECT`문을 마우스 오른쪽 단추로 클릭한 다음쿼리 실행. 이 테이블의 모든 행을 표시합니다.



7. 두 번째 텍스트 선택`SELECT`문을 마우스 오른쪽 단추로 클릭한 다음쿼리 실행. 이 로그 레코드를 나타내는 테이블의 행만 표시`Error`수준 심각도입니다. 이러한 종류의 쿼리는 잠재적으로 큰 로그 레코드 집합에서 흥미로운 로그 레코드를 찾습니다.



다음 단계

AWS 관리 콘솔을 사용하여 자습서 중에 생성된 리소스를 정리합니다.

1. EC2 인스턴스를 종료합니다 (Amazon EC2 Windows 인스턴스 시작하기).

> **Important**
>
> 인스턴스 내에 있지 않은 인스턴스를 시작한 경우AWS 프리 티어을 종료하려면 인스턴스를 종료하기 전까지 인스턴스에 대한 요금이 청구됩니다.

2. Kinesis Data Firehose 전송 스트림을 삭제합니다.

   a. Kinesis Data Firehose 콘솔을https://console.aws.amazon.com/firehose/.
   b. 생성한 전송 스트림을 선택합니다.
   c. 삭제를 선택합니다.
   d. 선택전송 스트림 삭제.

3. S3 버킷을 삭제합니다. 지침은 을 참조하십시오.S3 버킷을 삭제하려면 어떻게 해야 합니까?의Amazon Simple Storage Service 콘솔 사용자 안내서.

자세한 내용은 다음 주제를 참조하십시오.

- 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성
- Amazon Kinesis Data Firehose 란 무엇인가요?
- Amazon S3 이란 무엇인가요?
- Amazon Athena 무엇인가?

마이크로소프트 윈도우용 Amazon Kinesis 에이전트 문제 해결

다음 지침에 따라 Microsoft Windows용 Amazon Kinesis 에이전트를 사용할 때 문제를 진단하고 수정합니다.

주제

- 데스크톱 또는 서버에서 예상 AWS 서비스로 스트리밍되는 데이터가 없음
- 예상 데이터가 때때로 누락됨
- 데이터가 잘못된 형식으로 도착함
- 성능 문제
- 디스크 공간 부족
- 도구 문제 해결

데스크톱 또는 서버에서 예상 AWS 서비스로 스트리밍되는 데이터가 없음

Symptoms

Windows용 Kinesis 에이전트에서 데이터 스트림을 수신하도록 구성된 다양한 AWS 서비스에서 호스팅하는 로그, 이벤트 및 메트릭을 검사할 때 Windows용 Kinesis 에이전트에 의해 데이터가 스트리밍되지 않습니다.

Causes

이러한 문제가 발생하는 몇 가지 원인이 있을 수 있습니다.

- 소스, 싱크 또는 파이프가 잘못 구성되었습니다.
- Windows용 Kinesis 에이전트에 대한 인증이 잘못 구성되었습니다.
- Windows용 Kinesis 에이전트에 대한 권한 부여가 잘못 구성되었습니다.
- 에 제공된 정규 표현식에 오류가 있습니다.`DirectorySource`선언을 사용합니다.
- 존재하지 않는 디렉토리는`DirectorySource`선언을 사용합니다.
- AWS 서비스에 잘못된 값이 제공되어 Windows용 Kinesis 에이전트의 요청이 거부됩니다.

- 싱크는 지정된 또는 암시적 AWS 리전에 존재하지 않는 리소스를 참조하고 있습니다.
- 잘못된 쿼리는`WindowsEventLogSource`선언을 사용합니다.
- 잘못 된 값이 지정 된`InitialPosition`소스의 키-값 페어의 입니다.
- 이`appsettings.json`구성 파일이 해당 파일에 대한 JSON 스키마를 준수하지 않습니다.
- 데이터는 AWS 관리 콘솔에서 선택한 리전과 다른 리전으로 스트리밍됩니다.
- Windows용 Kinesis 에이전트가 올바르게 설치되지 않았거나 실행 중이 아닙니다.

Resolutions

데이터가 스트리밍되지 않는 문제를 해결하려면 다음 단계를 수행하십시오.

1. Windows 용 Kinesis 에이전트 로그의`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`디렉토리로 이동합니다. 문자열 검색`ERROR`.
   a. 소스 또는 싱크대가 로드되지 않은 경우 다음을 수행합니다.
      i. 오류 메시지를 검사하고`Id`소스 또는 싱크.
      ii. 해당 소스 또는 싱크 선언을 확인하십시오.`Id`의`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에서 발견된 오류 메시지와 관련된 모든 오류를 확인하십시오. 자세한 내용은 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성 단원을 참조하십시오.
      iii. 오류와 관련된 모든 구성 파일 문제를 수정합니다.
      iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.
   b. 오류 메시지가 나타내는 경우`SourceRef`또는`SinkRef`파이프에 대해 찾을 수 없는 경우 다음을 수행합니다.
      i. 파이프를 기록합니다.`Id`.
      ii. 에서 파이프 선언을 검사`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에 해당하는`Id`. 의 값이 있는지 확인합니다.`SourceRef`및`SinkRef`키-값 페어의 철자가 올바로 지정되어 있습니다.`Id`를 참조하려는 소스 및 싱크 선언에 대해 설명합니다. 오타 또는 맞춤법 오류를 수정합니다. 구성 파일에서 소스 또는 싱크 선언이 누락 된 경우 선언을 추가하십시오. 자세한 내용은 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성 섹션을 참조하세요.
      iii. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

c. 오류 메시지에 특정 IAM 사용자 또는 역할이 특정 작업을 수행할 수 있는 권한이 없음을 나타내는 경우 다음을 수행합니다.

   i. 올바른 IAM 사용자 또는 역할을 Windows용 Kinesis 에이전트에서 사용하고 있는지 확인합니다. 그렇지 않은 경우sink 보안 구성을 클릭하고 올바른 IAM 사용자 또는 역할이 사용되고 있는지 확인하기 위해 Kinesis Windows용 에이전트가 인증하는 방법을 조정합니다.

   ii. AWS Management Console을 사용하여 올바른 IAM 사용자 또는 역할을 사용하고 있는 경우 사용자 또는 역할과 연결된 정책을 검토합니다. 사용자 또는 역할에 Windows용 Kinesis 에이전트가 액세스하는 모든 AWS 리소스에 대해 오류 메시지에 언급된 모든 권한이 있는지 확인합니다. 자세한 내용은 권한 부여 구성 섹션을 참조하세요.

   iii. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 보안 문제가 해결되었는지 확인합니다.

d. 오류 메시지에 포함 된 정규 표현식을 구문 분석 할 때 인수 오류가 있음을 나타내는 경우`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에서 다음을 수행합니다.

   i. 구성 파일에서 정규 표현식을 검사합니다.

   ii. 정규 표현식의 구문을 확인합니다. 정규식을 확인하는 데 사용할 수 있는 여러 웹 사이트가 있습니다. 또는 다음 명령줄을 사용하여 정규식을`DirectorySource`소스 선언:

```
cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
ktdiag.exe /r sourceId
```

   ReplaceSourceId의 값으로`Id`키-값 페어의`DirectorySource`소스 선언을 잘못된 정규 표현식으로 바꿉니다.

   iii. 구성 파일의 정규 표현식에 필요한 사항을 수정하여 유효합니다.

   iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

e. 오류 메시지에 포함되어 있지 않은 정규식을 구문 분석할 때 인수 오류가 있음을 나타내는 경우`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일 및 특정 싱크와 관련된 경우 다음을 수행하십시오.

   i. 구성 파일에서 싱크 선언을 찾습니다.

   ii. AWS 서비스와 특별히 관련된 키-값 쌍이 해당 서비스에 대한 유효성 검사 규칙을 준수하는 이름을 사용하고 있는지 확인합니다. 예를 들어 CloudWatch Logs 그룹 이름에는 정규식을 사용하여 지정된 특정 문자 집합만 포함되어야 합니다.`[\.\-_/#A-Za-z0-9]+`.

iii. 싱크 선언에 대한 키-값 쌍에서 잘못된 이름을 수정하고 해당 리소스가 AWS 에서 제대로 구성되었는지 확인합니다.

iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

f. 오류 메시지가 null 또는 누락 된 매개 변수로 인해 소스 또는 싱크를 로드할 수 없음을 나타내는 경우 다음을 수행하십시오.

i. 을 기록합니다.`Id`소스 또는 싱크.

ii. 언급 된 것과 일치하는 소스 또는 싱크 선언을 찾습니다.`Id`의`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일을 저장합니다.

iii. 소스 또는 싱크 유형 요구 사항과 비교하여 소스 또는 싱크 선언에서 제공되는 키 - 값 쌍을 검토마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성관련 싱크 유형에 대한 설명서를 참조하십시오. 누락 된 필수 키 - 값 쌍을 소스 또는 싱크 선언에 추가합니다.

iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

g. 오류 메시지가 디렉터리 이름이 잘못되었음을 나타내면 다음을 수행합니다.

i. 잘못된 디렉터리 이름을 찾습니다.`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일을 저장합니다.

ii. 이 디렉터리가 존재하고 스트리밍해야 하는 로그 파일이 포함되어 있는지 확인합니다.

iii. 구성 파일에 지정된 디렉토리 이름의 오타 또는 실수를 수정합니다.

iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

h. 오류 메시지가 자원이 존재하지 않음을 나타내는 경우:

i. 싱크 선언에 존재하지 않는 리소스에 대한 리소스 참조를 찾습니다`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에 저장됩니다.

ii. AWS 관리 콘솔을 사용하여 싱크 선언에 사용해야 하는 올바른 AWS 리전에서 리소스를 찾습니다. 구성 파일에 지정된 항목과 비교합니다.

iii. 올바른 자원 이름과 올바른 지역을 가지고 구성 파일의 싱크 선언을 변경합니다.

iv. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

i. 오류 메시지가 특정 쿼리가 유효하지 않음을 나타내는 경우`WindowsEventLogSource`에서 다음을 수행합니다.

i. 에서`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에 대한 자세한 내용은`WindowsEventLogSource`선언을 동일한`Id`오류 메시지와 같습니다.

ii. 값을 확인 하는Query소스 선언의 키-값 페어는이벤트 쿼리 및 이벤트 XML.

iii. 쿼리를 변경하여 규정을 준수하도록 만듭니다.

iv. 중지 및 시작AWSKinesisTap서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

j. 오류 메시지에 잘못된 초기 위치가 있음을 나타내는 경우 다음을 수행합니다.

i. 에서%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json구성 파일에서 동일한Id오류 메시지로 표시됩니다.

ii. 의 값을 변경합니다.InitialPosition에 설명 된대로 허용 된 값을 준수하기 위해 소스 선언의 키 - 값 쌍책갈피 구성.

iii. 중지 및 시작AWSKinesisTap서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

2. 파일이 설치되어 있는지 확인합니다.%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json구성 파일은 JSON 스키마를 준수합니다.

a. 명령 프롬프트 창에서 다음 줄을 호출합니다.

```
cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
%PROGRAMFILES%\Amazon\AWSKinesisTap\ktdiag.exe /c
```

b. 로 감지된 모든 문제를 수정하십시오.%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json구성 파일에 저장됩니다.

c. 중지 및 시작AWSKinesisTap서비스. 그런 다음 최신 로그 파일을 확인하여 구성 문제가 해결되었는지 확인합니다.

3. 로깅 수준을 변경하여 자세한 로깅 정보를 얻으십시오.

a. 를 대체합니다.%PROGRAMFILES%\Amazon\AWSKinesisTap\nlog.xml구성 파일에 다음 콘텐츠가 포함되어 있습니다.

```
<?xml version="1.0" encoding="utf-8" ?>
<nlog xmlns="http://www.nlog-project.org/schemas/NLog.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.nlog-project.org/schemas/NLog.xsd NLog.xsd"
      autoReload="true"
      throwExceptions="false"
      internalLogLevel="Off" internalLogFile="c:\temp\nlog-internal.log" >

  <!--
  See https://github.com/nlog/nlog/wiki/Configuration-file
```

```
  for information on customizing logging rules and outputs.
   -->
  <targets>
    <!--
    add your targets here
    See https://github.com/nlog/NLog/wiki/Targets for possible targets.
    See https://github.com/nlog/NLog/wiki/Layout-Renderers for the possible layout
 renderers.
    -->

    <target name="logfile"
            xsi:type="File"
            layout="${longdate} ${logger} ${uppercase:${level}} ${message}"
            fileName="${specialfolder:folder=CommonApplicationData}/Amazon/
KinesisTap/logs/KinesisTap.log"
     maxArchiveFiles="90"
     archiveFileName="${specialfolder:folder=CommonApplicationData}/Amazon/
KinesisTap/logs/Archive-{################}.log"
     archiveNumbering="Date"
     archiveDateFormat="yyyy-MM-dd"
     archiveEvery="Day"
     />
  </targets>

  <rules>
    <logger name="*" minlevel="Debug" writeTo="logfile" />
  </rules>
</nlog>
```

b. 중지 및 시작`AWSKinesisTap`서비스. 그런 다음 최신 로그 파일을 확인하여 문제를 진단하고 해결하는 데 도움이 될 수 있는 추가 메시지가 로그에 있는지 확인합니다.

4. AWS 관리 콘솔에서 올바른 리전의 리소스를 확인하고 있는지 확인합니다.

5. Windows용 Kinesis 에이전트 에이전트가 설치되어 실행 중인지 확인합니다.

   a. Windows에서Start을 클릭한 다음제어판,관리 도구,서비스.

   b. 를 찾습니다.AW스키네시스탭서비스.

   c. AWSkinesistap 서비스가 표시되지 않는 경우Microsoft Windows용 Amazon Kinesis 에이전트 시작하기.

   d. 서비스가 표시되는 경우 서비스가 실행 중인지 확인합니다. 이 서비스가 실행되고 있지 않으면 서비스의 컨텍스트 (마우스 오른쪽 버튼 클릭) 메뉴를 열고Start.

e. 최신 로그 파일을 검사하여 서비스가 시작되었는지 확인합니다.`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`디렉토리로 이동합니다.

적용 대상

이 정보는 Kinesis 에이전트 버전 1.0.0.115 이상에 적용됩니다.

예상 데이터가 때때로 누락됨

Symptoms

Windows용 Kinesis 에이전트는 대부분의 경우 데이터를 성공적으로 스트리밍하지만 일부 데이터가 누락되는 경우가 있습니다.

Causes

이러한 문제가 발생하는 몇 가지 원인이 있을 수 있습니다.

- 북마크 기능이 사용되고 있지 않습니다.
- AWS 서비스의 데이터 속도 제한은 해당 서비스의 현재 구성에 따라 초과됩니다.
- AWS 서비스에 대한 API 호출 요금 제한은 현재`appsettings.json`구성 파일 및 AWS 계정 제한에 따라 다릅니다.

Resolutions

누락된 데이터와 관련된 문제를 해결하려면 다음 단계를 수행합니다.

1. 에 설명 된 책갈피 기능을 사용 하는 것이 좋습니다.책갈피 구성. 이를 통해 Windows용 Kinesis 에이전트가 중지되고 시작될 때에도 모든 데이터가 최종적으로 전송될 수 있습니다.
2. Windows용 Kinesis 에이전트의 기본 제공 메트릭을 사용하여 문제를 발견합니다.
   a. 에 설명된 대로 Windows용 Kinesis 에이전트 메트릭의 스트리밍을 활성화합니다.Windows 메트릭 파이프용 Kinesis 에이전트 구성.
   b. 하나 이상의 싱크에 대해 많은 수의 복구할 수 없는 오류가 있는 경우 초당 전송되는 바이트 또는 레코드 수를 확인합니다. 그런 다음 데이터가 스트리밍되는 리전 및 계정의 해당 AWS 서비스에 대해 구성된 제한 범위 내에 있는지 확인합니다.

c. 한도가 초과되면 전송되는 데이터의 속도 또는 양을 줄이거나, 요청 제한을 늘리거나, 해당하는 경우 샤딩을 늘립니다.

d. 조정 후 Kinesis Windows용 기본 제공 메트릭을 계속 모니터링하여 상황이 해결되었는지 확인합니다.

Kinesis Data Streams 한도에 대한 자세한 내용은 단원을 참조하십시오.Kinesis Data Streams 제한의Kinesis Data Streams 개발자 가이드. Kinesis Data Firehose 한도에 대한 자세한 내용은Amazon Kinesis Data Firehose 제한.

적용 대상

이 정보는 Kinesis 에이전트 버전 1.0.0.115 이상에 적용됩니다.

데이터가 잘못된 형식으로 도착함

Symptoms

데이터가 잘못된 형식으로 AWS 서비스에 도착합니다.

Causes

이러한 문제가 발생하는 몇 가지 원인이 있을 수 있습니다.

- 의 값입니다.`Format`Kinesis 선언에 대한 키-값 페어의`appsettings.json`구성 파일이 올바르지 않습니다.
- 의 값입니다.`RecordParser`키-값 페어의`DirectorySource`선언이 올바르지 않습니다.
- A의 정규 표현식`DirectorySource`선언을 사용하는`Regex`레코드 파서가 올바르지 않습니다.

Resolutions

잘못된 서식 문제를 해결하려면 다음 단계를 수행합니다.

1. 의 싱크 선언을 검토`%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`구성 파일에 저장됩니다.
2. 올바른 값이 있는지 확인 하십시오.`Format`키 - 값 쌍은 각 싱크 선언에 대해 지정됩니다. 자세한 내용은 sink 선언 섹션을 참조하세요.

3. 다음과 같은 소스`DirectorySource`선언은 파이프에 의해`xml`또는`json`값에 대한`Format`키 - 값 쌍에 대한 다음 값 중 하나를 지정했는지 확인하십시오.`RecordParser`키-값 페어:
   - `SingleLineJson`
   - `Regex`
   - `SysLog`
   - `Delimited`

   다른 레코드 파서는 텍스트 기반이며 XML 또는 JSON 서식이 필요한 싱크에서는 올바르게 작동하지 않습니다.
4. 로그 레코드가 올바르게 구문 분석되지 않는 경우`DirectorySource`소스 유형에 지정된 타임 스탬프 및 정규 표현식 키 - 값 쌍을 확인하기 위해 명령 프롬프트 창에서 다음 행을 호출합니다`DirectorySource`선언을 수행합니다.

   ```
   cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
   ktdiag.exe /r sourceID
   ```

   Replace*`SourceId`*의 값으로`Id`키-값 페어의`DirectorySource`소스 선언이 올바르게 작동하지 않는 것으로 보입니다. 에 의해 보고된 모든 문제 해결`ktdiag.exe`.

적용 대상

이 정보는 Kinesis 에이전트 버전 1.0.0.115 이상에 적용됩니다.

성능 문제

Symptoms

Windows용 Kinesis 에이전트를 설치하고 시작한 후 응용 프로그램 및 서비스의 대기 시간이 늘어났습니다.

Causes

이러한 문제가 발생하는 몇 가지 원인이 있을 수 있습니다.

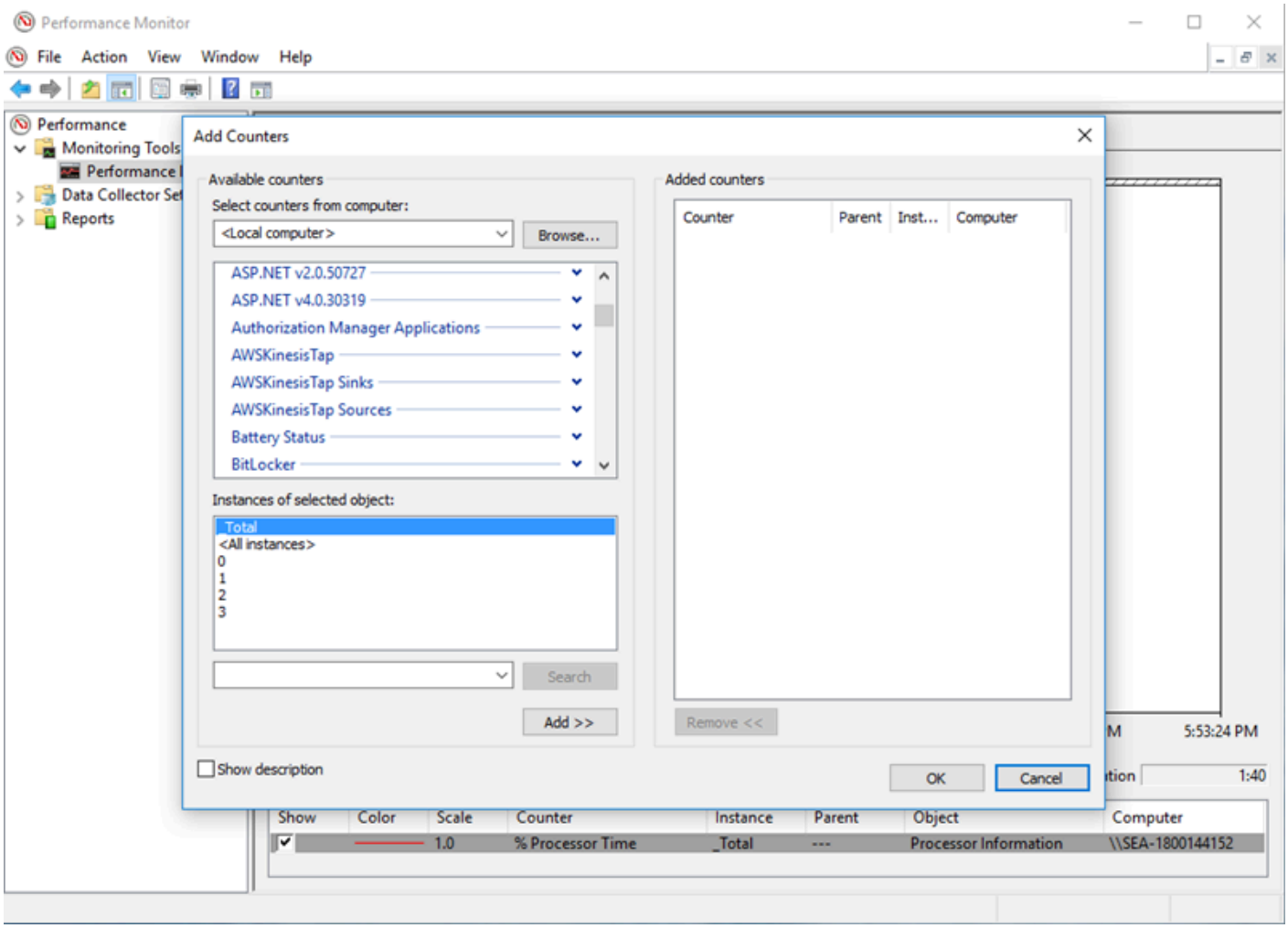
- Windows용 Kinesis 에이전트가 실행되는 시스템의 용량이 부족하여 원하는 데이터 양을 스트리밍할 수 없습니다.

- 불필요한 데이터가 하나 이상의 AWS 서비스로 스트리밍됩니다.
- Windows용 Kinesis 에이전트는 이렇게 높은 데이터 전송률로 구성되지 않은 AWS 서비스로 데이터를 스트리밍하고 있습니다.
- Windows용 Kinesis 에이전트가 API 호출 속도 제한이 너무 낮은 계정의 AWS 서비스에서 작업을 호출합니다.

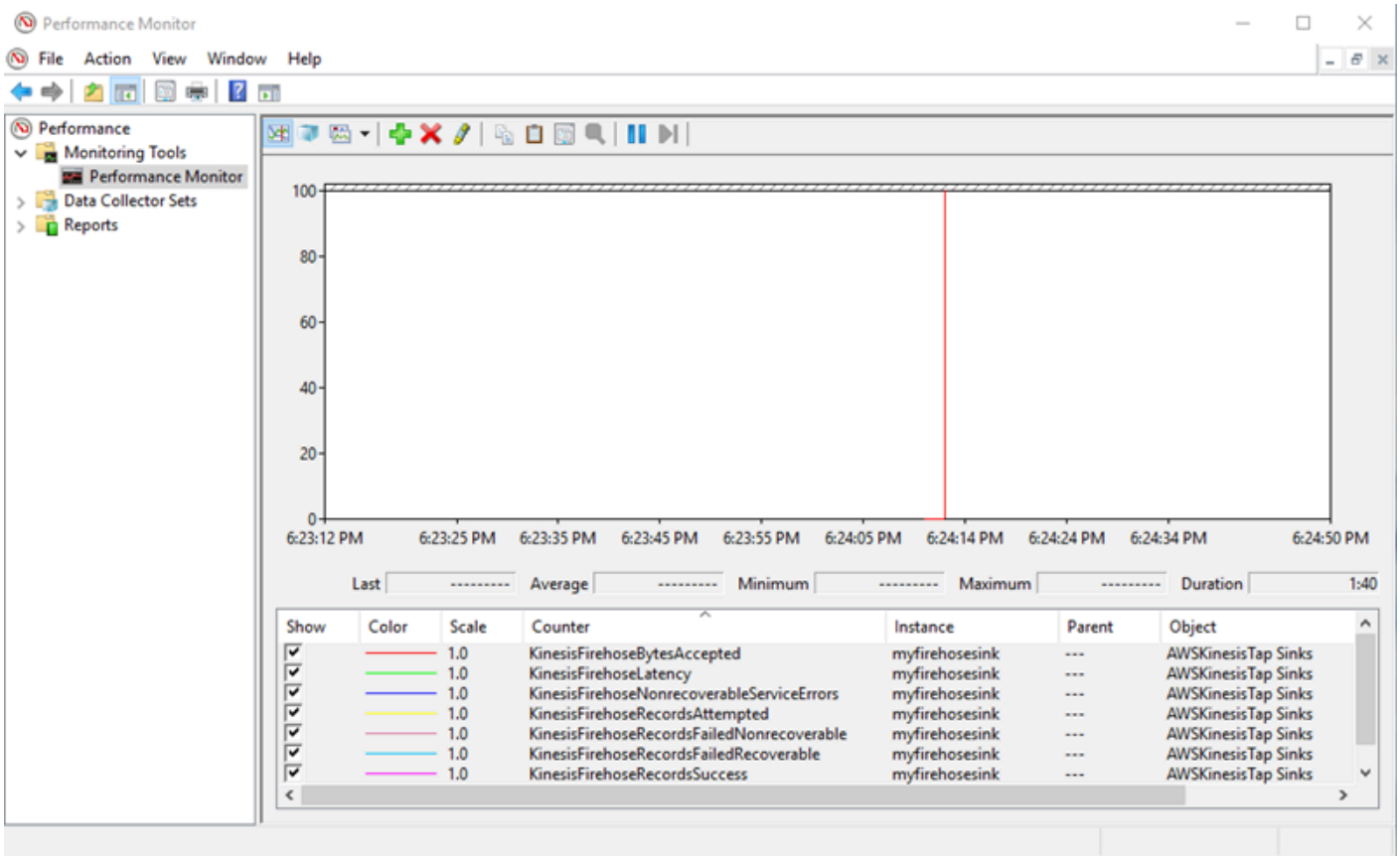
Resolutions

성능 문제를 해결하려면 다음 단계를 수행합니다.

1. Windows 리소스 모니터 응용 프로그램을 사용하여 메모리, CPU, 디스크 및 네트워크 사용량을 확인합니다. Windows용 Kinesis Agent를 사용하여 대량의 데이터를 스트리밍해야 하는 경우 구성에 따라 이러한 영역 중 일부에서 더 많은 용량의 시스템을 프로비저닝해야 할 수 있습니다.
2. 필터링을 사용하여 기록된 데이터의 양을 줄일 수 있습니다.
   - 단원을 참조하십시오.`Query`키-값 페어를 입력합니다.창 세븐로그소스 구성.
   - 파이프라인 필터링 참조파이프 구성.
   - Amazon CloudWatch 지표 필터링을CloudWatch 싱크 구성).
3. Windows 성능 모니터 애플리케이션을 사용하여 Windows용 Kinesis 에이전트 메트릭을 보거나 해당 메트릭을 CloudWatch 로 스트리밍합니다 (Windows용 Kinesis 에이전트 기본 제공 메트릭 소스). Windows 성능 모니터 응용 프로그램에서 Windows용 Kinesis 에이전트 싱크 및 소스에 대한 카운터를 추가할 수 있습니다. 그들은 아래에 나열되어Kinesis Kinesis Kinesis및Amazon Kinesis stap 소스카운터 범주를 선택합니다.



예를 들어, Kinesis Data Firehose 성능 문제를 진단하려면Kinesis Firehose 싱크대성능 카운터를 사용합니다.



복구 가능한 오류가 많은 경우`%PROGRAMDATA%\Amazon\AWSKInesisTap\logs`디렉토리로 이동합니다. 에 대한 스로틀이 발생하는 경우`KinesisStream`또는`KinesisFirehose`싱크대에 대해 다음을 수행합니다.

- 스트리밍 데이터로 인해 제한이 너무 빨리 발생하면 Kinesis 데이터 스트림의 샤드 수를 늘리는 것이 좋습니다. 자세한 내용은 단원을 참조하십시오.리샤딩, 크기 조정 및 병렬 처리의Kinesis Data Streams 개발자 가이드.
- Kinesis Data Streams 에 대한 API 호출 제한을 늘리거나 API 호출이 제한되는 경우 싱크의 버퍼 크기를 늘리는 것이 좋습니다. 자세한 내용은 단원을 참조하십시오.Kinesis Data Streams 제한의Kinesis Data Streams 개발자 가이드.
- 데이터가 너무 빨리 스트리밍되면 Kinesis Data Firehose 전송 스트림을 위한 속도 제한 증가를 요청하는 것이 좋습니다. 또는 API 호출이 제한되는 경우 API 호출 제한 증가를 요청하십시오 (Amazon Kinesis Data Firehose 제한) 또는 싱크의 버퍼 크기를 늘리십시오.
- Kinesis 데이터 스트림 스트림의 샤드 수를 늘리거나 KinKinesis Data Firehose 전송 스트림에 대한 속도 제한을 늘린 후 Windows용 Kinesis 에이전트를 수정합니다.`appsettings.json`구성 파일을 사용하여 싱크의 초당 레코드 또는 초당 바이트를 늘릴 수 있습니다. 그렇지 않으면 Windows용 Kinesis 에이전트가 증가된 제한을 활용할 수 없습니다.

적용 대상

이 정보는 Kinesis 에이전트 버전 1.0.0.115 이상에 적용됩니다.

디스크 공간 부족

Symptoms

Windows용 Kinesis 에이전트가 하나 이상의 디스크 드라이브에서 디스크 공간이 매우 부족한 시스템에서 실행되고 있습니다.

Causes

이러한 문제가 발생하는 몇 가지 원인이 있을 수 있습니다.

- Windows용 Kinesis 에이전트 로깅 구성 파일이 잘못되었습니다.
- Windows용 Kinesis 에이전트 영구 큐가 잘못 구성되었습니다.
- 일부 다른 응용 프로그램이나 서비스에서 디스크 공간을 사용하고 있습니다.

Resolutions

디스크 공간 문제를 해결하려면 다음 단계를 수행하십시오.

- Windows용 Kinesis 에이전트 로그 파일이 들어 있는 디스크의 디스크 공간이 부족한 경우 로그 파일 디렉터리 (일반적으로`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`). 적절한 수의 로그 파일이 보존되고 있고 로그 파일의 크기가 적당한지 확인합니다. Kinesis Windows용 Kinesis 에이전트 로그의 위치, 보존 및 세부 정보를 제어하려면`%PROGRAMFILES%\Amazon\AWSKinesisTap\Nlog.xml`구성 파일에 저장됩니다.
- 싱크 큐 기능을 사용하는 경우 해당 기능을 사용하는 싱크 선언을 검사합니다. 파일이 설치되어 있는지 확인합니다.`QueuePath`키-값 쌍을 사용하여 지정된 최대 배치 수를 포함하기에 충분한 공간이 있는 디스크 드라이브를 참조합니다.`QueueMaxBatches`키-값 페어를 지정합니다. 이것이 가능하지 않은 경우`QueueMaxBatches`키 - 값 쌍을 사용하여 지정된 디스크 드라이브의 나머지 디스크 공간에 데이터가 쉽게 들어갈 수 있습니다.
- Windows 파일 탐색기를 사용하여 디스크 공간을 사용하는 파일을 찾고 초과 파일을 전송하거나 삭제합니다. 많은 양의 디스크 공간을 소비하는 응용 프로그램 또는 서비스의 구성을 변경합니다.

적용 대상

이 정보는 Kinesis 에이전트 버전 1.0.0.115 이상에 적용됩니다.

도구 문제 해결

구성 파일을 확인하는 것 외에도`ktdiag.exe`도구를 사용하여 Windows용 Kinesis 에이전트를 구성하고 사용할 때 문제를 진단하고 해결할 수 있는 몇 가지 다른 기능을 제공합니다. 이`ktdiag.exe`도구는`%PROGRAMFILES%\Amazon\AWSKinesisTap`디렉토리로 이동합니다.

- 특정 파일 패턴을 가진 로그 파일이 디렉터리에 기록되고 있지만 Windows용 Kinesis 에이전트에서 처리하지 않는다고 생각되는 경우`/w`스위치를 사용 하 여 이러한 변경 내용이 검색 되 고 있는지 확인 합니다. 예를 들어 로그 파일이`*.log`파일 이름 패턴이`c:\foo`디렉토리로 이동합니다. 다음을 수행할 수 있습니다.`/w`스위치를 실행할 때`ktdiag.exe`도구를 사용하여 디렉토리 및 파일 패턴을 지정할 수 있습니다.

  ```
  cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
  ktdiag /w c:\foo *.log
  ```

  로그 파일이 작성되면 다음과 비슷한 출력이 표시됩니다.

  ```
  Type any key to exit this program...
  File: c:\foo\log1.log ChangeType: Created
  File: c:\foo\log1.log ChangeType: Deleted
  File: c:\foo\log1.log ChangeType: Created
  File: c:\foo\log1.log ChangeType: Changed
  File: c:\foo\log1.log ChangeType: Changed
  File: c:\foo\log1.log ChangeType: Changed
  File: c:\foo\log1.log ChangeType: Changed
  ```

  이러한 출력이 발생하지 않으면 로그를 작성할 때 응용 프로그램 또는 서비스 문제가 있거나 Windows용 Kinesis 에이전트에 문제가 아니라 보안 구성 문제가 있는 것입니다. 이러한 출력이 발생하지만 Windows용 Kinesis 에이전트가 여전히 로그를 처리하지 않는 경우데스크톱 또는 서버에서 예상 AWS 서비스로 스트리밍되는 데이터가 없음.

- 로그가 때때로 기록되는 경우도 있지만 Windows용 Kinesis 에이전트가 올바르게 작동하는지 확인하는 것이 유용합니다. 사용`/log4net`스위치를 사용하여 로그를 작성하는 응용 프로그램을 시뮬레이트하는`Log4net`라이브러리를 사용할 수 있습니다. 예:

```
cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
KTDiag.exe /log4net c:\foo\log2.log
```

이 쓰기`Log4net`스타일 로그 파일을`c:\foo\log2.log`로그 파일을 만들고 키를 누를 때까지 새 로그 항목을 계속 추가합니다. 파일 이름 뒤에 선택적으로 지정된 추가 스위치를 사용하여 여러 옵션을 구성할 수 있습니다.

잠금:`-lm`,`-li`또는`-le`

로그 파일이 잠기는 방법을 제어하는 다음 잠금 스위치 중 하나를 지정할 수 있습니다.

`-lm`

로그 파일에 대한 최소 잠금 양이 사용되므로 로그 파일에 대한 최대 액세스가 가능합니다.

`-li`

동일한 프로세스 내의 스레드만 동시에 로그에 액세스할 수 있습니다.

`-le`

한 번에 하나의 스레드만 로그에 액세스할 수 있습니다. 이 값이 기본값입니다.

`-tn:###`

의 수를 지정합니다.`###`로그 항목을 쓰는 사이. 기본값은 1000밀리초 (1초) 입니다.

`-sm:###`

의 수를 지정합니다.`###`각 로그 항목에 대해 설명합니다. 기본값은 1,000바이트입니다.

`-bk:##`

를 지정합니다.`##`한 번에 쓸 로그 항목. 기본값은 1입니다.

- 때로는 Windows 이벤트 로그에 쓰는 응용 프로그램을 시뮬레이트하는 것이 유용합니다. 사용`/e`스위치를 사용하여 Windows 이벤트 로그에 로그 항목을 기록할 수 있습니다. 예를 들면 다음과 같습니다.

```
cd /D %PROGRAMFILES%\Amazon\AWSKinesisTap
KTDiag.exe /e Application
```

키를 누를 때까지 Windows 응용 프로그램 이벤트 로그에 로그 항목이 기록됩니다. 필요에 따라 로그 이름 뒤에 다음과 같은 추가 옵션을 지정할 수 있습니다.

`-tn:###`

의 수를 지정합니다.###로그 항목을 쓰는 사이. 기본값은 1000밀리초 (1초) 입니다.

`-sm:###`

의 수를 지정합니다.###각 로그 항목에 대해 설명합니다. 기본값은 1,000바이트입니다.

`-bk:##`

를 지정합니다.##한 번에 쓸 로그 항목. 기본값은 1입니다.

윈도 플러그인용 Kinesis 에이전트 생성

대부분의 경우, 마이크로소프트 윈도우용 Amazon Kinesis 에이전트 플러그인을 생성할 필요가 없습니다. Windows용 Kinesis 에이전트는 고도로 구성할 수 있으며 다음과 같은 강력한 소스 및 싱크가 포함되어 있습니다.`DirectorySource`및`KinesisStream`이며 대부분의 시나리오에 충분합니다. 기존 소스 및 싱크에 대한 자세한 내용은 단원을 참조하십시오.마이크로소프트 윈도우용 Amazon Kinesis 에이전트 구성.

비정상적인 시나리오의 경우 사용자 지정 플러그인을 사용하여 Windows용 Kinesis 에이전트를 확장해야 할 수 있습니다. 몇 가지 이러한 시나리오는 다음과 같습니다.

- 컴플렉스 패키징`DirectorySource`를 사용하여 선언을`Regex`또는`Delimited`레코드 파서를 사용하여 다양한 종류의 구성 파일에 쉽게 적용 할 수 있습니다.
- 파일 기반이 아니거나 기존 레코드 파서가 제공하는 구문 분석 기능을 초과하는 새로운 소스를 만듭니다.
- 현재 지원되지 않는 AWS 서비스용 싱크 생성

주제

- 윈도 플러그인용 Kinesis 에이전트 시작하기
- Windows 플러그인 팩토리를 위한 Kinesis 에이전트 구현
- Windows 플러그인 소스용 Kinesis 에이전트 구현
- Windows 플러그인 싱크용 Kinesis 에이전트 구현

윈도 플러그인용 Kinesis 에이전트 시작하기

사용자 정의 플러그인에는 특별한 것이 없습니다. 기존의 모든 소스 및 싱크는 사용자 정의 플러그인이 Windows용 Kinesis 에이전트가 시작될 때 로드하는 데 사용하는 것과 동일한 메커니즘을 사용하며`appsettings.json`구성 파일을 참조하십시오.

Windows용 Kinesis 에이전트가 시작되면 다음 순서가 발생합니다.

1. Windows용 Kinesis 에이전트는 설치 디렉토리 (`%PROGRAMFILES%\Amazon\AWSKinesisTap`)를 구현 하는 클래스에 대 한`IFactory<T>`인터페이스에 정의 된`Amazon.KinesisTap.Core`어셈블리를 생성합니다. 이 인터페이스는`Amazon.KinesisTap.Core\Infrastructure\IFactory.cs`를 Windows용 Kinesis 에이전트 소스 코드에 입력합니다.

2. Windows용 Kinesis 에이전트는 이러한 클래스를 포함하는 어셈블리를 로드하고`RegisterFactory`메서드를 호출합니다.
3. Windows용 Kinesis 에이전트는`appsettings.json`구성 파일을 참조하십시오. 구성 파일의 각 소스 및 싱크에 대해`SourceType`및`SinkType`키-값 페어가 검사됩니다. 의 값과 같은 이름으로 등록된 공장이있는 경우`SourceType`및`SinkType`키-값 페어의 경우`CreateInstance`메서드가 해당 팩토리에서 호출됩니다. 이`CreateInstance`메서드는 구성 및 기타 정보를`IPluginContext`객체입니다. 이`CreateInstance`메소드는 플러그인을 구성하고 초기화하는 것을 담당합니다.

플러그인이 올바르게 작동하려면 플러그인을 생성하는 등록 된 팩토리 클래스가 있어야하며 플러그인 클래스 자체를 정의해야합니다.

Windows용 Kinesis 에이전트 소스 코드는https://github.com/awslabs/kinesis-agent-windows.

Windows 플러그인 팩토리를 위한 Kinesis 에이전트 구현

다음 단계에 따라 Windows용 Kinesis 에이전트 플러그인 팩토리를 구현합니다.

Windows용 Kinesis 에이전트 플러그인 팩토리를 생성하려면

1. .NET Framework 4.6을 대상으로 C# 라이브러리 프로젝트를 생성합니다.
2. 를 참조할 항목을 추가합니다.`Amazon.KinesisTap.Core`어셈블리를 생성합니다. 이 어셈블리는`%PROGRAMFILES%\Amazon\AWSKinesisTap`디렉터리에 설치할 수 있습니다.
3. `NuGet`를 사용하여 `Microsoft.Extensions.Configuration.Abstractions` 패키지를 설치합니다.
4. `NuGet`를 사용하여 `System.Reactive` 패키지를 설치합니다.
5. `NuGet`를 사용하여 `Microsoft.Extensions.Logging` 패키지를 설치합니다.
6. 다음 중 하나를 구현하는 팩토리 클래스를 만듭니다.`IFactory<IEventSource>`소스 또는`IFactory<IEventSink>`싱크 용. 를 추가합니다.`RegisterFactory`및`CreateInstance`메서드를 사용합니다.

   예를 들어 다음 코드에서는 임의의 데이터를 생성하는 소스를 생성하는 Windows용 Kinesis 에이전트 플러그인 팩토리를 생성합니다.

```
using System;
using Amazon.KinesisTap.Core;
using Microsoft.Extensions.Configuration;
```

```
namespace MyCompany.MySources
{
    public class RandomSourceFactory : IFactory<ISource>
    {
        public void RegisterFactory(IFactoryCatalog<ISource> catalog)
        {
            catalog.RegisterFactory("randomsource", this);
        }

        public ISource CreateInstance(string entry, IPlugInContext context)
        {
            IConfiguration config = context.Configuration;

            switch (entry.ToLower())
            {
                case "randomsource":
                    string rateString = config["Rate"];
                    string maxString = config["Max"];
                    TimeSpan rate;
                    int max;

                    if (string.IsNullOrWhiteSpace(rateString))
                    {
                        rate = TimeSpan.FromSeconds(30);
                    }
                    else
                    {
                        if (!TimeSpan.TryParse(rateString, out rate))
                        {
                            throw new Exception($"Rate {rateString} is invalid for
 RandomSource.");
                        }
                    }

                    if (string.IsNullOrWhiteSpace(maxString))
                    {
                        max = 1000;
                    }
                    else
                    {
                        if (!int.TryParse(maxString, out max))
                        {
                            throw new Exception($"Max {maxString} is invalid for
 RandomSource.");
```

```
                        }
                    }

                    return new RandomSource(rate, max, context);
                default:
                    throw new ArgumentException($"Source {entry} is not
 recognized.", entry);
            }
        }
    }
}
```

이switch문에 사용되는CreateInstance메서드를 사용하면 결국 팩토리를 향상시켜 다른 종류의 인스턴스를 만들 수 있습니다.

아무 것도하지 않는 싱크를 만드는 싱크 팩토리를 만들려면 다음과 유사한 클래스를 사용하십시오.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Amazon.KinesisTap.Core;
using Microsoft.Extensions.Configuration;

namespace MyCompany.MySinks
{
    public class NullSinkFactory : IFactory<IEventSink>
    {
        public void RegisterFactory(IFactoryCatalog<IEventSink> catalog)
        {
            catalog.RegisterFactory("nullsink", this);
        }

        public IEventSink CreateInstance(string entry, IPlugInContext context)
        {
            IConfiguration config = context.Configuration;

            switch (entry.ToLower())
            {
                case "nullsink":
                    return new NullSink(context);
```

```
                default:
                    throw new Exception("Unrecognized sink type {entry}.");
            }
        }
    }
}
```

Windows 플러그인 소스용 Kinesis 에이전트 구현

다음 단계에 따라 Windows용 Kinesis 에이전트 플러그인 소스를 구현합니다.

Windows용 Kinesis 에이전트 플러그인 소스를 생성하려면

1. 를 구현하는 클래스를 추가합니다.`IEventSource<out T>`인터페이스를 원본에 대해 이전에 생성 된 프로젝트에 추가합니다.

   예를 들어, 다음 코드를 사용하여 임의의 데이터를 생성하는 원본을 정의합니다.

```
using System;
using System.Reactive.Subjects;
using System.Timers;
using Amazon.KinesisTap.Core;
using Microsoft.Extensions.Logging;

namespace MyCompany.MySources
{
    public class RandomSource : EventSource<RandomData>, IDisposable
    {
        private TimeSpan _rate;
        private int _max;
        private Timer _timer = null;
        private Random _random = new Random();
        private ISubject<IEnvelope<RandomData>> _recordSubject = new
 Subject<IEnvelope<RandomData>>();



        public RandomSource(TimeSpan rate, int max, IPlugInContext context) :
 base(context)
        {
            _rate = rate;
            _max = max;
```

```
        }

        public override void Start()
        {
            try
            {
                CleanupTimer();
                _timer = new Timer(_rate.TotalMilliseconds);
                _timer.Elapsed += (Object source, ElapsedEventArgs args) =>
                {
                    var data = new RandomData()
                    {
                        RandomValue = _random.Next(_max)
                    };
                    _recordSubject.OnNext(new Envelope<RandomData>(data));
                };
                _timer.AutoReset = true;
                _timer.Enabled = true;
                _logger?.LogInformation($"Random source id {this.Id} started with
rate {_rate.TotalMilliseconds}.");
            }
            catch (Exception e)
            {
                _logger?.LogError($"Exception during start of RandomSource id
{this.Id}: {e}");
            }
        }

        public override void Stop()
        {
            try
            {
                CleanupTimer();
                _logger?.LogInformation($"Random source id {this.Id} stopped.");

            }
            catch (Exception e)
            {
                _logger?.LogError($"Exception during stop of RandomSource id
{this.Id}: {e}");
            }
        }

        private void CleanupTimer()
```

```
        {
            if (_timer != null)
            {
                _timer.Enabled = false;
                _timer?.Dispose();
                _timer = null;
            }
        }

        public override IDisposable Subscribe(IObserver<IEnvelope<RandomData>>
 observer)
        {
            return this._recordSubject.Subscribe(observer);
        }

        public void Dispose()
        {
            CleanupTimer();
        }
    }
}
```

이 예에서는RandomSource클래스에서 상속EventSource<T>클래스를 제공하기 때문에Id속성입니다. 이 예제에서는 책갈피를 지원하지 않지만 이 기본 클래스는 해당 기능을 구현하는 데에도 유용합니다. 봉투는 메타 데이터를 저장하고 싱크로 스트리밍하기 위해 임의의 데이터를 래핑하는 방법을 제공합니다. 이RandomData클래스는 다음 단계에서 정의되며 이 소스의 출력 객체 유형을 나타냅니다.

2. 원본에서 스트리밍되는 데이터가 들어 있는 이전에 정의된 프로젝트에 클래스를 추가합니다.

   예를 들어, 랜덤 데이터에 대한 컨테이너는 다음과 같이 정의 될 수 있습니다.

```
namespace MyCompany.MySources
{
    public class RandomData
    {
        public int RandomValue { get; set; }
    }
}
```

3. 이전에 정의 된 프로젝트를 컴파일하십시오.
4. 어셈블리를 Windows용 Kinesis 에이전트의 설치 디렉터리에 복사합니다.

5. 생성 또는 업데이트`appsettings.json`구성 파일을 열고 Windows용 Kinesis 에이전트의 설치 디렉토리에 저장합니다.
6. Windows용 Kinesis 에이전트를 중지했다가 시작합니다.
7. 현재 Windows용 Kinesis 에이전트 로그 파일 (일반적으로`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`디렉토리) 를 사용하여 사용자 정의 소스 플러그인에 문제가 없는지 확인하십시오.
8. 데이터가 원하는 AWS 서비스에 도착하는지 확인합니다.

를 확장하는 방법의 예는 단원을 참조하십시오.`DirectorySource`기능을 사용하여 특정 로그 형식의 구문 분석을 구현하는 방법에 대한 자세한 내용은`Amazon.KinesisTap.Uls\UlsSourceFactory.cs`및`Amazon.KinesisTap.Uls\UlsLogParser.cs`를 Windows용 Kinesis 에이전트 소스 코드에 입력합니다.

책갈피 기능을 제공하는 소스를 생성하는 방법의 예제는 단원을 참조하십시오.`Amazon.KinesisTap.Windows\WindowsSourceFactory.cs`및`Amazon.KinesisTap.Windows\EventLogSource.cs`를 Windows용 Kinesis 에이전트 소스 코드에 입력합니다.

Windows 플러그인 싱크용 Kinesis 에이전트 구현

다음 단계에 따라 Windows용 Kinesis 에이전트 플러그인 싱크를 구현합니다.

Windows용 Kinesis 에이전트 플러그인 싱크를 만들려면

1. 구현 하는 이전에 정의 된 프로젝트에 클래스를 추가`IEventSink`인터페이스를 구현해야 합니다.

   예를 들어, 다음 코드는 다음 폐기되는 레코드의 도착을 기록 이외의 아무것도하지 않는 싱크를 구현합니다.

```
using Amazon.KinesisTap.Core;
using Microsoft.Extensions.Logging;

namespace MyCompany.MySinks
{
    public class NullSink : EventSink
    {
        public NullSink(IPlugInContext context) : base(context)
        {
        }
```

```
        public override void OnNext(IEnvelope envelope)
        {
            _logger.LogInformation($"Null sink {Id} received
 {GetRecord(envelope)}.");
        }

        public override void Start()
        {
            _logger.LogInformation($"Null sink {Id} starting.");
        }

        public override void Stop()
        {
            _logger.LogInformation($"Null sink {Id} stopped.");
        }
    }
}
```

이 예에서는`NullSink`싱크 클래스는 상속`EventSink`클래스는 레코드를 JSON 및 XML과 같은 다른 직렬화 형식으로 변환하는 기능을 제공하기 때문입니다.

2. 이전에 정의 된 프로젝트를 컴파일하십시오.
3. 어셈블리를 Windows용 Kinesis 에이전트의 설치 디렉터리에 복사합니다.
4. 생성 또는 업데이트`appsettings.json`구성 파일을 찾아 Windows용 Kinesis 에이전트의 설치 디렉토리에 저장합니다. 예를 들어 를 사용합니다.`RandomSource`및`NullSink`사용자 정의 플러그인을 사용할 수 있습니다.`appsettings.json`구성 파일:

```
{
  "Sources": [
  {
 "Id": "MyRandomSource",
 "SourceType": "RandomSource",
 "Rate": "00:00:10",
 "Max": 50
  }

  ],
  "Sinks": [
  {
    "Id": "MyNullSink",
 "SinkType": "NullSink",
```

```
 "Format": "json"
  }
  ],
  "Pipes": [
    {
   "Id": "MyRandomToNullPipe",
   "SourceRef": "MyRandomSource",
   "SinkRef": "MyNullSink"
 }
  ]
}
```

이 구성은의 인스턴스를 보내는 소스를 만듭니다`RandomData`와`RandomValue`10초마다 0에서 50 사이의 임의 숫자로 설정됩니다. 들어오는 것을 변형시키는 싱크를 만듭니다.`RandomData`인스턴스를 JSON으로 변환하고 해당 JSON을 기록한 다음 인스턴스를 삭제합니다. 두 예제 팩토리를 모두 포함해야 합니다.`RandomSource`소스 클래스 및`NullSink`싱크 클래스를 사용하여 예제 구성 파일을 사용할 수 있습니다.

5. Windows용 Kinesis 에이전트를 중지했다가 시작합니다.
6. 현재 Windows용 Kinesis 에이전트 로그 파일 (일반적으로`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`디렉토리) 를 사용하여 사용자 정의 싱크 플러그인에 문제가 없는지 확인하십시오.
7. 데이터가 원하는 AWS 서비스에 도착하는지 확인합니다. 예가 있기 때문에`NullSink`가 AWS 서비스로 스트리밍되지 않는 경우 레코드가 수신되었음을 나타내는 로그 메시지를 찾아 싱크의 올바른 작동을 확인할 수 있습니다.

   예를 들어 다음과 유사한 로그 파일을 볼 수 있습니다.

```
2018-10-18 12:36:36.3647 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.AWS.AWSEventSinkFactory.
2018-10-18 12:36:36.4018 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Windows.PerformanceCounterSinkFactory.
2018-10-18 12:36:36.4018 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory MyCompany.MySinks.NullSinkFactory.
2018-10-18 12:36:36.6926 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Core.DirectorySourceFactory.
2018-10-18 12:36:36.6926 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.ExchangeSource.ExchangeSourceFactory.
2018-10-18 12:36:36.6926 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Uls.UlsSourceFactory.
```

```
2018-10-18 12:36:36.6926 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Windows.WindowsSourceFactory.
2018-10-18 12:36:36.6926 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory MyCompany.MySources.RandomSourceFactory.
2018-10-18 12:36:36.9601 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Core.Pipes.PipeFactory.
2018-10-18 12:36:37.4694 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.AutoUpdate.AutoUpdateFactory.
2018-10-18 12:36:37.4807 Amazon.KinesisTap.Hosting.LogManager INFO Performance
 counter sink  started.
2018-10-18 12:36:37.6250 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink starting.
2018-10-18 12:36:37.6250 Amazon.KinesisTap.Hosting.LogManager INFO Connected source
 MyRandomSource to sink MyNullSink
2018-10-18 12:36:37.6333 Amazon.KinesisTap.Hosting.LogManager INFO Random source id
 MyRandomSource started with rate 10000.
2018-10-18 12:36:47.8084 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":14}.
2018-10-18 12:36:57.6339 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":5}.
2018-10-18 12:37:07.6490 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":9}.
2018-10-18 12:37:17.6494 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":47}.
2018-10-18 12:37:27.6520 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":25}.
2018-10-18 12:37:37.6676 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":21}.
2018-10-18 12:37:47.6688 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":29}.
2018-10-18 12:37:57.6700 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":22}.
2018-10-18 12:38:07.6838 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":32}.
2018-10-18 12:38:17.6848 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":12}.
2018-10-18 12:38:27.6866 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":46}.
2018-10-18 12:38:37.6880 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":48}.
2018-10-18 12:38:47.6893 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":39}.
2018-10-18 12:38:57.6906 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":18}.
```

```
2018-10-18 12:39:07.6995 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":6}.
2018-10-18 12:39:17.7004 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":0}.
2018-10-18 12:39:27.7021 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":3}.
2018-10-18 12:39:37.7023 Amazon.KinesisTap.Hosting.LogManager INFO Null sink
 MyNullSink received {"RandomValue":19}.
```

AWS 서비스에 액세스하는 싱크를 만드는 경우 유용한 기본 클래스가 있습니다. 사용하는 싱크의 경우`AWSBufferedEventSink`기본 클래스에 대한 자세한 내용은`Amazon.KinesisTap.AWS\CloudWatchLogsSink.cs`를 Windows용 Kinesis 에이전트의 소스 코드에 입력하십시오.

Microsoft Windows용 Amazon Kinesis Agent 설명서 설명서 설명서 설명서 설명서 설명서 설명서

API 버전: 2018-10-15

다음 표에 변경 사항에 대한 설명이 나와 있습니다.마이크로소프트 윈도우용 Amazon Kinesis 에이전트 사용 설명서(이 문서) 를 참조하십시오.

update-history-change	update-history-description	update-history-date
주요 설명서 업데이트	MSI 설치에 대한 지침을 추가했습니다. 디렉터리 원본 구성을 업데이트하고 창 세븐로그 폴링소스가 추가되었습니다. 싱크 구성의 경우 로컬 파일 시스템 동기화 구성, ProfileRefreshingAWScredentialProvider, 텍스트 장식, 싱크 속성의 변수 확인, 싱크에 대한 STS 리전 엔드포인트 구성, VPC 엔드포인트 구성, 대체 프록시 서버 구성에 대한 정보가 추가되었습니다. 파이프의 경우 구성 속성이 추가되었습니다.	20121년 23월 2일
설명서 업데이트	Amazon S3 위치 사양이 대/소문자를 구분함을 나타내기 위해 항목이 업데이트되었습니다.	2018년 11월 7일
초기 릴리스, 버전 1.0.0.115	Windows용 Kinesis 에이전트 사용 설명서의 첫 번째 릴리스입니다.	2018년 11월 5일

AWS 용어집

For the latest AWS terminology, see the AWS glossary in the AWS General Reference.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.