aws

Benutzerhandbuch

Amazon Kinesis Agent für Microsoft Windows

Amazon Kinesis Agent für Microsoft Windows: Benutzerhandbuch

Table of Contents

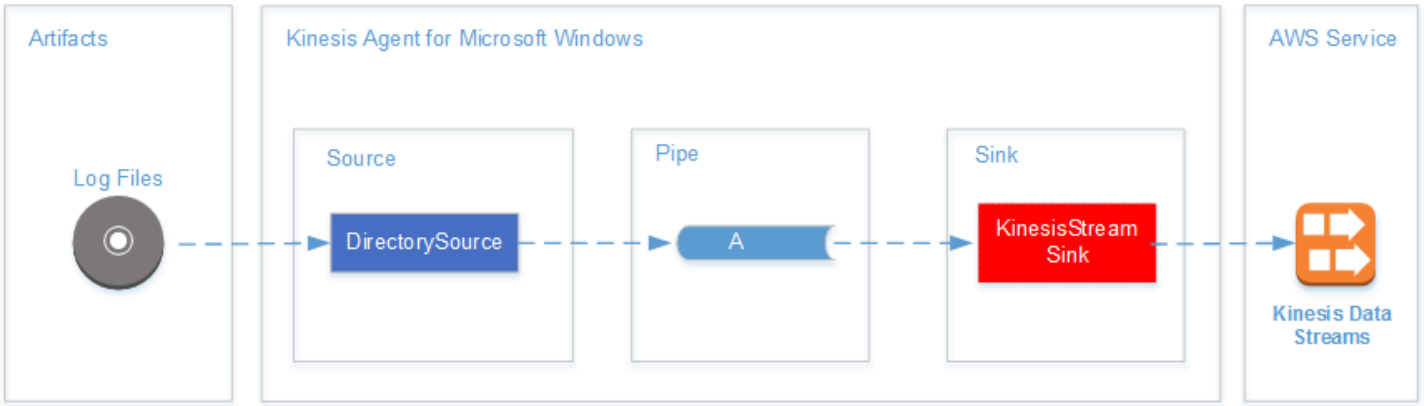
Was ist der Amazon Kinesis Agent für Microsoft Windows?

Amazon Kinesis Agent für Microsoft Windows (Kinesis Agent für Windows) ist ein konfigurierbarer und erweiterbarer Agent. Er wird auf der Flotten von Windows Desktop-Computern und Servern entweder lokal oder in der AWS Cloud ausgeführt. Der Kinesis Agent für Windows sammelt, analysiert, transformiert und streamt Protokolle, Ereignisse und Metriken effizient und zuverlässig für verschiedene AWS -Services, einschließlichKinesis Data Streams,Kinesis Data Firehose,Amazon CloudWatch, undCloudWatch-Protokolle.

Über diese Services können Sie die Daten dann mithilfe einer Vielzahl anderer AWS-Services speichern, analysieren und visualisieren, einschließlich der folgenden:

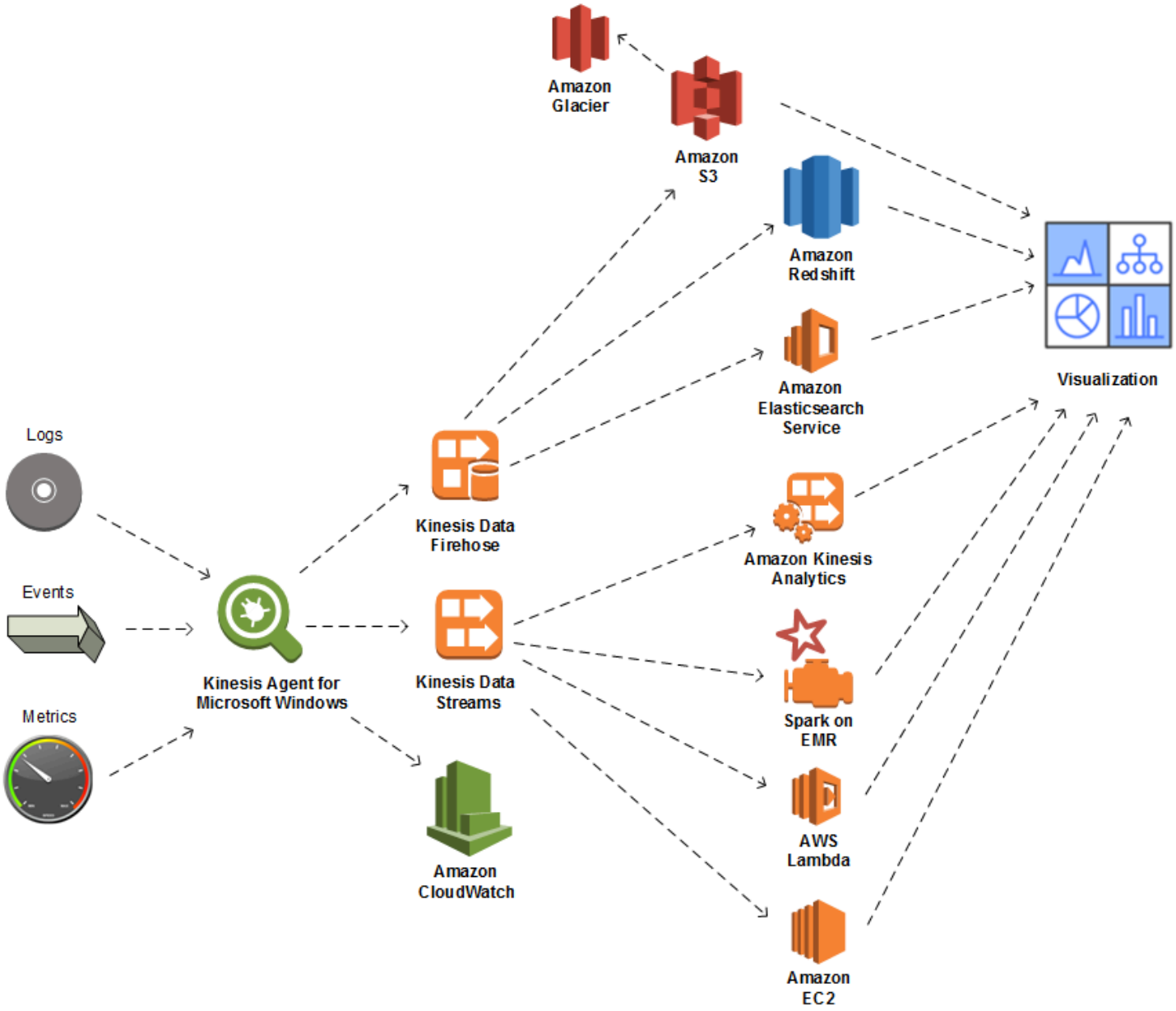
- Amazon Simple Storage Service (Amazon S3)
- Amazon Redshift
- Amazon Elasticsearch Service (Amazon ES)
- Analyse von Kinesis Data Analytics
- Amazon QuickSight
- Amazon Athena
- Kibana

Das folgende Diagramm veranschaulicht eine einfache Konfiguration von Kinesis Agent für Windows, mit der Protokolldateien an Kinesis Data Streams gestreamt werden.



Weitere Informationen zu Quellen, Pipes und Senken finden Sie unter Konzepte von Amazon Kinesis Agent für Microsoft Windows.

Das folgende Diagramm veranschaulicht einige der Möglichkeiten für die Erstellung von benutzerdefinierten Echtzeit-Daten-Pipelines unter Verwendung von Stream-Verarbeitungs-Frameworks. Zu diesen Frameworks gehören Kinesis Data Analytics, Apache Spark auf Amazon EMR und AWS Lambda.



Themen

Über AWS

Amazon Web Services (AWS) ist eine Sammlung von digitalen Infrastruktur-Services, die Sie bei der Anwendungsentwicklung nutzen können. Die Services umfassen Computing, Speicher, Datenbank, Analyse und Anwendungssynchronisation (Messaging und Queuing). AWS verwendet ein Servicemodell mit nutzungsabhängiger Bezahlung. Es werden Ihnen aber nur die Services in Rechnung gestellt, die Sie — oder Ihre Anwendungen — nutzen. Um seine Services für Prototyping und Tests besser zugänglich zu machen, bietet AWS darüber hinaus ein kostenloses Nutzungskontingent. Im Rahmen dieses Kontingents sind die Services bis zu einem bestimmten Nutzungsumfang kostenlos. Weitere Informationen über AWS Kosten und denKostenloses Kontingent, finden Sie unter denErste Schritte mit dem Ressourcencenter. Um ein AWS-Konto zu erstellen, öffnen Sie die AWS-Homepage und registrieren Sie sich für eines.

Welche Möglichkeiten bietet Kinesis Agent für Windows?

Der Kinesis Agent für Windows bietet die folgenden Funktionen und Möglichkeiten:

Erfassen von Protokollen, Ereignissen und Metrikdaten

Der Kinesis Agent für Windows sammelt, analysiert, transformiert und streamt Protokolle, Ereignisse und Metriken von Flotten von Servern und Desktops für einen oder mehrere AWS -Services. Die von den Services empfangene Nutzlast kann sich in einem anderen Format als das der ursprünglichen Quelle befinden. Beispiel: Ein Protokoll befindet sich möglicherweise in einem bestimmten Textformat (z. B. Syslog-Format) auf einem Server. Der Kinesis Agent für Windows kann Text erfassen und analysieren und ihn optional in das JSON-Format transformieren, so beispielsweise vor dem Streamen an AWS. Dies ermöglicht eine einfachere Verarbeitung durch einige AWS-Services, die JSON nutzen. Über Kinesis Data Streams gestreamte Daten können fortlaufend von Kinesis Data Analytics verarbeitet werden, um zusätzliche Metriken und aggregierte Metriken zu erstellen, wodurch wiederum Live-Dashboards betrieben werden können. Sie können die Daten mithilfe einer Vielzahl von AWS -Services (z. B. Amazon S3) speichern, je nachdem, wie die Daten in einer Downstream-Daten-Pipeline verwendet werden.

Integrieren in AWS-Services

Sie können Kinesis Agent für Windows so konfigurieren, dass Protokolldateien, Ereignisse und Metriken an verschiedene AWS -Services gesendet werden:

- Kinesis Data Firehose— Speichern Sie gestreamte Daten einfach in Amazon S3, Amazon Redshift, Amazon ES oderSplunkzur weiteren Analyse.
- Kinesis Data Streams— Verarbeiten Sie gestreamte Daten mithilfe benutzerdefinierter Anwendungen, die in Kinesis Data Analytics oder Apache Spark aufAmazon EMR. Oder verwenden Sie benutzerdefinierten Code, der aufAmazon EC2-Instances oder benutzerdefinierte serverlose Funktionen, die inAWS Lambda.
- CloudWatch— Zeigen Sie gestreamte Metriken in Diagrammen an, die Sie zu Dashboards kombinieren können. Legen Sie anschließend CloudWatch Alarme fest, die durch Metrikwerte ausgelöst werden, die voreingestellte Schwellenwerte überschreiten.
- CloudWatch-Protokolle— Speichern Sie gestreamte Protokolle und Ereignisse und zeigen Sie sie in der AWS Management Console an und durchsuchen Sie sie oder verarbeiten Sie sie weiter nachgelagert in einer Datenpipeline.

Schnelles Installieren und Konfigurieren

Sie können Kinesis Agent für Windows in nur wenigen Schritten installieren und konfigurieren. Weitere Informationen finden Sie unter Installieren von Kinesis Agent für Windows und Konfigurieren von Amazon Kinesis Agent für Microsoft Windows. Eine einfache deklarative Konfigurationsdatei gibt Folgendes an:

- Die Quellen und Formate der zu sammelnden Protokollen, Ereignisse und Metriken.
- Die auf die gesammelten Daten anzuwendenden Transformationen. Zusätzliche Daten können eingeschlossen und vorhandene Daten können transformiert und gefiltert werden.

- Die Ziele, an die die endgültigen Daten per Streaming übertragen werden, sowie Pufferung, Sharding und Formatierung für die Streaming-Nutzlasten.

Der Kinesis Agent für Windows verfügt über integrierte Parser für Protokolldateien, die für gängige Microsoft Enterprise Services erstellt werden, wie z. B.:

- Microsoft Exchange
- SharePoint
- Active Directory-Domänencontroller
- DHCP-Server

Keine fortlaufende Verwaltung

Der Kinesis Agent für Windows passt sich automatisch an verschiedene Situationen an, ohne dass Daten verloren gehen. Dazu gehören Protokollrotation, Wiederherstellung nach einem Neustart und temporäre Netzwerk- oder Service-Unterbrechungen. Sie können Kinesis Agent für Windows so konfigurieren, dass die Aktualisierung auf eine neue Version automatisch durchgeführt wird. In keiner dieser Situationen ist ein Benutzereingriff erforderlich.

Erweitern mithilfe von Open Architecture

Wenn die deklarative Funktionen und integrierten Plug-Ins für die Überwachung von Server- oder Desktop-Systemen nicht ausreichen, können Sie Kinesis Agent für Windows durch Erstellen von Plug-Ins erweitern. Neue Plug-Ins ermöglichen neue Quelle und Ziele für Protokolle, Ereignisse und Metriken. Der Quellcode für Kinesis Agent für Windows ist unterhttps://github.com/awslabs/kinesis-agent-windows.

Benefits

Der Kinesis Agent für Windows führt die anfängliche Datenerfassung, Transformation und das Streaming für Protokolle, Ereignisse und Metriken für Daten-Pipelines durch. Das Erstellen solcher Daten-Pipelines bringt zahlreiche Vorteile:

Datenanalyse und -visualisierung

Die Integration von Kinesis Agent für Windows in Kinesis Data Firehose und seine Transformationsfunktionen erleichtern die Integration in verschiedene analytische und Visualisierungs-Services:

- Amazon QuickSight— Ein cloudbasierter BI-Dienst, der aus vielen verschiedenen Quellen aufgenommen werden kann. Kinesis Agent für Windows kann Daten transformieren und über Kinesis Data Firehose an Amazon S3 und Amazon Redshift streamen. Dieser Prozess ermöglicht die Gewinnung tiefer Einblicke anhand der Daten mithilfe von Amazon QuickSight - Visualisierungen.
- Athena— Ein interaktiver Abfragedienst, der SQL-basiertes Abfragen von Daten ermöglicht. Kinesis Agent für Windows kann Daten über Kinesis Data Firehose transformieren und an Amazon S3 streamen. Athena kann dann interaktiv SQL-Abfragen für diese Daten ausführen, um Protokolle und Ereignisse schnell zu überprüfen und zu analysieren.
- Kibana— Ein Open-Source-Datenvisualisierungstool. Kinesis Agent für Windows kann Daten über Kinesis Data Firehose transformieren und an Amazon ES streamen. Sie können dann mithilfe von Kibana mit diesen Daten arbeiten. Erstellen und öffnen Sie verschiedene Visualisierungen, einschließlich Histogramme, Liniendiagramme, Tortendiagramme, Heatmaps und raumbezogenen Grafiken.

Security

Eine Protokoll- und Ereignisdaten-Analyse-Pipeline, die Kinesis Agent für Windows enthält, kann Sicherheitsverletzungen in Organisationen erkennen und vor ihnen warnen, wodurch Sie Angriffe blockieren oder beenden können.

Anwendungsleistung

Der Kinesis Agent für Windows kann Protokolle, Ereignisse und Metrikdaten über die Leistung von Anwendungen und Services sammeln. Diese Daten könne dann von einer vollständigen Daten-Pipeline analysiert werden. Anhand dieser Analyse können Sie die Leistung und Zuverlässigkeit Ihrer Anwendungen und Services durch Erkennen von Fehlern und Erstellen entsprechender Berichte verbessern, die andernfalls möglicherweise nicht offensichtlich sind. Sie können beispielsweise signifikante Änderungen der Ausführungszeiten von Service-API-Aufrufen erkennen. Bei Bezug auf eine Bereitstellung hilft diese Funktion Ihnen, neue Performance-Probleme mit Services in Ihrem Besitz aufzudecken und zu beseitigen.

Serviceoperationen

Eine Daten-Pipeline kann die gesammelten Daten zur Vorhersage potenzieller Betriebsprobleme und für weitere Informationen darüber, wie sich Service-Ausfälle vermeiden lassen, analysieren. Sie können z. B. Protokolle, Ereignisse und Metriken analysieren, um aktuelle und projizierte Kapazitätsnutzung zu bestimmen. So können Sie zusätzliche Kapazitäten online bringen, bevor Benutzer der Services betroffen sind. Wenn ein Service-Ausfall auftritt, können Sie die Daten analysieren, um die Auswirkungen während des Ausfalls auf Kunden zu bestimmen.

Auditing

Eine Daten-Pipeline kann die Protokolle, Ereignisse und Metriken verarbeiten, die Kinesis Agent für Windows erfasst und transformiert. Sie können diese verarbeiteten Daten dann mit verschiedenen AWS-Services prüfen. Beispielsweise kann Kinesis Data Firehose einen Daten-Stream von Kinesis Agent für Windows empfangen, der die Daten in Amazon S3 speichert. Sie können diese Daten dann prüfen, indem Sie interaktive SQL-Abfragen mithilfe von Athena ausführen.

Archiving

Oft sind die wichtigsten operativen Daten die vor Kurzem erfassten Daten. Die Analyse von Daten, die bezüglich Anwendungen und Services über mehrere Jahre hinweg gesammelt wurden, kann auch nützlich sein, z. B. für die langfristige Planung. Das Speichern großer Datenmengen kann teuer sein. Kinesis Agent für Windows kann Daten über Kinesis Data Firehose in Amazon S3 erfassen, transformieren und speichern. Daher gilt,Amazon S3 Glacierzur Verfügung, um die Kosten für die Archivierung älterer Daten zu reduzieren.

Alerting

Kinesis Agent für Windows streamt Metriken zu CloudWatch. Im Gegenzug können Sie CloudWatch Alarme erstellen, um über eine -Benachrichtigung zu sendenAmazon Simple Notification Service (Amazon SNS), wenn eine Metrik konsistent gegen einen bestimmten Schwellenwert verstößt. Auf diese Weise werden Technikern betriebliche Probleme mit ihren Anwendungen und Services stärker bewusst.

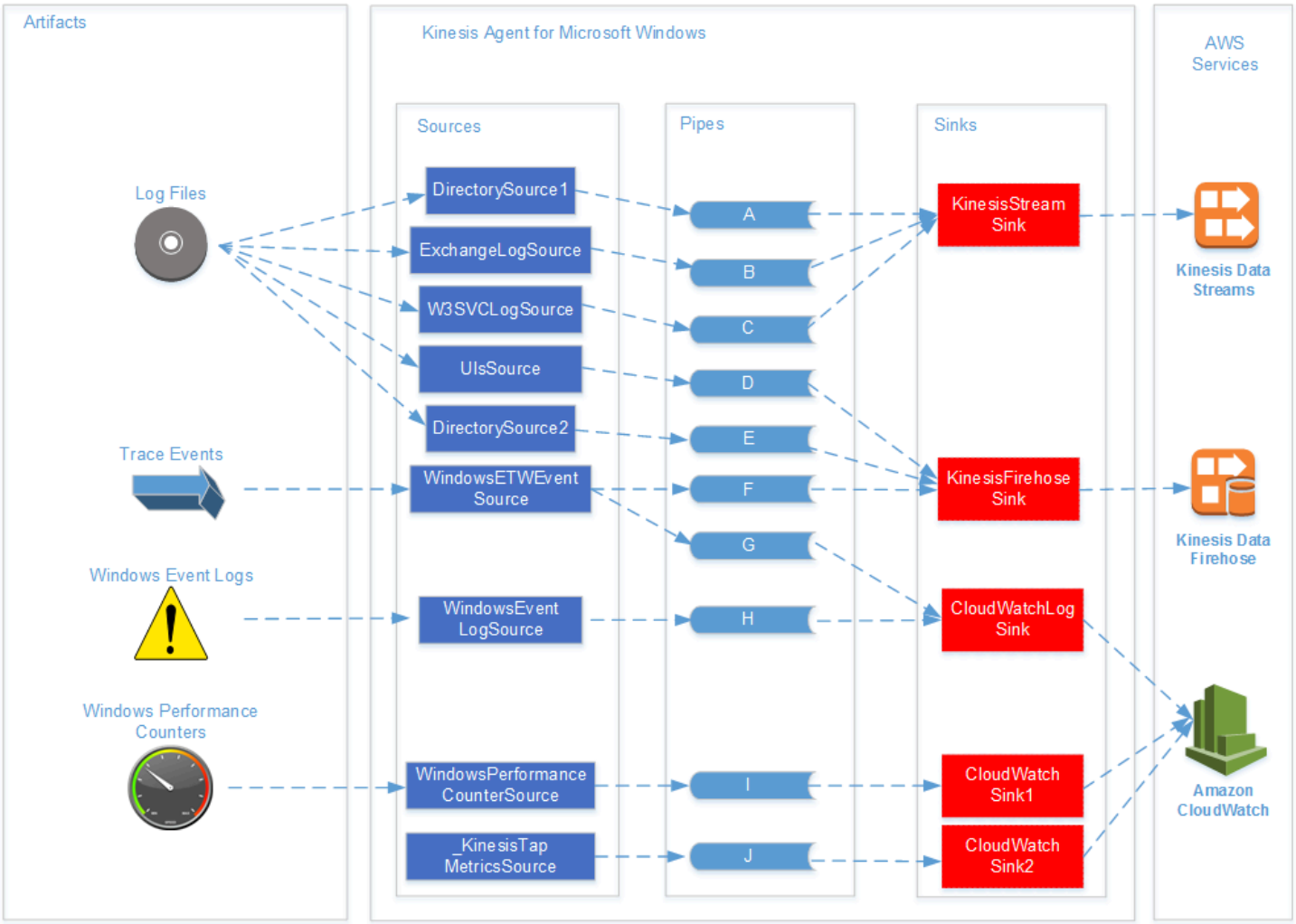
Erste Schritte mit Kinesis Agent für Windows

Um weitere Informationen über Kinesis Agent für Windows zu erhalten, empfehlen wir, dass Sie mit den folgenden Abschnitten beginnen:

- Konzepte von Amazon Kinesis Agent für Microsoft Windows
- Erste Schritte mit Amazon Kinesis Agent für Microsoft Windows

Konzepte von Amazon Kinesis Agent für Microsoft Windows

Ein Verständnis der wichtigsten Konzepte von Amazon Kinesis Agent für Microsoft Windows (Kinesis Agent für Windows) kann Ihnen das Sammeln und Streamen von Daten in Desktop- und Server-Flotten an die restliche Daten-Pipeline zwecks Verarbeitung erleichtern.



Dieses Diagramm einer Daten-Pipeline veranschaulicht die folgenden Komponenten und Prozesse:

Server und Desktops haben Artefakte wie Protokolldateien, Ereignisse und Metriken, die von einem oder mehreren Kinesis Agent für Windows gesammelt werden.Quellen. Die Daten können optional aus einem Flat-Datei-Textformat in ein Objekt transformiert werden.

Die Daten (entweder in Objekt- oder Textform) können dann in einen oder mehrere Kinesis Agent für Windows fließen.Pipes. Eine Pipe verbindet eine Quelle mit einem Kinesis Agent für Windowssink. Die Pipe kann optional unnötige Daten herausfiltern.

Eine Senke kann optional die in Objekte analysierten Daten in das JSON- oder XML-Format transformieren. Die Senke sendet die Daten an einen bestimmten AWS -Service, wie z. B. Kinesis Data Streams, Kinesis Data Firehose oder Amazon CloudWatch.

Unter Verwendung mehrerer Pipes kann eine einzelne Quelle dieselben Daten zu mehreren Senken senden (so z. B. die Pipes F und G im Diagramm). Unter Verwendung mehrerer Pipes können unterschiedliche Quellen Daten an eine einzelne Senke streamen (so z. B. die Pipes A, B und C im Diagramm). Es ist auch möglich, mit mehreren Pipes Daten aus mehreren Senken an mehrere Quellen zu streamen. Es gibt verschiedene Typen von Quellen, Senken und Pipes und es können mehrere Quellen, Senken oder Pipes desselben Typs vorhanden sein.

Beispiele für Konfigurationsdateien, die Quellen, Senken und Pipes deklarieren, finden Sie unter Konfigurationsbeispiele für Kinesis Agent für Windows.

Themen

Daten-Pipelines

ADaten-Pipelinewird verwendet, um Alarme für Anwendungen und Dienste zu sammeln, zu verarbeiten, zu visualisieren und möglicherweise zu generieren. Kinesis Agent für Windows passt zu Beginn in Datenpipelines, in denen Protokolle, Ereignisse und Metriken von Flotten von Desktop-Computern oder Servern gesammelt werden. Kinesis Agent für Windows streamt die gesammelten Daten an die verschiedenen AWS -Services, die die restliche Daten-Pipeline bilden. Eine Daten-Pipeline dient einem bestimmten Zweck, wie z. B. der Visualisierung des Zustands eines bestimmten Service in Echtzeit, um Techniker bei der effektiveren Nutzung dieses Service zu unterstützen. Eine Pipeline der Service-Zustandsdaten kann beliebige der folgenden Zwecke erfüllen:

- Sie kann Techniker auf Probleme aufmerksam machen, bevor die mit den Services gemachten Erfahrungen der Kunden durch diese Probleme beeinträchtigt werden.
- Sie kann Techniker durch Anzeigen von Nutzungstrends der Ressourcen zu effizientem Kostenmanagement befähigen. Anhand dieser Trends können sie Ressourcen-Bestände entsprechend korrigieren oder sogar Auto Scaling-Szenarien implementieren.

- Sie kann Einblicke in die Ursachen der Probleme gewähren, die von Kunden des Service gemeldet werden. Auf diese Weise lassen sich diese Probleme schneller lösen und Support-Kosten reduzieren.

Ein Schritt-für-Schritt-Beispiel für die Erstellung einer Daten-Pipeline mithilfe von Kinesis Agent für Windows finden Sie unterTutorial: Streamen von JSON-Protokolldateien mit Kinesis Agent für Windows zu Amazon S3.

Sources

Ein Kinesis Agent für Windowssourcesammelt Protokolle, Ereignisse oder Metriken. Eine Quelle sammelt Daten einer bestimmten Art von einem bestimmten Produzenten dieser Daten basierend auf dem Typ der Quelle. Beispiel: Der Typ `DirectorySource` sammelt Protokolldateien aus bestimmten Verzeichnissen im Dateisystem. Wenn die Daten noch nicht strukturiert sind (wie bei bestimmten Arten von Protokolldateien), kann eine Quelle bei der Analyse der Daten in einer strukturierten Form nützlich sein. Jede Quelle entspricht einer bestimmtenQuell-Deklarim Kinesis Agent für Windows`appsettings.json`-Konfigurationsdatei. Die Quell-Deklaration bietet wesentliche Details zum Konfigurieren der Quelle, um die Quelle entsprechend den spezifischen Datenerfassungsanforderungen anzupassen. Welche Details konfiguriert werden können, ist je nach Quelltyp verschieden. Beispiel: Bei dem Quelltyp `DirectorySource` muss das Verzeichnis mit den Protokolldateien angegebenen werden.

Weitere Informationen zu Quelltypen und Quell-Deklarationen finden Sie unter Quell-Deklarationen.

Sinks

Ein Kinesis Agent für WindowssinkEine -Quelle streamt die von einer -Quelle von Kinesis Agent für Windows gesammelten Daten an einen von mehreren möglichen AWS -Services, die die restliche Daten-Pipeline bilden. Jede Senke entspricht einer bestimmtenDeklarationim Kinesis Agent für Windows`appsettings.json`-Konfigurationsdatei. Die Senken-Deklaration bietet wesentliche Details zum Konfigurieren der Senke, um die Senke entsprechend den spezifischen Daten-Streaming-Anforderungen anzupassen. Welche Details konfiguriert werden können, ist je nach Senkentyp verschieden. Beispiel: Bei einigen Senken-Typen kann mit einer Senken-Deklaration ein bestimmtes `Format` der Serialisierung für Daten angegeben werden, die für sie bereitgestellt werden. Wenn diese Option in der Senken-Deklaration angegeben wird, wird zuerst eine Serialisierung der erfassten Daten durchgeführt, bevor die Daten an den der Senke zugeordneten AWS-Service gestreamt werden.

Weitere Informationen zu Senken-Typen und Senken-Deklarationen finden Sie unter Senken-Deklarationen.

Pipes

Ein Kinesis Agent für WindowsPipeEine -Quelle verbindet die Ausgabe einer Kinesis Agent für Windows-Quelle mit der Eingabe einer -Senke von Kinesis Agent für Windows. Während die Daten durch die Pipe strömen, können sie optional transformiert werden. Jede Pipe entspricht einer bestimmten Pipe-Deklaration in Kinesis Agent für Windows`appsettings.json`-Konfigurationsdatei. Die Pipe-Deklaration bietet wesentliche Details für die Konfiguration der Pipe, wie z. B. die Quelle und Senke für die Pipe.

Weitere Informationen zu Pipe-Typen und Pipe-Deklarationen finden Sie unter Pipe-Deklarationen.

Erste Schritte mit Amazon Kinesis Agent für Microsoft Windows

Sie können Amazon Kinesis -Agenten für Microsoft Windows (Kinesis -Agenten für Windows) zum Sammeln, Analysieren, Transformieren und Streamen von Protokollen, Ereignissen und Metriken aus Ihrer Windows-Flotte für verschiedene AWS-Services verwenden. Die folgenden Informationen enthalten Voraussetzungen und Schritt-für-Schritt-Anweisungen für die Installation und Konfiguration von Kinesis -Agenten für Windows.

Themen

Prerequisites

Stellen vor dem Installieren von Kinesis Agent für Windows sicher, dass die folgenden Voraussetzungen erfüllt sind:

- Vertrautheit mit Kinesis Agent für Windows-Konzepten Weitere Informationen finden Sie unter Konzepte von Amazon Kinesis Agent für Microsoft Windows.
- Ein AWS Konto für die Verwendung der verschiedenen AWS-Dienste im Zusammenhang mit Ihrer Datenpipeline. Weitere Informationen zum Erstellen und Konfigurieren eines AWS-Kontos finden Sie unter Einrichten eines AWS-Kontos.
- Microsoft. NET Framework 4.6 oder höher auf jedem Desktop oder Server, auf dem Kinesis Agent für Windows ausgeführt wird. Weitere Informationen finden Sie unter Install the .NET Framework for developers in der Microsoft .NET-Dokumentation.

  Um die neueste Version des .NET Framework zu bestimmen, das auf einem Desktop oder Server installiert ist, verwenden Sie das folgende PowerShell-Skript:

```
[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
```

- Die Streams, an die Sie Daten von Kinesis Agent für Windows senden möchten (sofern Amazon Kinesis Data Streams verwendet wird). Erstellen Sie die Streams mit demKinesis Data Streams Konsole, dieAWS CLI, oderAWS-Tools für Windows PowerShell. Weitere Informationen finden Sie unterErstellen und Aktualisieren von Daten-StreamsimAmazon Kinesis Data Streams Entwicklerhandbuch.
- Die Firehose-Bereitstellungs-Streams, an die Sie Daten von Kinesis Agent für Windows senden möchten (sofern Amazon Kinesis Data Firehose verwendet wird). Erstellen Sie Übermittlungsströme mit demKinesis Data Firehose Konsole, dieAWS CLI, oderAWS-Tools für Windows PowerShell. Weitere Informationen finden Sie unter Erstellen eines Amazon Kinesis Data Firehose-Bereitstellungs-Streams im Amazon Kinesis Data Firehose-Entwicklerhandbuch.

Einrichten eines AWS-Kontos

Wenn Sie kein AWS-Konto haben, führen Sie die folgenden Schritte zum Erstellen eines Kontos durch.

Für ein AWS-Konto registrieren Sie sich wie folgt:

1. Öffnen Sie https://portal.aws.amazon.com/billing/signup.
2. Folgen Sie den Onlineanweisungen.

   Der Anmeldeprozess beinhaltet auch einen Telefonanruf und die Eingabe eines Verifizierungscodes über die Telefontastatur.

So erstellen Sie einen Administratorbenutzer für sich selbst und fügen ihn einer Administratorengruppe hinzu (Konsole)

1. Melden Sie sich bei der IAM-Konsole als Kontoinhaber an, indem Sie Root user (Stammbenutzer) auswählen und die E-Mail-Adresse Ihres AWS-Kontos eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

> **Note**
>
> Wir empfehlen nachdrücklich, die bewährten Methoden mit dem**Administrator**IAMBenutzer, der die Anmeldeinformationen des Stammbenutzers an einem sicheren Ort ablegen. Melden Sie sich als Stammbenutzer an, um einige Konto- und Service-Verwaltungsaufgaben durchzuführen.

2. Wählen Sie im Navigationsbereich Users und dann Add User aus.
3. Geben Sie unter Benutzername **Administrator** als Benutzernamen ein.
4. Markieren Sie das Kontrollkästchen neben AWS Management Console access (Zugriff auf AWS-Managementkonsole). Wählen Sie dann Custom password (Benutzerdefiniertes Passwort) aus und geben Sie danach Ihr neues Passwort in das Textfeld ein.
5. (Optional) Standardmäßig erfordert AWS, dass der neue Benutzer bei der ersten Anmeldung ein neues Passwort erstellt. Sie können das Kontrollkästchen neben User must create a new password at next sign-in (Benutzer muss bei der nächsten Anmeldung ein neues Passwort erstellen) deaktivieren, um dem neuen Benutzer zu ermöglichen, sein Kennwort nach der Anmeldung zurückzusetzen.
6. Klicken Sie aufWeiter: Berechtigungen
7. Wählen Sie unter Set permissions (Berechtigungen festlegen) die Option Add user to group (Benutzer der Gruppe hinzufügen) aus.
8. Wählen Sie Create group (Gruppe erstellen) aus.
9. Geben Sie im Dialogfeld Create group (Gruppe erstellen) unter Group name (Gruppenname) **Administrators** ein.
10. Klicken Sie aufFilterrichtlinienWählen Sie und dann ausAWS verwaltet - Auftragsfunktion, um den Tabelleninhalt zu filtern.
11. Aktivieren Sie in der Richtlinienliste das Kontrollkästchen AdministratorAccess. Wählen Sie dann Create group aus.

> **Note**
>
> Sie müssen IAM-Benutzer- und Rollenzugriff auf Billing aktivieren, bevor Sie die `AdministratorAccess`-Berechtigungen für den Zugriff auf die AWS Billing and Cost Management-Konsole verwenden können. Befolgen Sie hierzu die Anweisungen in Schritt 1 des Tutorials zum Delegieren des Zugriffs auf die Abrechnungskonsole.

12. Kehren Sie zur Gruppenliste zurück und aktivieren Sie das Kontrollkästchen der neuen Gruppe. Möglicherweise müssen Sie Refresh auswählen, damit die Gruppe in der Liste angezeigt wird.
13. Klicken Sie aufWeiter: Tags.
14. (Optional) Fügen Sie dem Benutzer Metadaten hinzu, indem Sie Tags als Schlüssel-Wert-Paare anfügen. Weitere Informationen zur Verwendung von Tags in IAM finden Sie unter Tagging von IAM-Entitäten im IAM-Benutzerhandbuch.
15. Klicken Sie aufWeiter: PrüfenKlicken Sie auf, um eine Liste der Gruppenmitgliedschaften anzuzeigen, die dem neuen Benutzer hinzugefügt werden soll. Wenn Sie bereit sind, fortzufahren, wählen Sie Create user (Benutzer erstellen) aus.

Mit diesen Schritten können Sie weitere Gruppen und Benutzer erstellen und Ihren Benutzern Zugriff auf Ihre AWS-Kontoressourcen gewähren. Weitere Informationen dazu, wie Sie die Berechtigungen eines Benutzers auf bestimmte AWS-Ressourcen mithilfe von Richtlinien beschränken, finden Sie unter Zugriffsverwaltung und Beispielrichtlinien.

So registrieren Sie sich bei AWS und erstellen ein Administratorkonto

1. Wenn Sie noch kein AWS Konto haben, öffnen Siehttps://aws.amazon.com/. Wählen Sie Create an AWS Account (Erstellen eines AWS-Kontos) und befolgen Sie die Online-Anweisungen.

   Der Anmeldeprozess beinhaltet auch einen Telefonanruf und die Eingabe einer PIN über die Telefontastatur.
2. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
3. Wählen Sie im Navigationsbereich Groups (Gruppen) und dann Create New Group (Neue Gruppe erstellen) aus.
4. Geben Sie für Group Name (Gruppenname) einen Namen für die Gruppe ein, z. B. **`Administrators`**. Wählen Sie dann Next Step (Nächster Schritt) aus.
5. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben der Richtlinie AdministratorAccess. Über das Menü Filter (Filtern) und das Feld Search (Suchen) können Sie die Liste filtern.
6. Klicken Sie auf Next Step. Wählen Sie Create Group (Gruppe erstellen). Ihre neue Gruppe wird unter Group Name (Gruppenname) angezeigt.
7. Wählen Sie im Navigationsbereich Users und dann Create New Users aus.

8. Geben Sie im Feld 1 einen Benutzernamen ein, deaktivieren Sie das Kontrollkästchen neben Generate an access key for each user (Zugriffsschlüssel für jeden Benutzer generieren) und wählen Sie dann Create (Erstellen).
9. Wählen Sie in der Benutzerliste den Namen – nicht das Kontrollkästchen – für den Benutzer aus, den Sie gerade erstellt haben. Sie können über das Feld Search (Suchen) nach dem Benutzernamen suchen.
10. Wählen Sie die Registerkarte Groups (Gruppen) und anschließend die Option Add User to Groups (Benutzer zu Gruppen hinzufügen) aus.
11. Aktivieren Sie das Kontrollkästchen neben der Administratorgruppe und wählen Sie dann Add to Groups (Zu Gruppen hinzufügen) aus.
12. Wechseln Sie zur Registerkarte Security Credentials. Wählen Sie unter Sign-In Credentials die Option Manage Password aus.
13. Klicken Sie auf Assign a custom password (Benutzerdefiniertes Passwort zuweisen), geben Sie ein Passwort in die Felder Password (Passwort) und Confirm Password (Passwort bestätigen) ein und klicken Sie dann auf Apply (Anwenden).

Installieren von Kinesis Agent für Windows

Es gibt drei Möglichkeiten, wie Sie Kinesis Agent für Windows unter Windows installieren können:

- Installieren Sie mithilfe von MSI (einem Windows-Installer-Paket).
- Installieren Sie vonAWS Systems ManagerEin Satz von Services für die Verwaltung von Servern und Desktops.
- Ausführen eines PowerShell -Skripts.

> **Note**
>
> Die folgenden Anweisungen verwenden gelegentlich die Begriffe KinesisTap und AWSKinesisTap. Diese Wörter haben dieselbe Bedeutung wie Kinesis Agent für Windows. Sie müssen jedoch bei der Ausführung dieser Anweisungen unverändert angeben.

Installieren von Kinesis Agent für Windows mit MSI

Sie können das neueste Kinesis Agent für Windows MSI-Paket vomKinesis-Agent-Windows-Repository auf GitHub. Nachdem Sie die MSI heruntergeladen haben, starten Sie sie mithilfe von Windows und folgen Sie den Anweisungen des Installationsprogramms. Nach der Installation können Sie wie jede Windows-Anwendung deinstallieren.

Sie können aber auch diemsiexecSo können Sie an der Windows-Eingabeaufforderung automatisch installieren, die Protokollierung aktivieren und deinstallieren, wie in den folgenden Beispielen dargestellt. Ersetzen*`AWSKinesisTap.1.1.216.4.msi with the appropriate version of Kinesis Agent for Windows for your application.`*

So installieren Sie Kinesis Agent für Windows automatisch:

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

So protokollieren Sie Installationsmeldungen zur Fehlerbehebung in einer Datei namens***`logfile.log`***:

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

So deinstallieren Sie Kinesis Agent für Windows über die Eingabeaufforderung:

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

Installieren von Kinesis Agent für Windows mit AWS Systems Manager

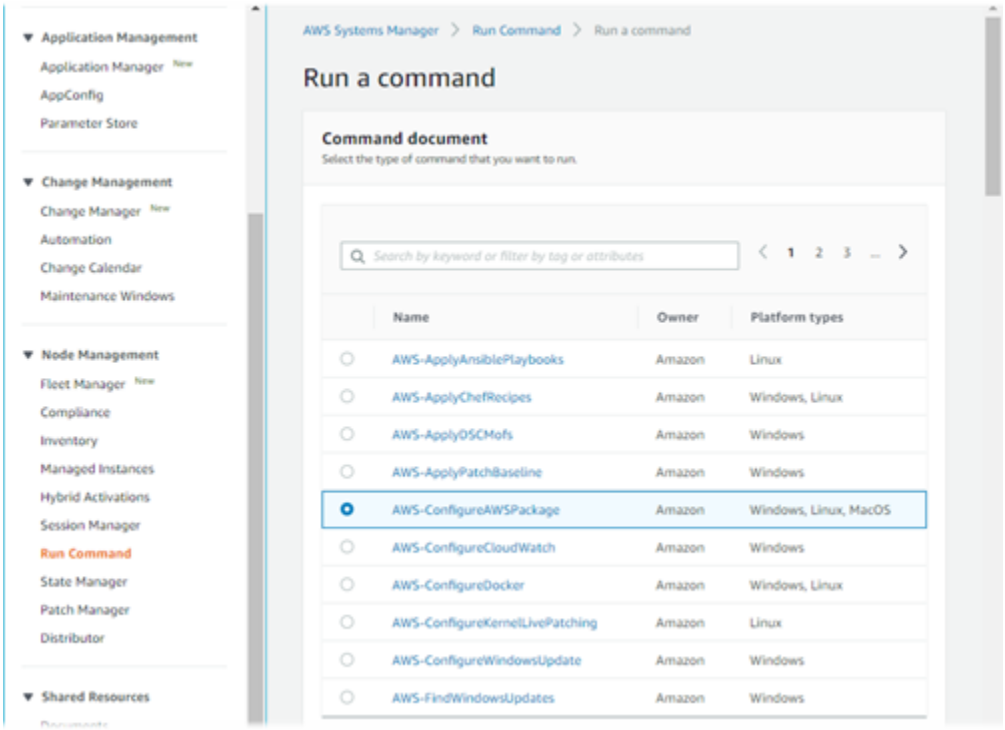
Führen Sie die folgenden Schritte aus, um Kinesis Agent für Windows mithilfe des Systems Manager Run Command zu installieren. Weitere Informationen zum Run Command finden Sie unterAWS Systems Manager Run CommandimAWS Systems Manager Benutzerhandbuch. Zusätzlich zu dem Systems Manager Run Command können Sie auch Systems ManagerWartungsfensterundState Manager, um die Bereitstellung von Kinesis Agent für Windows im Laufe der Zeit zu automatisieren.

> **Note**
>
> Die Systems Manager Installation für Kinesis Agent für Windows ist in den AWS Regionen verfügbar, die unterAWS Systems ManagerMit Ausnahme der folgenden:

- cn-north-1
- cn-northwest-1
- Alle AWS GovCloud Regionen.

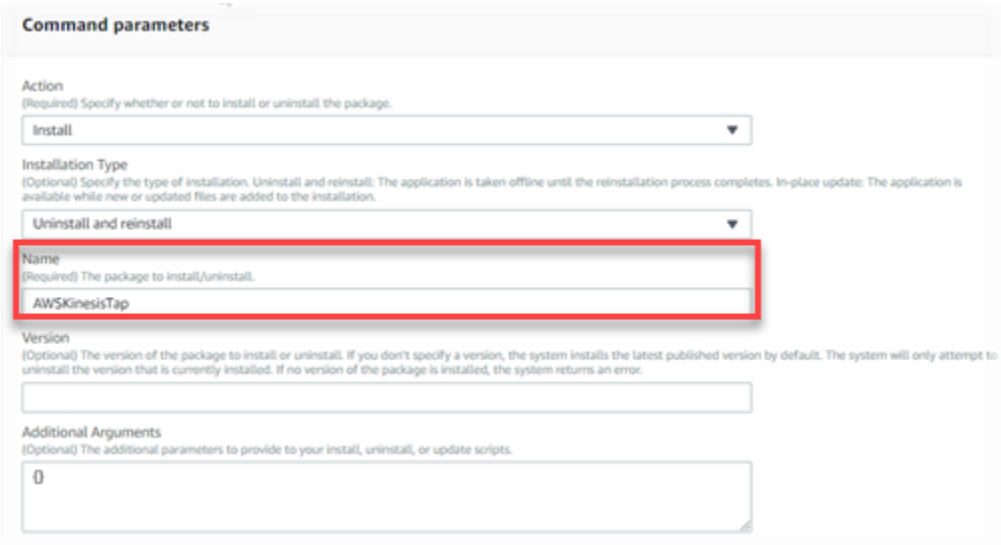
So installieren Sie Kinesis Agent für Windows mit Systems Manager

1. Stellen Sie sicher, dass Version 2.2.58.0 oder höher des SSM-Agenten auf Instances installiert ist, auf denen Sie Kinesis -Agenten für Windows installieren möchten. Weitere Informationen finden Sie unterInstallieren und Konfigurieren des SSM Agents auf Windows-InstancesimAWS Systems Manager Benutzerhandbuch.
2. Öffnen Sie die AWS Systems Manager Konsole unterhttps://console.aws.amazon.com/systems-manager/.
3. Klicken Sie im Navigationsbereich unterVerwaltung von Knoten, wählen SieRun CommandKlicken Sie auf und danach aufRun Command.
4. Über dieBefehlsdokumentaus, wählen Sie dieAWS-KonfigurationAWSPackagedocument.



5. UNDERBefehls-Parameter, fürName, enter AWSkineSiStap. Belassen Sie andere Einstellungen auf ihren Standardwerten.

> **Note**
>
> verlassenVersionGeben Sie die neueste Version des AWSkineSiStap-Pakets an. Optional können Sie eine spezifische Version eingeben, die installiert werden soll.



6. UNDERTargets (Ziele)Geben Sie die Instances an, auf denen der Befehl ausgeführt werden soll. Sie können Instanzen basierend auf Tags angeben, die mit Instanzen verknüpft sind, Sie können Instanzen manuell auswählen oder eine Ressourcengruppe angeben, die Instanzen enthält.
7. Belassen Sie alle anderen Einstellungen auf ihren Standardwerten und wählen SieFühren Sie Folgendes aus:.

Installieren von Kinesis Agent für Windows mit PowerShell

Verwenden Sie einen Texteditor, um die folgenden Befehle in eine Datei zu kopieren und als PowerShell -Skript zu speichern. Wir verwenden`InstallKinesisAgent.ps1`Im folgenden Beispiel.

```
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
}
```

Öffnen Sie ein Befehlszeilenfenster mit erhöhten Rechten. Verwenden Sie im Verzeichnis, in das die Datei heruntergeladen wurde, den folgenden Befehl, um das Skript auszuführen:

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

Um eine bestimmte Version von Kinesis Agent für Windows zu installieren, fügen Sie die`-version`:

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

Ersetzen*version*mit einer gültigen Versionsnummer für Kinesis Agent für Windows. Versionsinformationen finden Sie unterKinesis-Agent-Windows-Repository auf GitHub.

Es gibt zahlreiche Bereitstellungs-Tools, mit denen PowerShell-Skripts remote ausgeführt werden können. Sie können auch zum Automatisieren der Installation von Kinesis Agent für Windows auf Flotten von Servern oder Desktops verwendet werden.

Konfigurieren und Starten von Kinesis Agent für Windows

Nachdem Sie Kinesis Agent für Windows installiert haben, müssen Sie den Agenten konfigurieren und starten. Danach sollte bei der Operation kein weiterer Eingriff erforderlich sein.

So konfigurieren und starten Sie Kinesis Agent für Windows

1. Erstellen Sie eine Kinesis Agent für Windows -Konfigurationsdatei und stellen Sie sie bereit. Diese Datei konfiguriert die Quellen, Senken und Pipes zusammen mit anderen globalen Konfigurationselementen.

   Weitere Informationen zur Kinesis Agent für Windows -Konfiguration finden Sie unterKonfigurieren von Amazon Kinesis Agent für Microsoft Windows.

   Ausführliche Konfigurationsdateibeispiele, die Sie anpassen und installieren können, finden Sie unter Konfigurationsbeispiele für Kinesis Agent für Windows.
2. Öffnen Sie ein PowerShell Befehlszeilenfenster mit erhöhten Rechten und starten Sie Kinesis Agent für Windows mit dem folgenden PowerShell-Befehl:

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

Konfigurieren von Amazon Kinesis Agent für Microsoft Windows

Bevor Sie Amazon Kinesis Agent für Microsoft Windows starten, müssen Sie eine Konfigurationsdatei erstellen und bereitstellen. Die Konfigurationsdatei enthält die erforderlichen Informationen zum Erfassen, Transformieren und Streamen von Daten auf Windows-Server- und -Desktop-Computern für verschiedene AWS-Services. Konfigurationsdateien definieren Gruppen von Quellen, Senken und Pipes, die Quellen mit Senken verbinden, zusammen mit optionalen Transformationen.

Die Kinesis Agent für Windows-Konfigurationsdatei trägt den Name`appsettings.json`. Stellen Sie diese Datei für `%PROGRAMFILES%\Amazon\AWSKinesisTap` bereit.

Themen

Grundlegende Konfigurationsstruktur

Die grundlegende Struktur der Konfigurationsdatei von Amazon Kinesis Agent für Microsoft Windows ist ein JSON-Dokument mit der folgenden Vorlage:

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

- Bei dem Wert von `Sources` handelt es sich um eine oder mehrere Quell-Deklarationen.
- Bei dem Wert von `Sinks` handelt es sich um eine oder mehrere Senken-Deklarationen.

- Bei dem Wert von `Pipes` handelt es sich um eine oder mehrere Pipe-Deklarationen.

Weitere Informationen über die Konzepte von Kinesis Agent für Windows-Quelle, Pipe und Senke in finden Sie unterKonzepte von Amazon Kinesis Agent für Microsoft Windows.

Das folgende Beispiel ist eine vollständige`appsettings.json`Konfigurieren von, mit der Kinesis Agent für Windows zum Streamen von Windows-AnwendungsprotokollKinesis Data Firehose eignissen an konfiguriert wird.

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

Informationen über die einzelnen Deklarationen finden Sie in den folgenden Abschnitten:

- Quell-Deklarationen
- Senken-Deklarationen
- Pipe-Deklarationen

Groß-/Kleinschreibung bei der Konfiguration

Bei Dateien im JSON-Format werden in der Regel Groß- und Kleinschreibung beachtet, und Sie sollten davon ausgehen, dass bei allen Schlüsseln und Werten in -Konfigurationsdateien von Kinesis Agent für Windows ebenfalls zwischen Groß- und Kleinschreibung Bei einigen Schlüsseln und Werten in der Konfigurationsdatei `appsettings.json` wird nicht zwischen Groß- und Kleinschreibung unterschieden. Diese sind z. B.:

- Der Wert des `Format`-Schlüssel-Wert-Paares für Senken. Weitere Informationen finden Sie unter Senken-Deklarationen.
- Der Wert des `SourceType`-Schlüssel-Wert-Paares für Quellen, des `SinkType`-Schlüssel-Wert-Paares für Senken und des `Type`-Schlüssel-Wert-Paares für Pipes und Plug-Ins.
- Der Wert des `RecordParser`-Schlüssel-Wert-Paares für die Quelle `DirectorySource`. Weitere Informationen finden Sie unter DirectorySource-Konfiguration.
- Der Wert des `InitialPosition`-Schlüssel-Wert-Paares für Quellen. Weitere Informationen finden Sie unter Lesezeichen-Konfiguration.
- Präfixe für Variablensubstitutionen. Weitere Informationen finden Sie unter Konfigurieren von Senken-Variablensubstitutionen.

Quell-Deklarationen

In Amazon Kinesis Agent für Microsoft WindowsQuell-DeklarationenGeben Sie an, wo und welche Protokoll-, Ereignis- und Metrikdaten gesammelt werden sollen. Außerdem geben sie optional Informationen für die Analyse dieser Daten an, sodass sie umgewandelt werden können. Die folgenden Abschnitte beschreiben Konfigurationen für die integrierten Quell-Typen, die in Kinesis Agent für Windows verfügbar sind. Da Kinesis Agent für Windows erweiterbar ist, können Sie benutzerdefinierte Quelltypen hinzufügen. Jeder Quelltyp erfordert in der Regel bestimmte Schlüssel-Wert-Paare in den Konfigurationsobjekten, die für diesen Quelltyp relevant sind.

Alle Quell-Deklarationen müssen mindestens die folgenden Schlüssel-Wert-Paare enthalten:

`Id`

Eine eindeutige Zeichenfolge, die ein bestimmtes Quellobjekt innerhalb der Konfigurationsdatei identifiziert.

`SourceType`

Der Name des Quelltyps für dieses Quellobjekt. Der Quelltyp gibt den Ursprung der Protokoll-, Ereignis- oder Metrikdaten an, die von diesem Quellobjekt erfasst werden. Er steuert zudem, welche anderen Aspekte der Quelle deklariert werden können.

Beispiele für vollständige Konfigurationsdateien, die verschiedene Arten von Quell-Deklarationen verwenden, finden Sie unter Streamen aus verschiedenen Quellen an Kinesis Data Streams.

Themen

DirectorySource-Konfiguration

Overview

Der `DirectorySource`-Quelltyp sammelt Protokolle aus Dateien, die im angegebenen Verzeichnis gespeichert sind. Da es Protokolldateien in vielen verschiedenen Formate gibt, können Sie mit der `DirectorySource`-Deklaration das Format der Daten in der Protokolldatei angeben. Anschließend können Sie den Inhalt des Protokolls in ein Standardformat, z. B. JSON oder XML, transformieren, bevor Sie ihn an verschiedene AWS-Services streamen.

Es folgt ein Beispiel für eine `DirectorySource`-Deklaration:

```
{
    "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"
}
```

Alle `DirectorySource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge "`DirectorySource`" sein (erforderlich).

`Directory`

Der Pfad zum Verzeichnis mit den Protokolldateien (erforderlich).

`FileNameFilter`

Schränkt optional den Satz von Dateien in dem Verzeichnis ein, in dem Protokolldaten basierend auf einem Dateibenennungsmuster mit Platzhaltern gesammelt werden. Wenn Sie mehrere Protokolldateinamenmuster haben, können Sie mit dieser Funktion eine einzelne`DirectorySource`Verwenden Sie, wie im folgenden Beispiel gezeigt.

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

Systemadministratoren komprimieren manchmal Protokolldateien, bevor sie archiviert werden. Wenn Sie"`*.*`"in`FileNameFilter`, werden bekannte komprimierte Dateien jetzt ausgeschlossen. Diese Funktion verhindert`.zip`,`.gz`, und`.bz2`Dateien versehentlich gestreamt werden. Wenn dieses Schlüssel-Wert-Paar nicht angegeben wird, werden standardmäßig Daten aus allen Dateien im Verzeichnis gesammelt.

`IncludeSubdirectories`

Gibt an, dass Unterverzeichnisse in beliebiger Tiefe vom Betriebssystem begrenzt überwacht werden sollen. Diese Funktion ist nützlich für die Überwachung von Webservern mit mehreren

Websites. Sie können auch die verwenden:`IncludeDirectoryFilter`-Attribut, um nur bestimmte Unterverzeichnisse zu überwachen, die im Filter angegeben sind.

`RecordParser`

Gibt an, wie der `DirectorySource`-Quelltyp die Protokolldateien analysieren soll, die im angegebenen Verzeichnis gefunden werden. Dieses Schlüssel-Wert-Paar ist erforderlich, und die gültigen Werte sind wie folgt:

- `SingleLine`— Jede Zeile der Protokolldatei ist ein Protokolldatensatz.
- `SingleLineJson`— Jede Zeile der Protokolldatei ist ein JSON-formatierter Protokolldatensatz. Dieser Parser ist nützlich, wenn Sie zusätzliche Schlüssel-Wert-Paare zur JSON-fähigen Objektausstattung hinzufügen möchten. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen. Ein Beispiel für den `SingleLineJson`-Datensatz-Parser finden Sie unter Tutorial: Streamen von JSON-Protokolldateien mit Kinesis Agent für Windows zu Amazon S3.
- `Timestamp`— Eine oder mehrere Zeilen können einen Protokolldatensatz enthalten. Der Protokolldatensatz beginnt mit einem Zeitstempel. Diese Option erfordert die Angabe des `TimestampFormat`-Schlüssel-Wert-Paares.
- `Regex`— Jeder Datensatz beginnt mit Text, der einem bestimmten regulären Ausdruck entspricht. Diese Option erfordert die Angabe des `Pattern`-Schlüssel-Wert-Paares.
- `SysLog`— Gibt an, dass die Protokolldatei in dassyslogStandardformat. Die Analyse der Protokolldatei in Datensätze wird basierend auf dieser Spezifikation durchgeführt.
- `Delimited`— Eine einfachere Version des Regex-Datensatzparsers, bei dem Datenelemente in den Protokolldatensätzen durch ein konsistentes Trennzeichen getrennt werden. Diese Option ist einfacher in der Verwendung und schneller auszuführen als der Regex-Parser und wird bevorzugt, wenn diese Option verfügbar ist. Wenn Sie diese Option verwenden, müssen Sie das `Delimiter`-Schlüssel-Wert-Paar angeben.

`TimestampField`

Gibt an, welches JSON-Feld den Zeitstempel für den Datensatz enthält. Dies wird nur mit dem `SingleLineJson` `RecordParser` verwendet. Dieses Schlüssel-Wert-Paar ist optional. Wenn es nicht angegeben wird, verwendet Kinesis Agent für Windows den Zeitpunkt, als der Datensatz für den Zeitstempel gelesen wurde. Ein Vorteil für die Angabe dieses Schlüssel-Wert-Paares ist, dass von Kinesis Agent für Windows erstellte Latenzstatistiken genauer sind.

`TimestampFormat`

Gibt an, wie Datum und Uhrzeit, die dem Datensatz zugeordnet sind, zu analysieren sind. Der Wert ist entweder die Zeichenfolge `epoch` oder eine .NET-Formatzeichenfolge für Datum und Uhrzeit. Wenn der Wert `epoch` ist, wird die Uhrzeit basierend auf der UNIX-Epoche-Zeit analysiert. Weitere Informationen zur UNIX-Epoche-Zeit finden Sie unter Unix-Zeit. Weitere Informationen zu .NET-Formatzeichenfolgen für Datum/Uhrzeit finden Sie unter Custom Date and Time Format Strings in der Microsoft .NET-Dokumentation. Dieses Schlüssel-Wert-Paar ist nur erforderlich, wenn der `Timestamp`-Datensatz-Parser angegeben wird oder wenn der `SingleLineJson`-Datensatz-Parser zusammen mit dem `TimestampField`-Schlüssel-Wert-Paar angegeben wird.

`Pattern`

Gibt einen regulären Ausdruck an, der mit der ersten Zeile eines potenziell mehrzeiligen Datensatzes übereinstimmen muss. Dieses Schlüssel-Wert-Paar ist nur für den `Regex`-Datensatz-Parser erforderlich.

`ExtractionPattern`

Gibt einen regulären Ausdruck an, der benannte Gruppen verwenden soll. Der Datensatz wird unter Verwendung dieses regulären Ausdrucks analysiert, und die benannten Gruppen bilden die Felder des analysierten Datensatzes. Diese Felder werden dann als Basis für die Erstellung von JSON- oder XML-Objekten oder -Dokumenten verwendet, die durch Senken an verschiedene AWS-Services gestreamt werden. Dieses Schlüssel-Wert-Paar ist optional und ist mit der Option`Regex`Datensatzparser und der Timestamp-Parser.

Der `Timestamp`-Gruppenname wird speziell verarbeitet, da er dem `Regex`-Parser anzeigt, welches Feld das Datum und die Uhrzeit für jeden Datensatz in jeder Protokolldatei enthält.

`Delimiter`

Gibt das Zeichen und die Zeichenfolge an, durch die jedes Element in jedem Protokolldatensatz getrennt wird. Dieses Schlüssel-Wert-Paar darf (und kann) nur mit dem `Delimited`-Datensatz-Parser verwendet werden. Verwenden Sie zur Darstellung des Tabulatorzeichens die aus zwei Buchstaben bestehende Sequenz `\t`.

`HeaderPattern`

Gibt einen regulären Ausdruck zur Anpassung der Zeile in der Protokolldatei an, die die Gruppe von Headern für den Datensatz enthält. Wenn die Protokolldatei keine Header-Informationen

enthält, verwenden Sie das `Headers`-Schlüssel-Wert-Paar zur Angabe der impliziten Header. Dieses `HeaderPattern`-Schlüssel-Wert-Paar ist optional und ist nur für den `Delimited`-Datensatz-Parser gültig.

> **Note**
>
> Ein leerer (0 Länge) Header-Eintrag für eine Spalte bewirkt, dass die Daten für diese Spalte aus der endgültigen Ausgabe der analysierten `DirectorySource`-Ausgabe gefiltert werden.

`Headers`

Gibt die Namen für die analysierten Datenspalten unter Verwendung des angegebenen Trennzeichens an. Dieses Schlüssel-Wert-Paar ist optional und ist nur für den `Delimited`-Datensatz-Parser gültig.

> **Note**
>
> Ein leerer (0 Länge) Header-Eintrag für eine Spalte bewirkt, dass die Daten für diese Spalte aus der endgültigen Ausgabe der analysierten `DirectorySource`-Ausgabe gefiltert werden.

`RecordPattern`

Gibt einen regulären Ausdruck an, der Zeilen in der Protokolldatei mit Datensatzdaten identifiziert. Außer der von `HeaderPattern` identifizierten Header-Zeile werden Zeilen, die nicht mit dem angegebenen `RecordPattern` übereinstimmen, während der Datensatzverarbeitung ignoriert. Dieses Schlüssel-Wert-Paar ist optional und ist nur für den `Delimited`-Datensatz-Parser gültig. Wenn es nicht angegeben wird, wird standardmäßig eine beliebige Zeile, die nicht mit dem optionalen `HeaderPattern` oder dem optionalen `CommentPattern` übereinstimmt, als eine Zeile mit analysierbaren Datensatzdaten angesehen.

`CommentPattern`

Gibt einen regulären Ausdruck an, der Zeilen in der Protokolldatei identifiziert, die ausgeschlossen werden sollen, bevor die Daten in der Protokolldatei analysiert werden. Dieses Schlüssel-Wert-Paar ist optional und ist nur für den `Delimited`-Datensatz-Parser gültig. Wenn es

nicht angegeben wird, wird standardmäßig eine beliebige Zeile, die nicht mit dem optionalen `HeaderPattern` übereinstimmt, als eine Zeile mit analysierbaren Datensatzdaten angesehen, sofern nicht `RecordPattern` angegeben wird.

`TimeZoneKind`

Gibt an, ob sich der Zeitstempel in der Protokolldatei auf die lokale Zeitzone oder die UTC-Zeitzone bezieht. Dies ist optional und standardmäßig auf UTC eingestellt. Die einzigen gültigen Werte für dieses Schlüssel-Wert-Paar sind `Local` oder `UTC`. Der Zeitstempel wird niemals geändert, wenn `TimeZoneKind` nicht angegeben wird oder wenn der Wert UTC ist. Der Zeitstempel wird in UTC konvertiert, wenn der`TimeZoneKind`Wert ist`Local`Die Senke, die den Zeitstempel empfängt, ist CloudWatch Logs, oder wenn der analysierte Datensatz zu anderen Senken gesendet wird. Datums- und Zeitangaben, die in Meldungen eingebettet sind, werden nicht konvertiert.

`SkipLines`

Steuert, sofern angegeben, die Anzahl von Zeilen, die am Anfang jeder Protokolldatei ignoriert werden, bevor der Datensatz analysiert wird. Dies ist optional und der Standardwert ist 0.

Codierung

Standardmäßig kann Kinesis Agent für Windows die Codierung automatisch aus Bytemark erkennen. Bei einigen älteren Unicode-Formaten funktioniert die automatische Codierung jedoch möglicherweise nicht korrekt. Im folgenden Beispiel wird die Codierung angegeben, die zum Streamen eines Microsoft SQL Server-Protokolls erforderlich ist.

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

Eine Liste der Codierungsnamen finden Sie unterListe der Kodierungenin der Microsoft .NET-Dokumentation.

ExtractionRegexOptions

Sie können die Verwendung von`ExtractionRegexOptions`Reguläre Ausdrücke zu vereinfachen. Dieses Schlüssel-Wert-Paar ist optional. Der Standardwert ist `"None"`.

Im folgenden Beispiel wird angegeben, dass die Eigenschaft`"."`Ausdruck entspricht einem beliebigen Zeichen, einschließlich`\r\n`.

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

Eine Liste möglicher Felder für ExtractionRegexOptions finden Sie unter der DokumentationRegExOptions-Aufzählungsumin der Microsoft .NET-Dokumentation.

Regex-Datensatz-Parser

Sie können unstrukturierte Textprotokolle mit dem `Regex`-Datensatz-Parser zusammen mit den `TimestampFormat`–, `Pattern`- und `ExtractionPattern`-Schlüssel-Wert-Paaren analysieren. Angenommen, Ihre Protokolldatei sieht folgendermaßen aus:

```
[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'
```

Sie können in diesem Fall den folgenden regulären Ausdruck für das `Pattern`-Schlüssel-Wert-Paar angeben, um die Protokolldatei leichter in einzelne Protokolldatensätze aufzuteilen:

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

Dieser reguläre Ausdruck weist folgende Sequenz auf:

1. Die Startposition der auszuwertenden Zeichenfolge.
2. Ein oder mehrere Wortzeichen in eckigen Klammern.
3. Ein Zeitstempel in eckigen Klammern. Der Zeitstempel weist folgende Sequenz auf:
   a. Das vierstellige Jahr
   b. Ein Schrägstrich
   c. Der zweistellige Monat
   d. Ein Schrägstrich
   e. Der zweistellige Tag
   f. Ein Leerzeichen
   g. Die zweistellige Stunde
   h. Ein Doppelpunkt

i. Die zweistellige Minute
j. Ein Doppelpunkt
k. Die zweistellige Sekunde
l. Ein Punkt
m. Die dreistellige Millisekunde

Sie können das folgende Format für das `TimestampFormat`-Schlüssel-Wert-Paar angeben, um den Zeitstempel in Textform in ein Datum und eine Uhrzeit umzuwandeln:

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

Sie können den folgenden regulären Ausdruck zum Extrahieren der Felder des Protokolldatensatzes über das `ExtractionPattern`-Schlüssel-Wert-Paar extrahieren.

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

Dieser reguläre Ausdruck weist die folgenden Gruppen in der Sequenz auf:

1. `Severity`— Ein oder mehrere Wortzeichen in eckigen Klammern.
2. `TimeStamp`— Siehe die vorherige Beschreibung des Zeitstempels.
3. Drei unbenannte Sequenzen mit null oder mehr Zeichen in eckigen Klammern werden übersprungen.
4. `SubSystem`— Ein oder mehrere Wortzeichen in eckigen Klammern.
5. `Module`— Ein oder mehrere Wortzeichen in eckigen Klammern.
6. Eine unbenannte Sequenz mit null oder mehr Zeichen in eckigen Klammern wird übersprungen.
7. Eine unbenannte Leerstelle wird übersprungen.
8. `Message`— Null oder mehr Zeichen, die von einfachen Anführungszeichen umgeben sind.

Die folgende Quell-Deklaration kombiniert diese regulären Ausdrücke und das Format für Datum und Uhrzeit in den vollständigen Anleitungen für Kinesis Agent für Windows zum Analysieren dieser Art von Protokolldatei.

```
{
```

```
    "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**
>
> Umgekehrte Schrägstriche in JSON-formatierten Dateien müssen mit einem zusätzlichen umgekehrten Schrägstrich als Escapezeichen geschützt werden.

Weitere Informationen zu regulären Ausdrücken finden Sie unter Regular Expression Language - Quick Reference in der Microsoft .NET-Dokumentation.

`Delimited`-Datensatz-Parser

Sie können mit dem `Delimited`-Datensatz-Parser halbstrukturierte Protokoll- und Datendateien analysieren, bei denen die einzelnen Datenspalten in den einzelnen Datenzeile durch eine konsistente Zeichenfolge voneinander getrennt sind. Beispiel: In CSV-Dateien sind die einzelnen Spaltendaten durch ein Semikolon und bei TSV-Dateien durch ein Tabulatorzeichen voneinander getrennt.

Angenommen, Sie möchten eine Microsoft Protokolldatei im NPS-Datenbankformat analysieren, die von einem Netzwerkrichtlinienserver erstellt wurde. Eine solche Datei könnte in diesem Fall folgendermaßen aussehen:

```
"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,,,,
```

Das folgende Beispiel einer `appsettings.json`-Konfigurationsdatei enthält eine `DirectorySource`-Deklaration, die diesen Text mittels eines `Delimited`-Datensatz-Parsers als Objektdarstellung analysiert. Anschließend streamt sie die Daten im JSON-Format an Kinesis Data Firehose:

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

An Kinesis Data Firehose gestreamt Daten im JSON-Format sehen wie folgt aus:

```
{
    "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": ""
}
```

SysLog-Datensatz-Parser

Für den SysLog-Datensatz-Parser enthält die analysierte Ausgabe aus der Quelle die folgenden Informationen:

Attribut	Typ	Beschreibung
`SysLogTimeStamp`	Zeichenlge	Das ursprüngliche Datum und die ursprüngliche Uhrzeit aus der Syslog-formatierten Protokolldatei.
`Hostname`	Zeichenlge	Der Name des Computers, auf dem die Syslog-formatierte Protokolldatei gespeichert ist.
`Program`	Zeichenlge	Der Name der Anwendung oder des Service, mit der bzw. dem die Protokolldatei erstellt wurde.
`Message`	Zeichenlge	Der von der Anwendung oder dem Service erstellte Protokolleintrag.
`TimeStamp`	Zeichenlge	Das analysierte Datum und die analysierte Uhrzeit im ISO 8601-Format.

Im Folgenden finden Sie ein Beispiel von in das JSON-Format umgewandelten SysLog-Daten:

```
{
    "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

Im Folgenden finden Sie eine Zusammenfassung der verfügbaren Schlüssel-Wert-Paare für die `DirectorySource`-Quelle und die `RecordParser`, die sich auf diese Schlüssel-Wert-Paare beziehen.

Schlüsselname	RecordParser	Hinweise
`SourceType`	Erforderlich für alle	Der Wert muss `DirectorySource` lauten
`Directory`	Erforderlich für alle	
`FileNameFilter`	Optional für alle	
`RecordParser`	Erforderlich für alle	
`TimestampField`	Optional für `SingleLineJson`	
`TimestampFormat`	Erforderlich für `Timestamp` und für `SingleLineJson`, wenn `TimestampField` angegeben wird	
`Pattern`	Erforderlich für `Regex`	
`ExtractionPattern`	Optional für `Regex`	Erforderlich für `Regex`, wenn die Senke als Format `json` oder `xml` angibt
`Delimiter`	Erforderlich für `Delimited`	
`HeaderPattern`	Optional für `Delimited`	
`Headers`	Optional für `Delimited`	
`RecordPattern`	Optional für `Delimited`	
`CommentPattern`	Optional für `Delimited`	

Schlüsselname	RecordParser	Hinweise
`TimeZoneKind`	Optional für `Regex`, `Timestamp` , `SysLog` und `SingleLineJson` , wenn ein Zeitstempel-Feld identifiziert wird	
`SkipLines`	Optional für alle	

ExchangeLogSource-Konfiguration

Der `ExchangeLogSource`-Typ wird verwendet, um Protokolle von Microsoft Exchange zu sammeln. Exchange erstellt Protokolle in mehrere verschiedene Protokollformaten. Dieser Quelltyp analysiert alle von ihnen. Es ist zwar auch möglich, sie unter Verwendung des Typs `DirectorySource` mit dem Datensatz-Parser `Regex` zu analysieren, aber wesentlich einfacher, `ExchangeLogSource` zu verwenden. Dies liegt daran, dass Sie für die Protokolldateiformate keine regulären Ausdrücke entwerfen und bereitstellen müssen. Es folgt ein Beispiel für eine `ExchangeLogSource`-Deklaration:

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

Alle Exchange-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge `"ExchangeLogSource"` sein (erforderlich).

`Directory`

Der Pfad zum Verzeichnis mit den Protokolldateien (erforderlich).

`FileNameFilter`

Schränkt optional den Satz von Dateien in dem Verzeichnis ein, in dem Protokolldaten basierend auf einem Dateibenennungsmuster mit Platzhaltern gesammelt werden. Wenn dieses Schlüssel-

Wert-Paar nicht angegeben wird, werden standardmäßig Protokolldaten aus allen Dateien im Verzeichnis gesammelt.

`TimestampField`

Der Name der Spalte enthält das Datum und die Uhrzeit für den Datensatz. Dieses Schlüssel-Wert-Paar ist optional und muss nicht angegeben werden, wenn der Feldname `date-time` oder `DateTime` lautet. Andernfalls ist er erforderlich.

W3SVCLogSource-Konfiguration

Der Typ `W3SVCLogSource` wird verwendet, um Protokolle von Internet Information Services (IIS) für Windows zu sammeln.

Es folgt ein Beispiel für eine `W3SVCLogSource`-Deklaration:

```
{
   "Id": "MyW3SVCLog",
   "SourceType": "W3SVCLogSource",
   "Directory": "C:\\inetpub\\logs\\LogFiles\\W3SVC1",
   "FileNameFilter": "*.log"
}
```

Alle `W3SVCLogSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge `"W3SVCLogSource"` sein (erforderlich).

`Directory`

Der Pfad zum Verzeichnis mit den Protokolldateien (erforderlich).

`FileNameFilter`

Schränkt optional den Satz von Dateien in dem Verzeichnis ein, in dem Protokolldaten basierend auf einem Dateibenennungsmuster mit Platzhaltern gesammelt werden. Wenn dieses Schlüssel-Wert-Paar nicht angegeben wird, werden standardmäßig Protokolldaten aus allen Dateien im Verzeichnis gesammelt.

UlsSource-Configuration

Der `UlsSource`-Typ wird verwendet, um Protokolle von Microsoft SharePoint zu sammeln. Es folgt ein Beispiel für eine `UlsSource`-Deklaration:

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

Alle `UlsSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge `"UlsSource"` sein (erforderlich).

`Directory`

Der Pfad zum Verzeichnis mit den Protokolldateien (erforderlich).

`FileNameFilter`

Schränkt optional den Satz von Dateien in dem Verzeichnis ein, in dem Protokolldaten basierend auf einem Dateibenennungsmuster mit Platzhaltern gesammelt werden. Wenn dieses Schlüssel-Wert-Paar nicht angegeben wird, werden standardmäßig Protokolldaten aus allen Dateien im Verzeichnis gesammelt.

WindowsEventLogSource-Konfiguration

Der Typ `WindowsEventLogSource` wird verwendet, um Ereignisse vom Windows-Ereignisprotokoll-Service zu sammeln. Es folgt ein Beispiel für eine `WindowsEventLogSource`-Deklaration:

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

Alle `WindowsEventLogSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge `"WindowsEventLogSource"` sein (erforderlich).

`LogName`

Ereignisse werden aus dem angegebenen Protokoll gesammelt. Geläufige Werte sind `Application`, `Security` und `System`, Sie können aber einen beliebigen gültigen Windows-Ereignisprotokollnamen angeben. Dieser Schlüssel-Wert-Paar ist erforderlich.

`Query`

Schränkt optional ein, welche Ereignisse von `WindowsEventLogSource` ausgegeben werden. Wenn dieses Schlüssel-Wert-Paar nicht angegeben wird, werden standardmäßig alle Ereignisse ausgegeben. Weitere Informationen zur Syntax dieses Wertes finden Sie unter Event Queries and Event XML in der Windows-Dokumentation. Weitere Informationen zu Definitionen auf Protokollebene finden Sie unter Event Types in der Windows-Dokumentation.

`IncludeEventData`

Ermöglicht optional das Sammeln und Streamen von anbieterspezifischen Ereignisdaten im Zusammenhang mit Ereignissen aus dem angegebenen Windows-Ereignisprotokoll, wenn der Wert des Schlüssel-Wert-Paares `"true"` ist. Eingeschlossen werden nur Ereignisdaten, die erfolgreich serialisiert werden können. Dieses Schlüssel-Wert-Paar ist optional. Wenn es nicht angegeben wird, werden keine anbieterspezifische Ereignisdaten gesammelt.

> **Note**
>
> Wenn Ereignisdaten eingeschlossen werden, könnte sich die von dieser Quelle gestreamte Datenmenge signifikant erhöhen. Die maximale Größe eines Ereignisses mit eingeschlossenen Ereignisdaten kann 262.143 Bytes betragen.

Die analysierte Ausgabe von `WindowsEventLogSource` enthält die folgenden Informationen:

Attribut	Typ	Beschreibung
`EventId`	Int	Die ID der Art von Ereignisses.
`Description`	Zeichenfolge	Text zur Beschreibung der Details des Ereignisses.

Attribut	Typ	Beschreibung
`LevelDisplayName`	Zeichenfolge	Die Kategorie des Ereignisses (Fehler, Warnung, Informationen, Erfolgsüberwachung, Fehlerüberwachung).
`LogName`	Zeichenfolge	Wo das Ereignis aufgezeichnet wurde (typische Werte sind `Application`, `Security` und `System`, aber es gibt zahlreiche Möglichkeiten).
`MachineName`	Zeichenfolge	Von welchem Computer das Ereignis aufgezeichnet wurde.
`ProviderName`	Zeichenfolge	Von welcher Anwendung oder welchem Service das Ereignis aufgezeichnet wurde.
`TimeCreated`	Zeichenfolge	Wann das Ereignis aufgetreten ist, im ISO 8601-Format.
`Index`	Int	An welcher Stelle im Protokoll sich das Ereignis befindet.
`UserName`	Zeichenfolge	Wer den Eintrag verfasst hat, sofern bekannt.

Attribut	Typ	Beschreibung
`Keywords`	Zeichenfolge	Der Ereignistyp. Standard-Werte sind `AuditFailure` (fehlgeschlagene Sicherheitsüberwachungsereignisse), `AuditSuccess` (erfolgreiche Sicherheitsüberwachungsereignisse), `Classic` (mit der Funktion `RaiseEvent` gemeldete Ereignisse), `Correlation Hint` (Übertragungsereignisse), `SQM` (Ereignisse des Servicequalitätsmechanismus), `WDI Context` (Kontextereignisse der Windows Diagnostic Infrastructure) und `WDI Diag` (Diagnoseereignisse der Windows Diagnostic Infrastructure).
`EventData`	Liste von Objekten	Optionale anbieterspezifische zusätzliche Daten über das Protokollereignis. Diese werden nur eingeschlossen, wenn der Wert für das `IncludeEventData` -Schlüssel-Wert-Paar `"true"` lautet.

Es folgt ein Beispiel eines in das JSON-Format umgewandelten Ereignisses:

```
{[
    "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=="
]}
```

WindowSeventLogPollingSource-Konfiguration

`WindowsEventLogPollingSource`verwendet einen abfragebasierten Mechanismus, um alle neuen Ereignisse aus dem Ereignisprotokoll zu sammeln, die den konfigurierten Parametern entsprechen. Das Abrufintervall wird dynamisch zwischen 100 ms und 5000 ms aktualisiert, je nachdem, wie viele Ereignisse während der letzten Umfrage gesammelt wurden. Es folgt ein Beispiel für eine `WindowsEventLogPollingSource`-Deklaration:

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

Alle `WindowsEventLogPollingSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge "`WindowsEventLogPollingSource`" sein (erforderlich).

`LogName`

Gibt das Protokoll an. Gültige Optionen sind:`Application`,`Security`,`Systemoder` andere gültige Protokolle.

`IncludeEventData`

Optional. Wann`true`, gibt an, dass zusätzliche EventData beim Streamen als JSON und XML enthalten ist. Der Standardwert ist `false`.

`Query`

Optional. Windows-Ereignisprotokolle unterstützen das Abfragen von Ereignissen mithilfe von XPath-Ausdrücken, die Sie mit`Query`. Weitere Informationen finden Sie unterEreignisabfragen und Ereignis-XMLin der Microsoft-Dokumentation.

`CustomFilters`

Optional. Eine Liste von Filtern, die durch ein Semikolon getrennt sind (`;`) enthalten. Die folgenden Filter können angegeben werden.

`ExcludeOwnSecurityEvents`

Schließt Sicherheitsereignisse aus, die von Kinesis Agent für Windows selbst generiert wurden.

WindowsETWEventSource-Konfiguration

Der `WindowsETWEventSource`-Typ wird verwendet, um Ereignisablaufverfolgungen für Anwendungen und Services mithilfe der Funktion Event Tracing for Windows (ETW) zu sammeln. Weitere Informationen finden Sie unter Event Tracing in der Windows-Dokumentation.

Es folgt ein Beispiel für eine `WindowsETWEventSource`-Deklaration:

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

Alle `WindowsETWEventSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge "`WindowsETWEventSource`" sein (erforderlich).

`ProviderName`

Gibt an, welcher Ereignisanbieter zum Sammeln von Ablaufverfolgungsereignissen verwendet werden soll. Dies muss ein gültiger ETW-Anbietername für einen installierten Anbieter sein. Um

zu bestimmen, welche Anbieter installiert sind, führen Sie in einem Windows-Befehlszeilenfenster Folgendes aus:

```
logman query providers
```

`TraceLevel`

Gibt an, welche Kategorien von Ablaufverfolgungsereignissen erfasst werden sollen. Zulässige Werte sind `Critical`, `Error`, `Warning`, `Informational` und `Verbose`. Die genaue Bedeutung ist von dem jeweils ausgewählten ETW-Anbieter abhängig.

`MatchAnyKeyword`

Dieser Wert ist eine 64-Bit-Zahl, bei der jedes Bit ein einzelnes Schlüsselwort darstellt. Jedes Schlüsselwort beschreibt eine Kategorie von Ereignissen, die erfasst werden sollen. Informationen zu den unterstützten Schlüsselwörtern und deren Werten und Beziehung zu `TraceLevel` finden Sie in der Dokumentation des Anbieters. Beispiel: Informationen über den CLR-ETW-Anbieter finden Sie unter CLR ETW Keywords and Levels in der Microsoft .NET Framework-Dokumentation.

Im vorherigen Beispiel stellt 32768 (0x00008000) das `ExceptionKeyword` für den CLR-ETW-Anbieter dar, durch das der Anbieter angewiesen wird, Informationen zu den ausgelösten Ausnahmen zu erfassen. Obwohl JSON keine Hex-Konstanten unterstützt, können Sie sie für `MatchAnyKeyword` angeben, indem Sie sie in einer Zeichenfolge platzieren. Sie können auch mehrere, durch Kommata getrennte Konstanten eingeben. Führen Sie z. B. die folgenden Schritt durch, um sowohl `ExceptionKeyword` als auch `SecurityKeyword` (0x00000400) anzugeben:

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

Um sicherzustellen, dass alle angegebenen Schlüsselwörter für einen Anbieter aktiviert sind, werden mehrere Schlüsselwortwerte mit OR miteinander kombiniert und an diesen Anbieter übergeben.

Die Ausgabe von `WindowsETWEventSource` enthält die folgenden Informationen für jedes Ereignis:

Attribut	Typ	Beschreibung
`EventName`	Zeichenfolge	Welche Art von Ereignis aufgetreten ist.
`ProviderName`	Zeichenfolge	Welcher Anbieter das Ereignis erkannt hat.
`FormattedMessage`	Zeichenfolge	Ein Text mit einer Zusammenfassung des Ereignisses.
`ProcessID`	Int	Welcher Prozess das Ereignis gemeldet hat.
`ExecutingThreadID`	Int	Welcher Thread innerhalb der Prozesses das Ereignis gemeldet hat.
`MachineName`	Zeichenfolge	Der Name des Desktops oder Servers, von dem das Ereignis gemeldet wird.
`Payload`	Hashtabelle	Eine Tabelle mit einem Zeichenfolgenschlüssel und jeder Art von Objekt als Wert. Der Schlüssel ist der Name des Nutzlastelements und der Wert ist der Wert des Nutzlastelements. Die Nutzlast ist anbieterabhängig.

Es folgt ein Beispiel eines in das JSON-Format umgewandelten Ereignisses:

```
{
     "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
      }
}
```

WindowsPerformanceCounterSource-Konfiguration

Der `WindowsPerformanceCounterSource`-Typ erfasst die Leistungsindikator-Metriken von Windows. Es folgt ein Beispiel für eine `WindowsPerformanceCounterSource`-Deklaration:

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

Alle `WindowsPerformanceCounterSource`-Deklarationen können die folgenden Schlüssel-Wert-Paare angeben:

`SourceType`

Muss die Literalzeichenfolge `"WindowsPerformanceCounterSource"` sein (erforderlich).

`Categories`

Gibt einen Satz von Leistungsindikator-Metrikgruppen an, die von Windows erfasst werden sollen. Jede Metrikgruppe enthält die folgenden Schlüssel-Wert-Paare:

`Category`

Gibt den Leistungsindikatorsatz von Metriken an, die erfasst werden sollen (erforderlich).

`Instances`

Gibt den Satz relevanter Objekte an, wenn ein eindeutiger Satz von Leistungsindikatoren pro Objekt vorhanden ist. Wenn die Kategorie beispielsweise `LogicalDisk` lautet, ist ein Satz von Leistungsindikatoren pro Festplattenlaufwerk vorhanden. Dieses Schlüssel-Wert-Paar ist optional. Sie könne die Platzhalter `*` und `?` verwenden, um mehrere Instances abzugleichen. Um Werte für alle Instances zu aggregieren, geben Sie `_Total` an.

Sie können auch`InstanceRegex`, die reguläre Ausdrücke akzeptiert, die die`*`Platzhalterzeichen als Teil des Instanznamens.

`Counters`

Gibt an, welche Metriken für die angegebene Kategorie zu sammeln sind. Dieser Schlüssel-Wert-Paar ist erforderlich. Sie könne die Platzhalter `*` und `?` verwenden, um mehrere Leistungsindikatoren abzugleichen. Sie können `Counters` nur nach Namen oder nach Namen und Einheit angeben. Wenn keine Leistungsindikatoreinheiten angegeben werden, versucht Kinesis Agent für Windows, die Einheiten vom Namen abzuleiten. Wenn diese Inferenzen falsch sind, kann die Einheit explizit angegeben werden. Sie können die `Counter`-Namen auf

Wunsch ändern. Die komplexere Darstellung eines Leisungsindikators ist ein Objekt mit den folgenden Schlüssel-Wert-Paaren:

`Counter`

Der Name des Leistungsindikators. Dieser Schlüssel-Wert-Paar ist erforderlich.

`Rename`

Der der Senke anzuzeigende Name des Leistungsindikators. Dieses Schlüssel-Wert-Paar ist optional.

`Unit`

Die Bedeutung des Wertes, der mit dem Leistungsindikator verknüpft ist. Eine vollständige Liste gültiger Namen für Einheiten finden Sie in der Dokumentation zu Einheiten unterMetricDatum, mit derimAmazon CloudWatch ch-API-Referenz.

Es folgt ein Beispiel für eine komplexe Leistungsindikator-Spezifikation:

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

`WindowsPerformanceCounterSource`Kann nur mit einer Pipe verwendet werden, die eine Amazon CloudWatch ch-Senke angibt. Verwenden Sie eine separate Senke, wenn integrierte -Metriken von Kinesis Agent für Windows auch zu CloudWatch gestreamt werden. Überprüfen Sie nach dem Start des -Service das Kinesis Agent für Windows-Protokoll, welche Einheiten für Leistungsindikatoren abgeleitet wurden, wenn keine Einheiten in den`WindowsPerformanceCounterSource`-Deklarationen. Bestimmen Sie mithilfe von PowerShell die gültigen Namen für Kategorien, Instances und Leistungsindikatoren.

Um Informationen zu allen Kategorien, einschließlich mit Leistungsindikatorsätzen verknüpften Leistungsindikatoren, anzuzeigen, führen Sie diesen Befehl in einem PowerShell-Fenster aus:

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

Um zu bestimmen, welche Instances für jeden der Leistungsindikatoren im Leistungsindikatorsatz verfügbar sind, führen Sie in einem PowerShell-Fenster einen Befehl wie den folgenden aus:

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

Der Wert des `Counter`-Parameters sollte einer der Pfade aus einem untergeordneten `PathsWithInstances`-Elements sein, das durch den vorherigen `Get-Counter -ListSet`-Befehlsaufruf aufgeführt wurde.

Integrierte Quelle von Kinesis Agent für Windows -Metriken: Quelle

Zusätzlich zu gewöhnlichen Metrikquellen wie dem`WindowsPerformanceCounterSource`Typ (siehe[WindowsPerformanceCounterSource-Konfiguration](#)), kann der -Senkentyp von CloudWatch Metriken von einer besonderen Quelle erhalten, die Metriken zu Kinesis Agent für Windows selbst erfasst. Kinesis Agent für Windows-Metriken sind auch im Bereich`KinesisTap`-Kategorie von Windows-Leistungsindikatoren.

Die`MetricsFilter`Das -Schlüssel-Wert-Paar für die CloudWatch enken-Deklarationen gibt an, welche Metriken von der integrierten -Kinesis Agent für Windows-Metrikenquelle an CloudWatch gestreamt werden. Der Wert ist eine Zeichenfolge mit einem oder mehreren Filterausdrücken, die durch Semikolons getrennt sind, z. B.:

`"MetricsFilter": "FilterExpression1;FilterExpression2"`

Eine Metrik, die mit einem oder mehreren Filterausdrücken übereinstimmt, wird zu CloudWatch gestreamt.

Einzel-Instance-Metriken sind global und nicht an eine bestimmte Quelle oder Senke gebunden. Mehrfach-Instance-Metriken sind dimensional basierend auf der `Id` der Quell- oder Senken-Deklaration. Jeder Quell- oder Senkentyp kann über einen anderen Satz von Metriken verfügen.

Eine Liste integrierter -Metriknamen von Kinesis Agent für Windows finden Sie unter[Liste der Kinesis Agent für Windows-Metriken](#).

Bei Einzel-Instance-Metriken ist der Filterausdruck der Name der Metrik, z. B.:

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

Bei Mehrfach-Instance-Metriken ist der Filterausdruck der Name der Metrik, ein Punkt (`.`) und dann die `Id` der Quell- oder Senken-Deklaration, von der die Metrik generiert wurde. Angenommen, die `Id` einer Senken-Deklaration lautet `MyFirehose`:

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

Sie können in diesem Fall spezielle Platzhaltermuster verwenden, die zwischen Einzel-Instance- und Mehrfach-Instance-Metriken unterscheiden.

- Das Sternchen (`*`) entspricht null oder mehreren Zeichen außer einem Punkt (`.`).
- Das Fragezeichen (`?`) entspricht jedem beliebigen Zeichen außer einem Punkt.
- Alle anderen Zeichen entsprechen nur ihnen selbst.
- `_Total` ist ein spezielles Token, das die Aggregation aller übereinstimmenden Werte mehrerer Instances für die Dimension bewirkt.

Das folgende Beispiel entspricht allen Einzel-Instance-Metriken:

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

Da ein Sternchen keinem Punkt entspricht, werden nur Einzel-Instance-Metriken eingeschlossen.

Das folgende Beispiel entspricht allen Mehrfach-Instance-Metriken:

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

Das folgende Beispiel entspricht allen Metriken (Einzel- und Mehrfach-Instance):

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

Das folgende Beispiel fasst alle Mehrfach-Instance-Metriken für alle Quellen und Senken zusammen:

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

Im folgenden Beispiel werden alle Kinesis Data Firehose -Metriken für alle Kinesis Data Firehose Senken aggregiert:

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

Das folgende Beispiel entspricht allen Einzel- und Mehrfach-Instance-Fehlermetriken:

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

Das folgende Beispiel entspricht allen nicht wiederherstellbaren Fehlermetriken, die für alle Quellen und Senken aggregiert werden:

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

Informationen dazu, wie eine Pipe angegeben wird, die die integrierte -Metrikquelle von Kinesis Agent für Windows verwendet, finden Sie unterKonfigurieren von Kinesis Agent für Windows-Metrik-Pipes.

Liste der Kinesis Agent für Windows-Metriken

Im Folgenden finden Sie eine Liste der Einzel-Instance- und Mehrfach-Instance-Metriken, die für Kinesis Agent für Windows verfügbar sind.

Einzel-Instance-Metriken

Die folgenden Einzel-Instance-Metriken sind verfügbar:

`KinesisTapBuildNumber`

Die Versionsnummer von Kinesis Agent für Windows.

`PipesConnected`

Anzahl von Pipes, die ihre Quelle erfolgreich mit ihrer Senke verbunden haben.

`PipesFailedToConnect`

Anzahl von Pipes, die ihre Quelle nicht erfolgreich mit ihrer Senke verbunden haben.

`SinkFactoriesFailedToLoad`

Anzahl von Senkungsarten, die nicht erfolgreich in Kinesis Agent für Windows geladen wurden.

`SinkFactoriesLoaded`

Anzahl von Senkungsarten, die erfolgreich in Kinesis Agent für Windows geladen wurden.

`SinksFailedToStart`

Anzahl von Senken, die gewöhnlich aufgrund inkorrekter Senken-Deklarationen nicht erfolgreich gestartet wurden.

`SinksStarted`

Anzahl von Senken, die erfolgreich gestartet wurden.

`SourcesFailedToStart`

Anzahl von Quellen, die gewöhnlich aufgrund inkorrekter Quell-Deklarationen nicht erfolgreich gestartet wurden.

`SourcesStarted`

Anzahl von Quellen, die erfolgreich gestartet wurden.

`SourceFactoriesFailedToLoad`

Anzahl von Quelltypen, die nicht erfolgreich in Kinesis Agent für Windows geladen wurden.

`SourceFactoriesLoaded`

Anzahl von Quelltypen, die erfolgreich in Kinesis Agent für Windows geladen wurden.

Mehrfach-Instance-Metriken

Die folgenden Mehrfach-Instance-Metriken sind verfügbar:

DirectorySource-Metriken

`DirectorySourceBytesRead`

Anzahl von Bytes, die während des Intervalls für diese `DirectorySource` gelesen wurden.

`DirectorySourceBytesToRead`

Anzahl der bekannten zum Lesen verfügbare Bytes, die von Kinesis Agent für Windows noch nicht gelesen wurden.

`DirectorySourceFilesToProcess`

Anzahl von bekannten zu untersuchenden bekannten Dateien, die von Kinesis Agent für Windows noch nicht untersucht wurden.

`DirectorySourceRecordsRead`

Anzahl von Datensätzen, die während des Intervalls für diese `DirectorySource` gelesen wurden.

WindowsEventLogSource-Metriken

`EventLogSourceEventsError`

Anzahl von Windows-Ereignisprotokoll-Ereignissen, die nicht erfolgreich gelesen wurden.

`EventLogSourceEventsRead`

Anzahl von Windows-Ereignisprotokoll-Ereignissen, die erfolgreich gelesen wurden.

KinesisFirehose-Senken-Metriken

`KinesisFirehoseBytesAccepted`

Anzahl von Bytes, die während des Intervalls akzeptiert wurden.

`KinesisFirehoseClientLatency`

Zeitspanne, die zwischen dem Erstellen und dem Streamen des Datensatzes an den -Service von Kinesis Data Firehose verstrichen ist.

`KinesisFirehoseLatency`

Zeitspanne, die zwischen dem Beginn und dem Ende des Datensatz-Streamings für den Kinesis Data Firehose -Service verstrichen ist.

`KinesisFirehoseNonrecoverableServiceErrors`

Häufigkeit, mit der Datensätze trotz Neuversuche nicht fehlerfrei an den Kinesis Data Firehose -Service gesendet werden konnten.

`KinesisFirehoseRecordsAttempted`

Anzahl von Datensätzen, bei denen das Streamen an den Kinesis Data Firehose -Service versucht wurde.

`KinesisFirehoseRecordsFailedNonrecoverable`

Anzahl von Datensätzen, die trotz Wiederholungsversuche nicht erfolgreich an den Kinesis Data Firehose -Service gestreamt wurden.

`KinesisFirehoseRecordsFailedRecoverable`

Anzahl von Datensätzen, die erfolgreich an den Kinesis Data Firehose -Service gestreamt wurden, aber nur mit Wiederholungsversuchen.

`KinesisFirehoseRecordsSuccess`

Anzahl von Datensätzen, die erfolgreich ohne Wiederholungsversuche an den Kinesis Data Firehose -Service gestreamt wurden.

`KinesisFirehoseRecoverableServiceErrors`

Häufigkeit, mit der Datensätze erfolgreich an den Kinesis Data Firehose -Service gesendet werden konnten, aber nur mit Wiederholungsversuchen.

KinesisStream-Metrics

`KinesisStreamBytesAccepted`

Anzahl von Bytes, die während des Intervalls akzeptiert wurden.

`KinesisStreamClientLatency`

Zeitspanne, die zwischen dem Erstellen und dem Streamen des Datensatzes an den -Service für Kinesis Data Streams verstrichen ist.

`KinesisStreamLatency`

Zeitspanne, die zwischen dem Beginn und dem Ende des Datensatz-Streamings für den -Service Kinesis Data Streams verstrichen ist.

`KinesisStreamNonrecoverableServiceErrors`

Häufigkeit, mit der Datensätze trotz Neuversuche nicht fehlerfrei an den Kinesis Data Streams -Service gesendet werden konnten.

`KinesisStreamRecordsAttempted`

Anzahl von Datensätzen, bei denen das Streamen an den Kinesis Data Streams -Service versucht wurde.

`KinesisStreamRecordsFailedNonrecoverable`

Anzahl von Datensätzen, die trotz Wiederholungsversuche nicht erfolgreich an den Kinesis Data Streams -Service gestreamt wurden.

`KinesisStreamRecordsFailedRecoverable`

Anzahl von Datensätzen, die erfolgreich an den Kinesis-Datenstream-Service gestreamt wurden, aber nur mit Wiederholungsversuchen.

`KinesisStreamRecordsSuccess`

Anzahl von Datensätzen, die erfolgreich ohne Wiederholungsversuche an den Kinesis Data Streams -Service gestreamt wurden.

`KinesisStreamRecoverableServiceErrors`

Häufigkeit, mit der Datensätze erfolgreich an den Kinesis Data Streams -Service gesendet werden konnten, aber nur mit Wiederholungsversuchen.

CloudWatchLog-Metriken

`CloudWatchLogBytesAccepted`

Anzahl von Bytes, die während des Intervalls akzeptiert wurden.

`CloudWatchLogClientLatency`

Anzahl von Datensatz-Generierung und Datensatz-Streamen an den CloudWatch Logs -Service.

`CloudWatchLogLatency`

Zeitspanne, die zwischen dem Beginn und dem Ende des Datensatz-Streamings für den CloudWatch Logs -Service verstrichen ist.

`CloudWatchLogNonrecoverableServiceErrors`

Häufigkeit, mit der Datensätze trotz Wiederholungsversuche nicht fehlerfrei an den CloudWatch Logs -Service gesendet werden konnten.

`CloudWatchLogRecordsAttempted`

Anzahl von Datensätzen, bei denen das Streamen an den CloudWatch Logs -Service versucht wurde.

`CloudWatchLogRecordsFailedNonrecoverable`

Anzahl von Datensätzen, die trotz Wiederholungsversuche nicht erfolgreich an den CloudWatch Logs -Service gestreamt wurden.

`CloudWatchLogRecordsFailedRecoverable`

Anzahl von Datensätzen, die erfolgreich an den CloudWatch Logs -Service gestreamt wurden, aber nur mit Wiederholungsversuchen.

`CloudWatchLogRecordsSuccess`

Anzahl von Datensätzen, die erfolgreich ohne Wiederholungsversuche an den CloudWatch Logs -Service gestreamt wurden.

`CloudWatchLogRecoverableServiceErrors`

Häufigkeit, mit der Datensätze erfolgreich an den CloudWatch Logs -Service gesendet werden konnten, aber nur mit Wiederholungsversuchen.

CloudWatch-Metriken

`CloudWatchLatency`

Durchschnittliche Zeitspanne, die zwischen dem Beginn und dem Ende des Metrik-Streamings für den CloudWatch -Service verstrichen ist.

`CloudWatchNonrecoverableServiceErrors`

Häufigkeit, mit der Metriken trotz Wiederholungsversuche nicht fehlerfrei an den CloudWatch ch-Service gesendet werden konnten.

`CloudWatchRecoverableServiceErrors`

Häufigkeit, mit der Metriken fehlerfrei an den CloudWatch ch-Service gesendet wurden, aber nur mit Wiederholungsversuchen.

`CloudWatchServiceSuccess`

Häufigkeit, mit der Metriken fehlerfrei an den CloudWatch ch-Service gesendet wurden, ohne erforderliche Wiederholungsversuche.

Lesezeichen-Konfiguration

Standardmäßig sendet Kinesis Agent für Windows Protokolldatensätze an Senken, die nach dem Starten des Agenten erstellt werden. Manchmal ist es sinnvoll, frühere Protokolldatensätze zu senden, z. B. Protokolldatensätze, die in dem Zeitraum erstellt werden, wenn Kinesis Agent für

Windows während einer automatischen Aktualisierung gestoppt wird. Die Lesezeichen-Funktion verfolgt, welche Datensätze zu Senken gesendet wurden. Wenn Kinesis Agent für Windows im Lesezeichenmodus gestartet wird, sendet er alle Protokolldatensätze, die erstellt wurden, nachdem Kinesis Agent für Windows gestoppt wurde, zusammen mit allen anschließend erstellten Protokolldatensätzen. Zur Steuerung dieses Verhaltens können dateibasierte Quell-Deklarationen optional die folgenden Schlüssel-Wert-Paare enthalten:

`InitialPosition`

Gibt die anfängliche Situation für das Lesezeichen an. Die möglichen Werte lauten wie folgt:

`EOS`

Legt das Ende des Streams- (EOS) fest. Es werden nur Protokolldatensätze, die bei aktiven Agenten erstellt wurden, an Senken gesendet.

`0`

Alle verfügbaren Protokolldatensätze und Ereignisse werden anfänglich gesendet. Danach wird ein Lesezeichen erstellt, um sicherzustellen, dass alle Protokolldatensätze und Ereignisse, die nach dem Hinzufügen des Lesezeichens erstellt wurden, auf alle Fälle gesendet werden, egal ob Kinesis Agent für Windows aktiv ist oder ob nicht.

`Bookmark`

Das Lesezeichen wird direkt nach dem neuesten Protokolldatensatz oder Ereignis initialisiert. Danach wird ein Lesezeichen erstellt, um sicherzustellen, dass alle Protokolldatensätze und Ereignisse, die nach dem Hinzufügen des Lesezeichens erstellt wurden, auf alle Fälle gesendet werden, egal ob Kinesis Agent für Windows aktiv ist oder ob nicht.

Standardmäßig sind Lesezeichen aktiviert. Dateien werden in der Eigenschaft`%ProgramData%\Amazon\KinesisTap`-Verzeichnis.

`Timestamp`

Es werden Protokolldatensätze und Ereignisse gesendet, die nach dem Wert `InitialPositionTimestamp` (Definition folgt) erstellt wurden. Danach wird ein Lesezeichen erstellt, um sicherzustellen, dass alle Protokolldatensätze und Ereignisse, die nach dem Hinzufügen des Lesezeichens erstellt wurden, auf alle Fälle gesendet werden, egal ob Kinesis Agent für Windows aktiv ist oder ob nicht.

`InitialPositionTimestamp`

Gibt den von Ihnen gewünschten frühesten Protokolldatensatz- oder Ereigniszeitstempel an. Geben Sie dieses Schlüssel-Wert-Paar nur dann an, wenn der Parameter `InitialPosition` den Wert `Timestamp` hat.

`BookmarkOnBufferFlush`

Diese Einstellung kann beliebigen Lesezeichenquellen hinzugefügt werden. Bei Einstellung auf`true`, stellt sicher, dass Lesezeichenaktualisierungen nur durchgeführt werden, wenn eine Senke erfolgreich ein Ereignis an AWS versendet. Sie können nur eine einzelne Senke für eine Quelle abonnieren. Wenn Sie Protokolle an mehrere Ziele senden, duplizieren Sie Ihre Quellen, um potenzielle Probleme mit Datenverlust zu vermeiden.

Wenn Kinesis Agent für Windows für eine lange Zeit angehalten wurde, müssen diese Lesezeichen möglicherweise gelöscht werden, da die mit einem Lesezeichen versehenen Protokolldatensätze und Ereignisse möglicherweise nicht mehr vorhanden sind. Lesezeichendateien für eine bestimmte Quell-ID befinden sich in `%PROGRAMDATA%\Amazon\AWSKinesisTap\`*`source id`*`.bm`.

Lesezeichen funktionieren nicht bei Dateien, die umbenannt oder abgeschnitten wurden. ETW-Ereignisse und Leistungsindikatoren können aufgrund ihrer Beschaffenheit nicht mit einem Lesezeichen versehen werden.

Senken-Deklarationen

Senken-Deklarationen geben an, wo und in welcher Form Protokolle, Ereignisse und Metriken an verschiedene AWS-Services gesendet werden sollen. Die folgenden Abschnitte beschreiben Konfigurationen für die integrierten Senken-Typen, die in Amazon Kinesis Agent für Microsoft Windows verfügbar sind. Da Kinesis Agent für Windows erweiterbar ist, können Sie benutzerdefinierte Senkentypen hinzufügen. Jeder Senkentyp erfordert in der Regel eindeutige Schlüssel-Wert-Paare in den Konfigurationsdeklarationen, die für diesen Senkentyp relevant sind.

Alle Senken-Deklarationen können die folgenden Schlüssel-Wert-Paare enthalten:

`Id`

Eine eindeutige Zeichenfolge, die eine bestimmtes Senke innerhalb der Konfigurationsdatei identifiziert (erforderlich).

`SinkType`

Der Name des Senkentyps für diese Senke (erforderlich). Der Senkentyp gibt das Ziel der Protokoll-, Ereignis- oder Metrikdaten an, die von dieser Senke gestreamt werden.

`AccessKey`

Gibt den AWS Zugriffsschlüssel an, der bei der Autorisierung des Zugriffs auf den AWS-Service verwendet werden soll, der diesem Senkentyp zugeordnet ist. Dieses Schlüssel-Wert-Paar ist optional. Weitere Informationen finden Sie unter Senken-Sicherheitskonfiguration.

`SecretKey`

Gibt den geheimen AWS schlüssel an, der bei der Autorisierung des Zugriffs auf den AWS-Service verwendet werden soll, der diesem Senkentyp zugeordnet ist. Dieses Schlüssel-Wert-Paar ist optional. Weitere Informationen finden Sie unter Senken-Sicherheitskonfiguration.

`Region`

Gibt an, in welcher AWS-Region sich die Ziel-Ressourcen für das Streaming befinden. Dieses Schlüssel-Wert-Paar ist optional.

`ProfileName`

Gibt an, welches AWS-Profil für die Authentifizierung verwendet wird. Dieses Schlüssel-Wert-Paar ist optional. Wenn es verwendet wird, überschreibt es jedoch alle angegebenen Zugriffsschlüssel und geheimen Schlüssel. Weitere Informationen finden Sie unter Senken-Sicherheitskonfiguration.

`RoleARN`

Gibt die IAM-Rolle an, die beim Zugriff auf den AWS -Service verwendet werden soll, der dem Senkentyp zugeordnet ist. Diese Option ist nützlich, wenn Kinesis Agent für Windows auf einer EC2-Instance ausgeführt wird, eine andere Rolle aber besser geeignet wäre, als die Rolle, auf die das Instance-Profil verweist. Beispiel: Eine kontoübergreifende Rolle kann verwendet werden, um auf Ressourcen abzuzielen, die sich nicht in demselben AWS-Konto wie die EC2-Instance befinden. Dieses Schlüssel-Wert-Paar ist optional.

`Format`

Gibt die Art der Serialisierung an, die vor dem Streaming auf Protokolle und Ereignisdaten angewendet wird. Gültige Werte sind `json` und `xml`. Diese Option ist hilfreich, wenn für Downstream-Analysen in der Daten-Pipeline Daten in einem bestimmten Format erforderlich sind oder bevorzugt werden. Dieses Schlüssel-Wert-Paar ist optional. Wenn es nicht angegeben wird,

wird normaler Text aus der Quelle von der Senke zum AWS-Service gestreamt, der der Senke zugeordnet ist.

`TextDecoration`

Wenn kein `Format` angegeben wird, legt `TextDecoration` fest, welcher zusätzlichen Text eingeschlossen werden sollte, wenn Protokoll- oder Ereignisdatensätze gestreamt werden. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen. Dieses Schlüssel-Wert-Paar ist optional.

`ObjectDecoration`

Wenn angegeben `Format` wird, legt `ObjectDecoration` fest, welche zusätzlichen Daten vor der Serialisierung und dem Streaming in den Protokoll- oder Ereignisdatensatz eingeschlossen werden. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen. Dieses Schlüssel-Wert-Paar ist optional.

`BufferInterval`

Um API-Aufrufe des dem Senkentyp zugeordneten AWS-Service auf ein Minimum zu reduzieren, puffert Kinesis Agent für Windows mehrere Protokoll-, Ereignis- oder Metrik-Datensätze vor dem Streamen. Dies kann bei Services Kosten sparen, die pro API-Aufruf eine Gebühr berechnen. `BufferInterval` gibt die maximale Zeitspanne (in Sekunden) an, über die hinweg Datensätze gepuffert werden sollten, bevor sie an den AWS-Service gestreamt werden. Dieses Schlüssel-Wert-Paar ist optional. Falls es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Zeichenfolge.

`BufferSize`

Um API-Aufrufe des dem Senkentyp zugeordneten AWS-Service auf ein Minimum zu reduzieren, puffert Kinesis Agent für Windows mehrere Protokoll-, Ereignis- oder Metrik-Datensätze vor dem Streamen. Dies kann bei Services Kosten sparen, die pro API-Aufruf eine Gebühr berechnen. `BufferSize` gibt die maximale Anzahl von Datensätzen an, die vor dem Streamen an den AWS-Service gepuffert werden sollen. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Zeichenfolge.

`MaxAttempts`

Gibt an, wie oft maximal versucht Kinesis einen Satz von Protokoll-, Ereignis- und Metrik-Datensätzen an einen AWS -Service zu streamen, wenn beim Streaming konsistent Fehler auftreten. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Zeichenfolge. Der Standardwert lautet "3".

Beispiele für vollständige Konfigurationsdateien, die verschiedene Arten von Senken verwenden, finden Sie unter Streamen aus dem Windows-Anwendungsereignisprotokoll an Senken.

Themen

`KinesisStream`-Senken-Konfiguration

Die`KinesisStream`Der -Senkentyp streamt Protokolldatensätze und Ereignisse an den Kinesis Data Streams -Service. Daten, die an Kinesis Data Streams gestreamt werden, werden üblicherweise von einer oder mehreren benutzerdefinierten Anwendungen verarbeitet, die mit verschiedenen AWS -Services ausgeführt werden. Die Daten werden an einen benannten Stream gestreamt, der mit Kinesis Data Streams konfiguriert wird. Weitere Informationen finden Sie im .Amazon Kinesis Data Streams Entwicklerhandbuch.

Es folgt ein Beispiel für eine -Senken-Deklaration von Kinesis Data Streams:

```
{
    "Id": "TestKinesisStreamSink",
    "SinkType": "KinesisStream",
```

```
    "StreamName": "MyTestStream",
    "Region": "us-west-2"
}
```

Alle `KinesisStream`-Senken-Deklarationen können die folgenden zusätzlichen Schlüssel-Wert-Paare angeben:

`SinkType`

Muss angegeben werden, und der Wert muss die Literalzeichenfolge `KinesisStream` sein.

`StreamName`

Gibt den Namen des Kinesis Daten-Streams an, der die vom`KinesisStream`Spüle Typ (erforderlich). Bevor Sie die Daten streamen, konfigurieren Sie den Stream in der AWS Management Console, der AWS-CLI oder über eine Anwendung unter Verwendung der Kinesis Data Streams -API.

`RecordsPerSecond`

Gibt die maximale Anzahl Kinesis Data Streams an. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Ganzzahl. Der Standardwert ist 1000 Datensätze.

`BytesPerSecond`

Gibt die maximale Anzahl der pro Sekunde Kinesis Data Streams. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Ganzzahl. Der Standardwert lautet 1 MB.

Der Standardwert `BufferInterval` für diesen Senkentyp beträgt 1 Sekunde, und der Standardwert `BufferSize` ist 500 Datensätze.

`KinesisFirehose`-Senken-Konfiguration

Die`KinesisFirehose`Der -Senkentyp streamt Protokolldatensätze und Ereignisse an den Kinesis Data Firehose -Service. Kinesis Data Firehose stellt die gestreamten Daten für die Speicherung an andere Services bereit. Die gespeicherten Daten werden dann gewöhnlich in nachfolgenden Phasen der Daten-Pipeline analysiert. Die Daten werden an einen benannten Bereitstellungs-Stream gestreamt, der mit Kinesis Data Firehose konfiguriert wird. Weitere Informationen finden Sie im .Amazon Kinesis Data Firehose Entwicklerhandbuch.

Es folgt ein Beispiel für eine Kinesis Data Firehose Deklaration:

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

Alle `KinesisFirehose`-Senken-Deklarationen können die folgenden zusätzlichen Schlüssel-Wert-Paare angeben:

`SinkType`

Muss angegeben werden, und der Wert muss die Literalzeichenfolge `KinesisFirehose` sein.

`StreamName`

Gibt den Namen des Kinesis Data Firehose Bereitstellungs-Streams an, der die vom`KinesisStream`Spüle Typ (erforderlich). Bevor Sie die Daten streamen, konfigurieren Sie den Bereitstellungs-Stream mithilfe der AWS Management Console, der AWS-CLI oder über eine Anwendung unter Verwendung der Kinesis Data Firehose -API.

`CombineRecords`

Wenn auf`true`, legt fest, dass mehrere kleine Datensätze zu einem großen Datensatz mit einer maximalen Größe von 5 KB kombiniert werden sollen. Dieses Schlüssel-Wert-Paar ist optional. Datensätze, die mit dieser Funktion kombiniert werden, werden durch`\n`. Wenn Sie AWS Lambda zum Umwandeln eines Kinesis Data Firehose Datensatzes verwenden, muss Ihre Lambda Funktion das Trennzeichen berücksichtigen.

`RecordsPerSecond`

Gibt die maximale Anzahl der pro Sekunde Kinesis Data Streams. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Ganzzahl. Der Standardwert ist 5000 Datensätze.

`BytesPerSecond`

Gibt die maximale Anzahl der pro Sekunde Kinesis Data Streams. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Ganzzahl. Der Standardwert lautet 5 MB.

Der Standardwert `BufferInterval` für diesen Senkentyp beträgt 1 Sekunde, und der Standardwert `BufferSize` ist 500 Datensätze.

Konfiguration der CloudWatch Senken-Konfiguration

Die`CloudWatch`Senkentyp streamt Metriken an den CloudWatch Service. Sie können die Metriken in der AWS Management Console anzeigen. Weitere Informationen finden Sie im Amazon CloudWatch-Benutzerhandbuch.

Es folgt ein Beispiel für eine `CloudWatch`-Senken-Deklaration:

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

Alle `CloudWatch`-Senken-Deklarationen können die folgenden zusätzlichen Schlüssel-Wert-Paare angeben:

`SinkType`

Muss angegeben werden, und der Wert muss die Literalzeichenfolge `CloudWatch` sein.

`Interval`

Gibt an, wie häufig (in Sekunden) Kinesis Agent für Windows Metriken an den CloudWatch Dienst meldet. Dieses Schlüssel-Wert-Paar ist optional. Wenn es angegeben wird, verwenden Sie zum Repräsentieren des Wertes eine Ganzzahl. Der Standardwert liegt bei 60 Sekunden. Geben Sie 1 Sekunde an, wenn Sie hochauflösende CloudWatch Metriken wünschen.

`Namespace`

Gibt den CloudWatch Namespace an, in dem die Metrikdaten gemeldet werden. CloudWatch Namespaces gruppieren einen Satz von Metriken zusammen. Dieses Schlüssel-Wert-Paar ist optional. Der Standardwert ist `KinesisTap`.

`Dimensions`

Gibt die CloudWatch Dimensionen an, die verwendet werden, um Metrikgruppen in einem Namespace zu isolieren. Dies kann nützlich sein, um z. B. für jeden Desktop oder Server separate Sätze von Metrikdaten bereitzustellen. Dieses Schlüssel-

Wert-Paar ist optional. Wenn es angegeben wird, muss der Wert das folgende Format aufweisen: "Schlüssel1=Wert1;Schlüssel2=Wert2...". Der Standardwert ist `"ComputerName={computername};InstanceId={instance_id}"`. Dieser Wert unterstützt die Ersetzung von Senken-Variablen. Weitere Informationen finden Sie unter Konfigurieren von Senken-Variablensubstitutionen.

`MetricsFilter`

Gibt an, welche Metriken aus der integrierten Kinesis Agent für Windows--Metrikenquelle an CloudWatch an. Weitere Informationen zur integrierten Kinesis Agent für Windows-Metrikenquelle, einschließlich der Details der Syntax des Wertes dieses Schlüssel-Wert-Paars finden Sie unterIntegrierte Quelle von Kinesis Agent für Windows -Metriken: Quelle.

`CloudWatchLogs`-Senken-Konfiguration

Die`CloudWatchLogs`Der -Senkentyp streamt Protokolldatensätze und Ereignisse an Amazon CloudWatch Logs. Sie können Protokolle in der AWS Management Console anzeigen oder sie über zusätzliche Phasen einer Daten-Pipeline verarbeiten. Die Daten werden an einen benannten Protokollstream gestreamt, der in CloudWatch Logs konfiguriert ist. Protokollstreams sind als benannte Protokollgruppen angeordnet. Weitere Informationen finden Sie im .Benutzerhandbuch für Amazon CloudWatch Logs.

Es folgt ein Beispiel für eine CloudWatch Logs enken-Deklaration:

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

Alle `CloudWatchLogs`-Senken-Deklarationen müssen die folgenden zusätzlichen Schlüssel-Wert-Paare angeben:

`SinkType`

Muss die Literalzeichenfolge `CloudWatchLogs` sein.

`LogGroup`

Gibt den Namen der CloudWatch Logs Protokollgruppe an, die den Protokollstream enthält, der die vom der gestreamten Protokoll- und Ereignisdatensätze erhält.`CloudWatchLogs`Spüle Typ. Wenn die angegebene Protokollgruppe nicht vorhanden Kinesis versucht, sie zu erstellen.

`LogStream`

Gibt den Namen des CloudWatch Logs Streams an, der den Protokoll- und Ereignisdatensätze erhält.`CloudWatchLogs`Spüle Typ. Dieser Wert unterstützt die Ersetzung von Senken-Variablen. Weitere Informationen finden Sie unter Konfigurieren von Senken-Variablensubstitutionen. Wenn der angegebene Protokollstream nicht vorhanden Kinesis versucht, ihn zu erstellen.

Der Standardwert `BufferInterval` für diesen Senkentyp beträgt 1 Sekunde, und der Standardwert `BufferSize` ist 500 Datensätze. Die maximale Puffergröße ist 10.000 Datensätze.

Local**`FileSystem`**-Senken-Konfiguration

Der Typ der Spüle`FileSystem`speichert Protokoll- und Ereignisdatensätze in einer Datei im lokalen Dateisystem, anstatt sie an AWS -Services zu streamen.`FileSystem`Senken sind nützlich für Tests und Diagnostik. Sie können diesen Senken-Typ beispielsweise verwenden, um Datensätze zu untersuchen, bevor Sie sie an AWS senden.

mit`FileSystem`Senken können Sie auch Konfigurationsparameter verwenden, um Batching, Drosselung und Retry-on-Fehler zu simulieren, um das Verhalten der tatsächlichen AWS S-Senks nachzuahmen.

Alle Datensätze aus allen Quellen, die mit einem`FileSystem`sinken werden in der einzelnen Datei gespeichert, die als`FilePath`. Wenn`FilePath`Wenn nichts angegeben ist, werden Datensätze in eine Datei mit dem Namen gespeichert.*`SinkId`*`.txt`im`%TEMP%`Verzeichnis, das in der Regel`C:\Users\`*`UserName`*`\AppData\Local\Temp`wobei*`SinkId`*ist der eindeutige Bezeichner der Senken-und*`UserName`*ist der Windows-Benutzername des aktiven Benutzers.

Dieser Senkentyp unterstützt Textdekorationsattribute. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen.

Ein Beispiel`FileSystem`Die -Senkentyp wird im folgenden Beispiel veranschaulicht.

```
{
```

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

Die`FileSystem`Diese Konfiguration besteht aus den folgenden Schlüssel-Wert-Paaren.

`SinkType`

Muss die Literalzeichenfolge `FileSystem` sein.

`FilePath`

Gibt den Pfad und die Datei an, in der Datensätze gespeichert werden. Dieses Schlüssel-Wert-Paar ist optional. Wenn nichts angegeben ist, ist der Standardwert*TempPath*\ *SinkId*`.txt`wobei*TempPath*ist der Ordner, der im`%TEMP%`Variable und*SinkId*ist der eindeutige Bezeichner der Senke.

`Format`

Gibt das Format des Ereignisses`json`oder `.xml`. Dieses Schlüssel-Wert-Paar ist optional und wird nicht zwischen Groß-/Kleinschreibung unterschieden Wenn nicht angegeben, werden Ereignisse im Nur-Text in die Datei geschrieben.

`TextDecoration`

Gilt nur für Ereignisse, die im Nur-Text geschrieben wurden. Dieses Schlüssel-Wert-Paar ist optional.

`ObjectDecoration`

Gilt nur für Ereignisse, bei denen`Format`wird auf gesetzt`json`. Dieses Schlüssel-Wert-Paar ist optional.

Erweiterte Nutzung — Aufzeichnung von Drosselung und Fehlersimulation

`FileSystem`kann das Verhalten von AWS S-Senks nachahmen, indem die Datensatzdrosselung simuliert wird. Sie können die folgenden Schlüssel-Wert-Paare verwenden, um Datensatzeinschränkungs- und Fehler-Simulationsattribute anzugeben.

Indem Sie eine Sperre für die Zieldatei anlegen und Schreibvorgänge verhindern, können Sie`FileSystem`Senken, um das Verhalten von AWS S-Senken zu simulieren und zu untersuchen, wenn das Netzwerk ausfällt.

Das folgende Beispiel zeigt einen`FileSystem`Konfiguration mit Simulationsattribute.

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

`RequestsPerSecond`

Optional und als String-Typ angegeben. Wenn dies nicht angegeben wird, ist der Standardwert`"5"`. Steuert die Rate der Anforderungen, die die Senke verarbeitet, d. h. in eine Datei schreibt, nicht die Anzahl der Datensätze. Kinesis Agent für Windows stellt Batch-Anforderungen an AWS Endpunkte, sodass eine Anforderung mehrere Datensätze enthält.

`BufferSize`

Optional und als String-Typ angegeben. Gibt die maximale Anzahl von Ereignisdatensätzen an, die die Stapel vor dem Speichern in einer Datei versenken.

`MaxBatchSize`

Optional und als String-Typ angegeben. Gibt die maximale Menge an Ereignisdatensatzdaten in Byte an, die die Stapel vor dem Speichern in Datei versenken.

Die maximale Datensatzratengrenze ist eine Funktion von`BufferSize`Gibt die maximale Anzahl der pro Anforderung an.`RequestsPerSecond`. Sie können die Datensatzratengrenze pro Sekunde mit der folgenden Formel berechnen.

Recordrate=`BufferSize`*`RequestsPerSecond`

Bei Konfigurationswerten im obigen Beispiel gibt es eine maximale Datensatzrate von 1000 Datensätzen pro Sekunde.

Senken-Sicherheitskonfiguration

Konfigurieren der Authentifizierung

Damit Kinesis Agent für Windows Protokolle, Ereignisse und Metriken an AWS -Services streamt, muss der Zugriff authentifiziert werden. Es gibt verschiedene Möglichkeiten, Authentifizierung für Kinesis Agent für Windows bereitzustellen. Die jeweilige Methode hängt von der Situation ab, in der Kinesis Agent für Windows ausgeführt wird, und von den spezifischen Sicherheitsanforderungen für eine bestimmte Organisation.

- Wenn Kinesis Agent für Windows auf einem Amazon EC2 Host ausgeführt wird, besteht die sicherste und einfachste Methode zur Bereitstellung von Authentifizierung darin, eine IAM-Rolle mit ausreichendem Zugriff auf die erforderlichen Operationen für die erforderlichen AWS -Services zu erstellen sowie ein EC2-Instance-Profil, mit dem auf diese Rolle verwiesen wird. Weitere Informationen zum Erstellen von Instance-Profilen finden Sie unter Verwenden von Instance-Profilen. Weitere Informationen zu den an die IAM-Rolle anzufügenden Richtlinien finden Sie unterKonfigurieren der Autorisierung.

  Nachdem Sie das Instance-Profil erstellt haben, können Sie es beliebigen EC2-Instances zuordnen, die Kinesis Agent für Windows verwenden. Wenn Instances bereits ein Instance-Profil zugeordnet ist, können Sie die entsprechenden Richtlinien an die Rolle anfügen, die dem Instance-Profil zugewiesen ist.
- Wenn Kinesis Agent für Windows auf einem EC2-Host in einem Konto ausführt, die Ressourcen, die Ziel der Senke sind, sich aber in einem anderen Konto befinden, können Sie eine IAM-Rolle für kontoübergreifenden Zugriff erstellen. Weitere Informationen finden Sie unterTutorial: Delegieren des Zugriffs in allen AWS Konten mithilfe von IAM-Rollen. Nachdem Sie die kontoübergreifende Rolle erstellt haben, geben Sie den Amazon-Ressourcennamen (ARN) für die kontoübergreifende Rolle als Wert der`RoleARN`-Schlüssel-Wert-Paar in der Senken-Deklaration. Kinesis Agent für Windows versucht dann beim Zugriff auf AWS Ressourcen, die dem Senkentyp dieser Senke zugeordnet sind, die angegebene kontoübergreifende Rolle anzunehmen.
- Wenn Kinesis Agent für Windows außerhalb von Amazon EC2 ausführt wird (z. B. lokal), sind mehrere Optionen vorhanden:
  - Wenn es akzeptabel ist, den lokalen Server oder Desktop als eine von Amazon EC2 Systems Manager verwaltete Instances zu registrieren, verfahren Sie zum Konfigurieren der Authentifizierung wie folgt:

1. Verwenden Sie den unter Einrichten von AWS Systems Manager in Hybridumgebungen beschriebenen Vorgang, um eine Servicerolle zu erstellen, eine Aktivierung für eine verwaltete Instance zu erstellen und den SSM-Agenten zu installieren.
2. Fügen Sie der Servicerolle die entsprechenden Richtlinien an, damit Kinesis Agent für Windows auf die Ressourcen zugreifen kann, die zum Streamen von Daten aus den konfigurierten Senken benötigt werden. Weitere Informationen zu den an die IAM-Rolle anzufügenden Richtlinien finden Sie unterKonfigurieren der Autorisierung.
3. Verwenden Sie den unterKonfigurierenProfileRefreshingAWSCredentialProviderSoaktualisieren Sie AWS Credentialsum AWS Anmeldeinformationen zu aktualisieren.

Dies ist die empfohlene Vorgehensweise für Nicht-EC2-Instances, da Anmeldeinformationen sicher von SSM und AWS verwaltet werden.

- Wenn es akzeptabel ist, den AWSKinesisTap-Service für Kinesis Agent für Windows unter einem bestimmten Benutzer anstelle des Standard-Systemkontos auszuführen, führen Sie das folgende Verfahren aus:

  1. Erstellen Sie einen IAM-Benutzer in dem AWS Konto, in dem die AWS-Services verwendet werden sollen. Erfassen Sie während des Erstellungsvorgangs den Zugriffsschlüssel und den geheimen Schlüssel des Benutzers. Diese Informationen werden später in diesem Verfahren benötigt.
  2. Fügen Sie Richtlinien an den IAM-Benutzer an, die den Zugriff auf die erforderlichen Vorgänge für die erforderlichen Services autorisieren. Weitere Informationen zu den an den IAM-Benutzer anzufügenden Richtlinien finden Sie unterKonfigurieren der Autorisierung.
  3. Ändern Sie den AWSKinesisTap-Service auf jedem Desktop oder Server so ab, dass er anstatt unter dem Standard-Systemkonto unter einem bestimmten Benutzer ausgeführt wird.
  4. Erstellen Sie mittels des zuvor notierten Zugriffsschlüssels und geheimen Schlüssels ein Profil im SDK Store. Weitere Informationen finden Sie unter Konfigurieren der AWS-Anmeldeinformationen.
  5. Aktualisieren Sie die Datei `AWSKinesisTap.exe.config` im Verzeichnis `%PROGRAMFILES%\Amazon\AWSKinesisTap` so, dass der Name des im vorherigen Schritt erstellten Profils angegeben wird. Weitere Informationen finden Sie unter Konfigurieren der AWS-Anmeldeinformationen.

  Dies ist die empfohlene Vorgehensweise für Nicht-EC2-Hosts, bei denen es sich nicht um verwaltete Instances handeln kann, da die Anmeldeinformationen für den spezifischen Host und den spezifischen Benutzer verschlüsselt sind.

- Wenn es erforderlich ist, den AWSKinesisTap-Service für Kinesis Agent für Windows unter dem Standard-Systemkonto auszuführen, müssen Sie eine freigegebene Datei mit den Anmeldeinformationen verwenden. Grund hierfür ist, dass das Systemkonto über kein Windows-Benutzerprofil zur Freigabe des SDK Store verfügt. Da freigegebene Anmeldeinformationsdateien nicht verschlüsselt sind, ist diese Vorgehensweise nicht zu empfehlen. Weitere Informationen zur Verwendung von freigegebenen Konfigurationsdateien finden Sie unterKonfigurieren der AWS-AnmeldeinformationenimAWS SDK für .NET. Wenn Sie diese Methode verwenden, empfehlen wir die Verwendung von NTFS-Verschlüsselung und Beschränkung des Dateizugriffs auf die freigegebene Konfigurationsdatei. Die Schlüssel sollten von einer Verwaltungsplattform rotiert werden. Im Falle einer Schlüsselrotation muss die freigegebene Konfigurationsdatei aktualisiert werden.

Obwohl es möglich ist, direkt in der Senken-Deklarationen Zugriffsschlüssel und geheime Schlüssel bereitzustellen, wird hiervon abgeraten, da die Deklarationen nicht verschlüsselt sind.

Konfigurieren der Autorisierung

Fügen Sie die entsprechenden nachstehenden Richtlinien dem IAM-Benutzer oder der Rolle von an, die Kinesis Agent für Windows zum Streamen von Daten an AWS -Services verwendet:

Kinesis Data Streams

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

Um die Autorisierung auf eine bestimmte Region, ein bestimmtes Konto oder einen bestimmten Stream-Namen zu begrenzen, ersetzen Sie das entsprechende Sternchen im ARN durch bestimmte

Werte. Weitere Informationen finden Sie unter „Amazon-Ressourcennamen (ARNs) für Kinesis Data Streams“ unter Steuern des Zugriffs auf Amazon Kinesis Data Streams-Ressourcen mithilfe von IAM.

Kinesis Data Firehose

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

Um die Autorisierung auf eine bestimmte Region, ein bestimmtes Konto oder einen bestimmten Bereitstellungs-Stream-Namen zu begrenzen, ersetzen Sie das entsprechende Sternchen im ARN durch bestimmte Werte. Weitere Informationen finden Sie unterSteuern des Zugriffs mit Amazon Kinesis Data FirehoseimAmazon Kinesis Data Firehose Entwicklerhandbuch.

CloudWatch

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

Weitere Informationen finden Sie unterÜbersicht über die Verwaltung von Zugriffsberechtigungen für Ihre CloudWatch RessourcenimBenutzerhandbuch für Amazon CloudWatch Logs.

CloudWatch Logs mit einer vorhandenen Protokollgruppe und einem vorhandenen Protokollstream

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

Um den Zugriff auf eine bestimmte Region, ein bestimmtes Konto, eine bestimmte Protokollgruppe oder einen bestimmten Protokollstream einzuschränken, ersetzen Sie das entsprechende Sternchen in die ARNs durch die entsprechenden Werte. Weitere Informationen finden Sie unterÜbersicht über die Verwaltung von Zugriffsberechtigungen für Ihre CloudWatch Logs RessourcenimBenutzerhandbuch für Amazon CloudWatch Logs.

CloudWatch Logs mit zusätzlichen Berechtigungen für Kinesis Agent für Windows zum Erstellen von Protokollgruppen und Protokollstreams

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

Um den Zugriff auf eine bestimmte Region, ein bestimmtes Konto, eine bestimmte Protokollgruppe oder einen bestimmten Protokollstream einzuschränken, ersetzen Sie das entsprechende Sternchen in die ARNs durch die entsprechenden Werte. Weitere Informationen finden Sie unterÜbersicht über die Verwaltung von Zugriffsberechtigungen für Ihre CloudWatch Logs RessourcenimBenutzerhandbuch für Amazon CloudWatch Logs.

Erforderliche Berechtigungen für die EC2-Tag-Variablen-Erweiterung

Für die Verwendung der Variablen-Erweiterung mit dem `ec2tag`-Variablen-Präfix wird die Berechtigung `ec2:Describe*` benötigt.

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

```
}
```

> **Note**
>
> Sie können mehrere Anweisungen in einer einzigen Richtlinie kombiniert, solange die `Sid` für jede Anweisung in dieser Richtlinie eindeutig ist. Weitere Informationen zum Erstellen von Richtlinien finden Sie unterErstellen von IAM-RichtlinienimIAM-Benutzerhandbuch.

KonfigurierenProfileRefreshingAWSCredentialProviderSo aktualisieren Sie AWS Credentials

Wenn Sie AWS Systems Manager für hybride Umgebungen zum Verwalten von AWS-Anmeldeinformationen verwenden, dreht Systems Manager die Sitzungsanmeldeinformationen in`c:\Windows\System32\config\systemprofile\.aws\credentials`. Weitere Informationen zu Systems Manager für Hybridumgebungen finden Sie unterEinrichten von AWS Systems Manager für hybride UmgebungenimAWS Systems Manager Benutzerhandbuch..

Da das AWS .net-SDK keine neuen Anmeldeinformationen automatisch abruft, stellen wir die`ProfileRefreshingAWSCredentialProvider`, um die Anmeldeinformationen zu aktualisieren.

Sie können das`CredentialRef`-Attribut jeder AWS -Synchronisierungskonfiguration verwenden, um auf ein`Credentials`Definition, bei der die`CredentialType`Attribut ist auf`ProfileRefreshingAWSCredentialProvider`Siehe folgendes Beispiel.

```
{
    "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
    }]
}
```

Eine Anmeldeinformationen besteht aus den folgenden Attributen als Schlüssel-Wert-Paare.

`Id`

Definiert die Zeichenfolge, die Senkendefinitionen mit`CredentialRef`, um auf diese Konfiguration der Anmeldeinformationen zu verweisen

`CredentialType`

Legen Sie auf die Literalzeichenfolge fest`ProfileRefreshingAWSCredentialProvider`.

`Profile`

Optional. Der Standardwert ist `default`.

`FilePath`

Optional. Gibt den Pfad zur AWS Anmeldeinformationsdatei an. Wenn nichts angegeben ist, wird standardmäßig `%USERPROFILE%/.aws/credentials` verwendet.

`RefreshingInterval`

Optional. Die Häufigkeit, mit der Anmeldeinformationen aktualisiert werden, in Sekunden. Wenn nichts angegeben ist, wird standardmäßig `300` verwendet.

Konfigurieren von Senken-Ausstattungen

Senken-Deklarationen können optional Schlüssel-Wert-Paare enthalten, die zusätzliche Daten zum Streamen an verschiedene AWS-Services angeben, um die aus der Quelle erfassten Datensätze zu optimieren.

`TextDecoration`

Verwenden Sie dieses Schlüssel-Wert-Paar, wenn in der Senken-Deklaration kein `Format` angegeben wird. Der Wert ist eine spezielle Formatzeichenfolge, bei der Variablensubstitution auftritt. Angenommen, für eine Senke wird als `TextDecoration` `"{ComputerName}:::{timestamp:yyyy-MM-dd HH:mm:ss}:::{_record}"` angegeben. Wenn eine Quelle einen Protokolldatensatz mit dem Text `The system has resumed from sleep.` ausgibt und diese Quelle über eine Pipe mit der Senke verbunden ist, dann wird der Text

`MyComputer1:::2017-10-26 06:14:22:::The system has resumed from sleep.` an den AWS-Service gestreamt, der dem Senken-Typ zugeordnet ist. Die Variable `{_record}` verweist auf den aus der Quelle bereitgestellten ursprüngliche Text-Datensatz.

`ObjectDecoration`

Verwenden Sie dieses Schlüssel-Wert-Paar, wenn in der Senken-Deklaration `Format` angegeben wird, um vor der Datensatzserialisierung zusätzliche Daten hinzuzufügen. Angenommen, für eine Senke, die als `Format` JSON angibt, wird als `ObjectDecoration` `"ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd HH:mm:ss}"` angegeben. Das sich ergebende JSON-Streaming an den AWS-Service, der dem Senken-Typ zugeordnet ist, enthält zusätzlich zu den ursprünglichen Daten aus der Quelle die folgenden Schlüssel-Wert-Paare:

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

Ein Beispiel für die Verwendung von `ObjectDecoration`finden Sie unter Tutorial: Streamen von JSON-Protokolldateien mit Kinesis Agent für Windows zu Amazon S3.

`ObjectDecorationEx`

Gibt einen Ausdruck an, der eine flexiblere Datenextraktion und -formatierung im Vergleich zu`ObjectDecoration`. Dieses Feld kann verwendet werden, wenn das Format der Senke`json`. Im folgenden Beispiel wird die Ausdruckssyntax veranschaulicht.

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

Beispielsweise können Sie die folgenden`ObjectDecorationEx`Attribut:

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

Transformiert den Literaldatensatz:

```
System log message
```

In ein JSON-Objekt wie folgt mit den Werten, die von den Ausdrücken zurückgegeben werden:

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

Weitere Informationen zum Formulieren von Ausdrücken finden Sie unterTipps für das Schreiben von Ausdrücken. Die meisten der`ObjectDecoration`-Deklaration sollte mit der neuen Syntax mit Ausnahme von Zeitstempel-Variablen funktionieren. A`{timestamp:yyyyMMdd}`-Feld im`ObjectDecoration`wird als`{format(_timestamp,'yyyyMMdd')}`in`ObjectDecorationEx`.

`TextDecorationEx`

Gibt einen Ausdruck an, der eine flexiblere Datenextraktion und -formatierung im Vergleich zu`TextDecoration`Siehe folgendes Beispiel.

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

Sie können`TextDecorationEx`, um JSON-Objekte zu erstellen. Verwenden Sie '@ {', um offene geschweifte Klammer zu entkommen, wie im folgenden Beispiel gezeigt.

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

Wenn der Quelltyp der Quelle, die mit der Senke verbunden ist, `DirectorySource` lautet, kann die Senke drei zusätzliche Variablen verwenden:

`_FilePath`

Der vollständige Pfad zur Protokolldatei.

`_FileName`

Der Dateiname und die Namenserweiterung der Datei.

`_Position`

Eine Ganzzahl, die angibt, wo in der Protokolldatei sich der Datensatz befindet.

Diese Variablen sind nützlich, wenn Sie eine Quelle verwenden, die Protokolldatensätze aus mehreren Dateien sammelt, die mit einer Senke verbunden sind, die alle Datensätze an einen einzigen Stream streamt. Da die Werte dieser Variablen in die Streaming-Datensätze eingefügt werden, können Downstream-Analysen in der Daten-Pipeline die Datensätze nach Datei und nach Position in jeder anordnen.

Tipps für das Schreiben von Ausdrücken

Ein Ausdruck kann einer der folgenden Möglichkeiten sein:

- Ein variabler Ausdruck.
- Ein konstanter Ausdruck, z. B.`'hello'`,`1`,`1.21`,`null`,`true`,`false`.
- Ein Aufrufausdruck, der eine -Funktion aufruft, wie im folgenden Beispiel gezeigt.

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

Sonderzeichen

Zwei umgekehrte Schrägstriche sind erforderlich, um Sonderzeichen zu entkommen.

Nesting

Funktionsaufrufe können verschachtelt werden, wie im folgenden Beispiel gezeigt.

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

Variables

Es gibt drei Arten von Variablen: lokal, meta und global.

- Lokale VariablenBeginnen Sie mit einemwie`message`. Sie werden verwendet, um die Eigenschaft des Ereignisobjekts aufzulösen, einen Eintrag, wenn das Ereignis ein Wörterbuch ist, oder ein Attribut, wenn das Ereignis ein JSON-Objekt ist. Wenn die lokale Variable Leerzeichen oder Sonderzeichen enthält, verwenden Sie eine lokale Variable in Anführungszeichen wie$`'date created'`.
- Metadaten VariablenBeginnen Sie mit einem Unterstrich (_) und werden verwendet, um die Metadaten des Ereignisses aufzulösen. Alle Ereignistypen unterstützen die folgenden Meta-Variablen.

`_timestamp`

Der Zeitstempel des Ereignisses.

`_record`

Darstellung des Ereignisses im Rohtext.

Protokollereignisse unterstützen die folgenden zusätzlichen Meta-Variablen.

`_filepath`

`_filename`

`_position`

`_linenumber`

- Globale Variablenin Umgebungsvariablen, EC2-Instanzmetadaten oder EC2Tag auflösen. Zur Verbesserung der Leistung empfehlen wir, dass Sie das Präfix verwenden, um den Suchbereich einzuschränken, z. B.`{env:ComputerName}`,`{ec2:InstanceId}`, und`{ec2tag:Name}`.

Integrierte Funktionen

Der Kinesis Agent für Windows unterstützt die folgenden integrierten Funktionen. Wenn eines der ArgumenteNULLund die Funktion ist nicht für den Umgang mitNULL, einNULL-Objekt zurückgegeben wird.

```
//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)
```

Konfigurieren von Senken-Variablensubstitutionen

Die `KinesisStream`-, `KinesisFirehose`- und `CloudWatchLogs`-Senken-Deklarationen erfordern entweder ein `LogStream`- oder ein `StreamName`-Schlüssel-Wert-Paar. Der Wert dieser Schlüssel-Werte kann Variablenverweise enthalten, die vom Kinesis Agent für Windows automatisch aufgelöst werden. Für`CloudWatchLogs`, die`LogGroup`Ein -Schlüssel-Wert-Paar ist ebenfalls erforderlich und es kann Variablenverweise enthalten, die von Kinesis Agent für Windows automatisch aufgelöst werden. Die Variablen werden mithilfe der Vorlage {*`prefix`*`:`*`variablename`*} angegeben, wobei *`prefix`*`:` optional ist. Die folgenden Präfixe werden unterstützt:

- `env`— Der Variablenverweis wird auf den Wert der Umgebungsvariable desselben Namens aufgelöst.
- `ec2`— Der Variablenverweis wird auf die EC2-Instance-Metadaten desselben Namens aufgelöst.
- `ec2tag`— Der Variablenverweis wird auf den Wert des EC2-Instance-Tags desselben Namens aufgelöst. Für den Zugriff auf Instance-Tags ist die `ec2:Describe*`-Berechtigung erforderlich. Weitere Informationen finden Sie unter Erforderliche Berechtigungen für die EC2-Tag-Variablen-Erweiterung.

Wenn kein Präfix angegeben wird und eine Umgebungsvariable mit demselben Namen wie `variablename` vorhanden ist, wird der Variablenverweis auf den Wert der Umgebungsvariable aufgelöst. Wenn `variablename` `instance_id` oder `hostname` ist, wird der Variablenverweis andernfalls auf den Wert der EC2-Metadaten desselben Namens aufgelöst. Andernfalls wird der Variablenverweis nicht aufgelöst.

Es folgen Beispiele gültiger Schlüssel-Wert-Paare mit Variablenverweisen:

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

Die `CloudWatchLogs`-Senken-Deklarationen unterstützen eine Zeitstempelvariable in einem speziellen Format, mit dem der Zeitstempel des ursprünglichen Protokoll- oder Ereignisdatensatzes aus der Quelle den Namen des Protokollstreams ändern kann. Das Format ist `{timestamp:timeformat}`. Sehen Sie sich das folgende Beispiel an:

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

Wenn der Protokoll- oder Ereignisdatensatz am 5. Juni 2017 erstellt wurde, würde der Wert des `LogStream`-Schlüssel-Wert-Paares im vorherigen Beispiel auf `"LogStream_20170605"` aufgelöst werden.

Wenn der `CloudWatchLogs`-Senken-Typ entsprechend autorisiert ist, kann er automatisch neue Protokollstreams erstellen, wenn dies basierend auf den erstellten Namen erforderlich ist. Dies ist für andere Senken-Typen nicht möglich, da sie neben den Namen des Streams eine zusätzliche Konfiguration erfordern.

In der Text- und Objektausstattung finden besondere Variablensubstitutionen statt. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen.

Konfigurieren der Senken-Warteschlange

Die `KinesisStream`-, `KinesisFirehose`- und `CloudWatchLogs`-Senken-Deklarationen können optional das Einreihen von Datensätzen in eine Warteschlange aktivieren, bei denen beim Streamen

an den AWS-Service, der diesen Senken-Typen zugeordnet ist, aufgrund von vorübergehenden Verbindungsproblemen Fehler aufgetreten sind. Um Wiederholungsversuche für das Verschieben in eine Warteschlangen und für das automatische Streaming zu ermöglichen, wenn die Konnektivität wiederhergestellt ist, verwenden Sie in den Senken-Deklarationen die folgenden Schlüssel-Wert-Paare:

`QueueType`

Gibt die Art des für die Warteschlange zu verwendenden Mechanismus an. Der einzige unterstützte Wert ist `file`. Dies bedeutet, dass Datensätze in eine Warteschlange in Form einer Datei gestellt werden. Dieses Schlüssel-Wert-Paar ist erforderlich, um die Warteschlangenfunktion von Kinesis Agent für Windows zu aktivieren. Wenn es nicht angegeben wird, werden Daten standardmäßig nur in eine Wartschlange im Arbeitsspeicher gestellt und nicht gestreamt, wenn die Warteschlangen-Limits für den Arbeitsspeicher erreicht sind.

`QueuePath`

Gibt den Pfad zu dem Ordner an, in dem sich die Dateien der in die Warteschlange gestellten Datensätze befinden. Dieses Schlüssel-Wert-Paar ist optional. Der Standardwert ist `%PROGRAMDATA%\KinesisTap\Queue\SinkId`, wobei SinkId die Kennung ist, die Sie als Wert der `Id` für die Senken-Deklaration zugewiesen haben.

`QueueMaxBatches`

Beschränkt den gesamten Speicherplatz, den Kinesis Agent für Windows verbrauchen kann, wenn Datensätze zum Streamen in die Warteschlange gestellt werden. Die Menge des Speicherplatzes ist auf den Wert dieses Schlüssel-Wert-Paares, multipliziert mit der maximalen Anzahl von Bytes pro Stapel eingeschränkt. Die maximalen Bytes pro Stapel für die `KinesisStream`–, `KinesisFirehose`- und `CloudWatchLogs`-Senken-Typen sind 5 MB, 4 MB bzw. 1 MB. Wenn dieses Limit erreicht ist, werden keine Streaming-Fehler in die Warteschlange gestellt, sondern alle davon werden als nicht behebbare Fehler gemeldet. Dieses Schlüssel-Wert-Paar ist optional. Der Standardwert ist 10.000 Stapel.

Konfigurieren eines Proxys für Senken

Um einen Proxy für alle Kinesis Agent für Windows-Senken-Typen zu konfigurieren, die auf AWS -Services zugreifen, bearbeiten Sie die Kinesis Agent für Windows-Konfigurationsdatei unter gespeichert`%Program Files%\Amazon\KinesisTap \AWSKinesisTap.exe.config`. Detaillierte Anweisungen finden Sie im`proxy`-Abschnitt

unterReferenz für Konfigurationsdateireferenz für AWS SDK for .NETimAWS SDK for .NET Entwicklerhandbuch.

Konfigurieren von Auflösungsvariablen in mehr Senken-Attributen

Das folgende Beispiel zeigt eine -Senken-Konfiguration, die die`Region`Umgebungsvariable für den Wert des`Region`-Attribut-Schlüssel-Wert-Paar. Für`RoleARN`, gibt es den EC2-Tag-Schlüssel`MyRoleARN`Dieser Wert berechnet den Wert, der diesem Schlüssel zugeordnet ist.

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

Konfigurieren regionaler AWS STS Endpunkte bei Verwendung der RoleARN -Eigenschaft in AWS S-Senks

Diese Funktion gilt nur, wenn Sie KineSiStap in Amazon EC2 verwenden und die`RoleARN`-Eigenschaft von AWS S-Senks verwenden, um eine externe IAM-Rolle für die Authentifizierung bei den AWS-Ziel-Services anzunehmen.

Durch das Setzen von`UseSTSRegionalEndpoints`auf`true`können Sie angeben, dass ein Agent den regionalen Endpunkt verwendet (z. B.`https://sts.us-east-1.amazonaws.com`) anstelle des globalen Endpunkts (z. B.`https://sts.amazonaws.com`) enthalten. Die Verwendung eines regionalen STS-Endpunkts reduziert die Roundtrip-Latenz für den Vorgang und schränkt die Auswirkung von Fehlern im globalen Endpunktdienst ein.

Konfigurieren von VPC Endpoint für AWS -Senks

Sie können einen VPC Endpunkt in der Senke-Konfiguration für`CloudWatchLogs`,`CloudWatch`,`KinesisStreams`, und`KinesisFirehose`Spüle Typen. Ein VPC-Endpunkt ermöglicht Ihnen, eine private Verbindung zwischen Ihrer VPC und unterstützten AWS-Services und VPC-Endpunktservices mit AWS PrivateLink einzurichten, ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine AWS Direct Connect-Verbindung erforderlich sind. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit den Ressourcen in dem Service zu kommunizieren. Der Datenverkehr zwischen Ihrer VPC und dem

anderen Service verlässt das Amazon-Netzwerk nicht. Weitere Informationen finden Sie unterVPC-EndpunkteimBenutzerhandbuch für Amazon VPC Benutzerhandbuch.

Sie geben den VPC Endpunkt mithilfe der`ServiceURL`Gezeigt wird dies im folgenden Beispiel einer`CloudWatchLogs`Senke-Konfiguration. Legen Sie den Wert von`ServiceURL`auf den Wert, der auf der RegisterkarteDetails zu VPC EndpunktenVerwenden Sie die Amazon VPC Konsole.

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

Konfigurieren einer alternativen Methode für Proxy

Mit dieser Funktion können Sie einen Proxyserver in einer Sink-Konfiguration konfigurieren, indem Sie die im AWS SDK anstelle von .NET integrierte Proxyunterstützung verwenden. Bisher bestand die einzige Möglichkeit, den Agenten für die Verwendung eines Proxys zu konfigurieren, darin, ein natives Feature von .NET zu verwenden, das automatisch alle HTTP/S-Anforderungen über den in der Proxydatei definierten Proxy weitergeleitet hat.

Wenn Sie den Agenten derzeit mit einem Proxyserver verwenden, müssen Sie diese Methode nicht ändern.

Sie können das`ProxyHost`und`ProxyPort`Verwenden Sie dazu, einen alternativen Proxy zu konfigurieren, wie im folgenden Beispiel gezeigt.

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

Pipe-Deklarationen

Verwenden vonPipe-Deklarationen, um eine Quelle zu verbinden (siehe Quell-Deklarationen) zu einer Spüle (siehe Senken-Deklarationen) in Amazon Kinesis Agent für Microsoft Windows. Eine Pipe-Deklaration wird als JSON-Objekt ausgedrückt. Nachdem Kinesis Agent für Windows gestartet wurde, werden die Protokolle, Ereignisse oder Metriken für eine bestimmte Pipe aus der Quelle erfasst. Sie werden dann unter Verwendung der Senke, die dieser Pipe zugeordnet ist, an verschiedene AWS-Services gestreamt.

Es folgt ein Beispiel für die Pipe-Deklaration :

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

Themen

- Konfigurieren von Pipes
- Konfigurieren von Kinesis Agent für Windows-Metrik-Pipes

Konfigurieren von Pipes

Alle Pipe-Deklarationen können die folgenden Schlüssel-Wert-Paare enthalten:

`Id`

Gibt den Namen der Pipe an (erforderlich). Er muss innerhalb der Konfigurationsdatei eindeutig sein.

`Type`

Gibt an, welche Art von Transformation (sofern zutreffend) durch die Pipe angewendet wird, während Protokolldaten von der Quelle an die Senke übertragen werden. Der einzige unterstützte Wert ist `RegexFilterPipe`. Dieser Wert ermöglicht das Filtern regulärer Ausdrücke der zugrunde liegenden Textdarstellung des Protokolldatensatzes. Durch Filtern lassen sich Übertragungs- und Speicherkosten reduzieren, indem nur relevante Protokolldatensätze zur Downstream-Daten-Pipeline gesendet werden. Dieses Schlüssel-Wert-Paar ist optional. Der Standardwert ist, keine Transformation durchzuführen.

`FilterPattern`

Gibt den regulären Ausdruck für `RegexFilterPipe`-Pipelines an, der zum Filtern der aus der Quelle erfassten Protokolldatensätze verwendet wird, bevor sie an die Senke übertragen werden. Protokolldatensätze werden von Pipes vom Typ `RegexFilterPipe` übertragen, wenn der reguläre Ausdruck mit der zugrunde liegenden Textdarstellung des Datensatzes übereinstimmt. Strukturierte Protokolldatensätze, die erstellt werden, z. B. bei der Verwendung des `ExtractionPattern`-Schlüssel-Wert-Paares in einer `DirectorySource`-Deklaration, können weiterhin mit dem `RegexFilterPipe`-Mechanismus gefiltert werden. Dies liegt daran, da dieser Mechanismus vor der Analyse mit der ursprünglichen Textdarstellung funktioniert hat. Dieses Schlüssel-Wert-Paar ist optional, muss aber angegeben werden, wenn die Pipe als Typ `RegexFilterPipe` angibt.

Es folgt ein Beispiel für die Pipe-Deklaration `RegexFilterPipe`:

```
{
 "Id": "MyAppLog2ToFirehose",
 "Type": "RegexFilterPipe",
 "SourceRef": "MyAppLog2",
 "SinkRef": "MyFirehose",
 "FilterPattern": "^(10|11),.*",
 "IgnoreCase": false,
 "Negate": false
}
```

`SourceRef`

Gibt den Namen (den Wert des `Id`-Schlüssel-Wert-Paares) der Quell-Deklaration an, die die Quelle definiert, mit der Protokoll-, Ereignis- und Metrikdaten für die Pipe gesammelt werden (erforderlich).

`SinkRef`

Gibt den Namen (den Wert des `Id`-Schlüssel-Wert-Paares) der Senken-Deklaration an, die die Senke definiert, von der die Protokoll-, Ereignis- und Metrikdaten für die Pipe empfangen werden (erforderlich).

`IgnoreCase`

Optional. Akzeptiert Werte von`true`oder .`false`. Wenn auf`true`, wird die Regex Datensätze in der Groß- und Kleinschreibung nicht berücksichtigt.

`Negate`

Optional. Akzeptiert Werte von`true`oder .`false`. Wenn auf`true`, leitet die Pipe die Datensätze weiter, dieer/dieSie entsprechen dem regulären Ausdruck.

Ein Beispiel für eine vollständige Konfigurationsdatei, die den Pipe-Typ `RegexFilterPipe` verwendet, finden Sie unter Verwenden von Pipes.

Konfigurieren von Kinesis Agent für Windows-Metrik-Pipes

Es ist eine integrierte Metrikquelle mit dem Namen vorhanden`_KinesisTapMetricsSource`, die Metriken über Kinesis Agent für Windows erstellt. Wenn es eine`CloudWatch`Waschbecken Deklaration mit einem`Id`von`MyCloudWatchSink`Nachdem die folgende Pipeline-Beispieldeklaration den Kinesis Agent für Windows-erstellten Metriken an die jeweilige Senke:

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

Weitere Informationen zu den integrierten -Metrikquellen von Kinesis Agent für Windows finden Sie unterIntegrierte Quelle von Kinesis Agent für Windows -Metriken: Quelle.

Wenn die Konfigurationsdatei zudem auch Windows-Leistungsindikator-Metriken streamt, empfehlen wir, dass Sie eine separate Pipe und Senke verwenden, anstatt dieselbe Senke sowohl für -Metriken für Windows-Metriken als auch Windows-Leistungsindikator-Metriken zu nutzen.

Konfigurieren von automatischen Updates

Verwenden der`appsettings.json`Um die automatische Aktualisierung von Amazon Kinesis Agent für Microsoft Windows und der Konfigurationsdatei für Kinesis Agent für Windows zu ermöglichen. Um das Aktualisierungsverhalten zu steuern, geben Sie das `Plugins`-Schlüssel-Wert-Paar auf derselben Ebene in der Konfigurationsdatei wie `Sources`, `Sinks` und `Pipes` ein.

Das `Plugins`-Schlüssel-Wert-Paar gibt die zu verwendende zusätzliche allgemeine Funktionalität an, die nicht speziell in die Kategorien von Quellen, Senken und Pipes fallen. Zum Beispiel gibt es ein Plug-In für die Aktualisierung von Kinesis Agent für Windows, und ein Plug-In für die Aktualisierung

desappsettings.json-Konfigurationsdatei. Plug-Ins werden als JSON-Objekte dargestellt und haben immer ein Type-Schlüssel-Wert-Paar. Der Type legt fest, welche anderen Schlüssel-Wert-Paare für das Plug-In angegeben werden können. Die folgenden Arten von Plug-Ins werden unterstützt:

`PackageUpdate`

Gibt an, dass Kinesis Agent für Windows regelmäßig eine Paketversions-Konfigurationsdatei überprüfen soll. Wenn die Paketversionsdatei angibt, dass eine andere Version von Kinesis Agent für Windows installiert werden soll, dann lädt Kinesis Agent für Windows diese Version herunter und installiert sie. Zu den Schlüssel-Wert-Paaren des `PackageUpdate`-Plug-Ins gehören:

`Type`

Der Wert muss die Zeichenfolge `PackageUpdate` sein und ist erforderlich.

`Interval`

Gibt in Form einer Zeichenfolge an, wie oft in Minuten die Paketversionsdatei auf Änderungen untersucht werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wenn es nicht angegeben wird, lautet der Standardwert 60 Minuten. Wenn der Wert kleiner als 1 ist, findet keine Überprüfung auf Aktualisierungen statt.

`PackageVersion`

Gibt den Speicherort der Paketversions-JSON-Datei an. Die Datei kann sich in einer Dateifreigabe (`file://`), eine Website (`http://`) oder Amazon S3 (`s3://`) enthalten. Beispiel: Der Wert von`s3://mycompany/config/agent-package-version.json`gibt an, dass Kinesis Agent für Windows den Inhalt der`config/agent-package-version.json`Die Datei in der`mycompany`Amazon S3 Bucket. Es sollte Aktualisierungen basierend auf dem Inhalt der Datei durchführen.

> **Note**
>
> Der Wert von`PackageVersion`Bei Amazon S3-Wert wird für Groß- und Kleinschreibung beachtet.

Es folgt ein Beispiel für den Inhalt einer Paketversionsdatei:

```
{
```

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

Die`Version`Schlüssel-Wert-Paar gibt an, welche Version von Kinesis Agent für Windows installiert werden soll, wenn sie noch nicht installiert ist. Die Referenz für die Variable `{Version}` in der `PackageUrl` wird auf den Wert aufgelöst, den Sie für das `Version`-Schlüssel-Wert-Paar angeben. In diesem Beispiel wird die Variable auf die Zeichenfolge `1.0.0.106` aufgelöst. Diese Variablenauflösung wird bereitgestellt, sodass die spezifische gewünschte Version an einem einzigen Ort in der Paketversionsdatei gespeichert werden kann. Sie können mehrere Paketversionsdateien zur Steuerung des Tempos der Bereitstellung neuer Versionen von Kinesis Agent für Windows verwenden, um eine neue Version vor einer größeren Bereitstellung zu validieren. Um für eine Bereitstellung von Kinesis Agent für Windows ein Rollback durchzuführen, ändern Sie eine oder mehrere Paketversionsdateien, um eine frühere Version von Kinesis Agent für Windows anzugeben, die in Ihrer Umgebung bekanntlich funktioniert.

Durch Variablensubstitution kann der Wert des `PackageVersion`-Schlüssel-Wert-Paares beeinflusst werden, wodurch die automatische Auswahl verschiedener Paketversionsdateien erleichtert wird. Weitere Informationen zur Variablensubstitution finden Sie unter Konfigurieren von Senken-Variablensubstitutionen.

`AccessKey`

Gibt an, welcher Zugriffsschlüssel beim Authentifizieren des Zugriffs auf die Paketversionsdatei in Amazon S3 verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wir raten von der Verwendung dieses Schlüssel-Wert-Paares ab. Alternative Authentifizierungsmethoden, die empfohlen werden, finden Sie unter Konfigurieren der Authentifizierung.

`SecretKey`

Gibt an, welcher geheime Schlüssel beim Authentifizieren des Zugriffs auf die Paketversionsdatei in Amazon S3 Code verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wir raten von der Verwendung dieses Schlüssel-Wert-Paares ab. Alternative Authentifizierungsmethoden, die empfohlen werden, finden Sie unter Konfigurieren der Authentifizierung.

`Region`

Gibt an, welcher Regionsendpunkt beim Zugriff auf die Paketversionsdatei aus Amazon S3 Code verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional.

`ProfileName`

Gibt an, welches Sicherheitsprofil beim Authentifizieren des Zugriffs auf die Paketversionsdatei in Amazon S3 verwendet werden soll. Weitere Informationen finden Sie unter Konfigurieren der Authentifizierung. Dieses Schlüssel-Wert-Paar ist optional.

`RoleARN`

Gibt an, welche Rolle übernommen werden soll, wenn der Zugriff auf die Paketversionsdatei in Amazon S3 in einem kontoübergreifenden Szenario authentifiziert wird. Weitere Informationen finden Sie unter Konfigurieren der Authentifizierung. Dieses Schlüssel-Wert-Paar ist optional.

Wenn kein `PackageUpdate`-Plug-In angegeben wird, werden keine Paketversionsdateien überprüft, um zu bestimmen, ob eine Aktualisierung erforderlich ist.

`ConfigUpdate`

Gibt an, dass Kinesis Agent für Windows regelmäßig nach einem aktualisierten`appsettings.json`Die Konfigurationsdatei wird in einer Dateifreigabe, auf einer Website oder in Amazon S3 Dateien gespeichert. Wenn eine aktualisierte Konfigurationsdatei vorhanden ist, wird sie von Kinesis Agent für Windows heruntergeladen und installiert.`ConfigUpdate`Zu den Schlüssel-Wert-Paaren gehören Folgendes:

`Type`

Der Wert muss die Zeichenfolge `ConfigUpdate` sein und ist erforderlich.

`Interval`

Gibt in Form einer Zeichenfolge an, wie oft in Minuten auf eine neue Konfigurationsdatei geprüft werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wenn es nicht angegeben wird, werden standardmäßig 5 Minuten eingestellt. Wenn der Wert kleiner als 1 ist, wird nach Aktualisierungen an der Konfigurationsdatei gesucht.

`Source`

Gibt an, wo nach einer aktualisierten Konfigurationsdatei gesucht werden soll. Die Datei kann sich in einer Dateifreigabe (`file://`), eine Website (`http://`) oder Amazon S3 (`s3://`) enthalten. Beispiel: Der Wert von`s3://mycompany/config/appsettings.json`gibt an,

dass Kinesis Agent für Windows nach Updates für die`config/appsettings.json`Die Datei in der`mycompany`Amazon S3 Bucket.

> **Note**
>
> Der Wert von`Source`Bei Amazon S3 Wert-Paar wird für Groß- und Kleinschreibung beachtet.

Durch Variablensubstitution kann der Wert des `Source`-Schlüssel-Wert-Paares beeinflusst werden, wodurch die automatische Auswahl verschiedener Konfigurationsdateien erleichtert wird. Weitere Informationen zur Variablensubstitution finden Sie unter Konfigurieren von Senken-Variablensubstitutionen.

`Destination`

Gibt den Speicherort für die Konfigurationsdatei auf dem lokalen Computer an. Hierbei kann es sich um einen relativen Pfad, einen absoluten Pfad oder einen Pfad mit Umgebungsvariablenverweisen, wie z. B. `%PROGRAMDATA%`, handeln. Wenn der Pfad relativ angegeben ist, ist er relativ zu dem Speicherort, an dem Kinesis Agent für Windows installiert ist. Der Wert sollte normalerweise `.\appsettings.json` lauten. Dieser Schlüssel-Wert-Paar ist erforderlich.

`AccessKey`

Gibt an, welcher Zugriffsschlüssel beim Authentifizieren des Zugriffs auf die Konfigurationsdatei in Amazon S3 Code verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wir raten von der Verwendung dieses Schlüssel-Wert-Paares ab. Alternative Authentifizierungsmethoden, die empfohlen werden, finden Sie unter Konfigurieren der Authentifizierung.

`SecretKey`

Gibt an, welcher geheime Schlüssel beim Authentifizieren des Zugriffs auf die Konfigurationsdatei in Amazon S3 Code verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional. Wir raten von der Verwendung dieses Schlüssel-Wert-Paares ab. Alternative Authentifizierungsmethoden, die empfohlen werden, finden Sie unter Konfigurieren der Authentifizierung.

`Region`

Gibt an, welcher Regionsendpunkt beim Zugriff auf die Konfigurationsdatei aus Amazon S3 Code verwendet werden soll. Dieses Schlüssel-Wert-Paar ist optional.

`ProfileName`

Gibt an, welches Sicherheitsprofil beim Authentifizieren des Zugriffs auf die Konfigurationsdatei in Amazon S3 Code verwendet werden soll. Weitere Informationen finden Sie unter Konfigurieren der Authentifizierung. Dieses Schlüssel-Wert-Paar ist optional.

`RoleARN`

Gibt an, welche Rolle übernommen werden soll, wenn der Zugriff auf die Konfigurationsdatei in Amazon S3 Code in einem kontoübergreifenden Szenario authentifiziert wird. Weitere Informationen finden Sie unter Konfigurieren der Authentifizierung. Dieses Schlüssel-Wert-Paar ist optional.

Wenn kein `ConfigUpdate`-Plug-In angegeben wird, werden keine Konfigurationsdateien überprüft, um zu bestimmen, ob eine Konfigurationsdatei aktualisiert werden muss.

In der folgenden Beispiel-Konfigurationsdatei `appsettings.json` wird die Verwendung der Plug-Ins `PackageUpdate` und `ConfigUpdate` veranschaulicht. In diesem Beispiel befindet sich eine Paketversionsdatei in der`mycompany`Amazon S3 Bucket mit der Bezeichnung`config/agent-package-version.json`. Diese Datei wird ca. alle 2 Stunden auf Änderungen überprüft. Wenn in der Paketversionsdatei eine andere Version von Kinesis Agent für Windows angegeben wird, wird die angegebene Agent-Version über den angegebenen Speicherort in der Paketversionsdatei installiert.

Darüber hinaus gibt es eine`appsettings.json`-Konfigurationsdatei, die in der`mycompany`Amazon S3 Bucket mit der Bezeichnung`config/appsettings.json`. Diese Datei wird ungefähr alle 30 Minuten mit der aktuellen Konfigurationsdatei verglichen. Wenn sie voneinander abweichen, wird die aktualisierte Konfigurationsdatei von Amazon S3 heruntergeladen und an dem typischen lokalen Speicherort für den`appsettings.json`-Konfigurationsdatei.

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

Konfigurationsbeispiele für Kinesis Agent für Windows

Die`appsettings.json`Die -Konfigurationsdatei ist ein JSON-Dokument, das steuert, wie Amazon Kinesis Agent für Microsoft Windows Protokolle, Ereignisse und Metriken sammelt. Sie steuert ebenfalls, wie Kinesis Agent für Windows diese Daten transformiert und an verschiedene AWS -Services streamt. Weitere Informationen über die Quell-, Senken- und Pipe-Deklarationen in der Konfigurationsdatei finden Sie unter Quell-Deklarationen, Senken-Deklarationen, und Pipe-Deklarationen.

Die folgenden Abschnitte enthalten Beispiele für Konfigurationsdateien für verschiedene Arten von Szenarien.

Themen

Streamen aus verschiedenen Quellen an Kinesis Data Streams

Das folgende Beispiel`appsettings.json`Beispiel-Konfigurationsdateien veranschaulichen das Streamen von Protokollen und Ereignissen aus verschiedenen Quellen an Kinesis Data Streams und von Windows-Leistungsindikatoren an Amazon CloudWatch Metriken.

`DirectorySource`-, **`SysLog`**-Datensatz-Parser

Die folgende Datei streamt Protokolldatensätze im Syslog-Format aus allen Dateien mit einem`.log`-Dateierweiterung in der`C:\LogSource\`-Verzeichnis an den`SyslogKinesisDataStream`Streamen aus Kinesis Data Streams in der Region us-east-1. Ein Lesezeichen wird eingerichtet, um sicherzustellen, dass sogar dann alle Daten aus den Protokolldateien gesendet werden, wenn der Agent heruntergefahren und zu einem späteren Zeitpunkt neu gestartet wird. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `SyslogKinesisDataStream`-Stream lesen und verarbeiten.

```
{
  "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**-Datensatz-Parser

Die folgende Datei streamt JSON-formatierte Protokolldatensätze aus allen Dateien mit einem `.log`-Dateierweiterung in der`C:\LogSource\`-Verzeichnis an den`JsonKinesisDataStream`Streamen aus Kinesis Data Streams in der Region us-east-1. Vor dem Streamen werden Schlüssel-Wert-Paare für die Schlüssel `ComputerName` und `DT` zu jedem JSON-Objekt mit Werten für den Computernamen und das Datum und die Uhrzeit der Datensatzverarbeitung hinzugefügt. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `JsonKinesisDataStream`-Stream lesen und verarbeiten.

```
{
  "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

Die folgende Datei streamt Protokolldatensätze, die von Microsoft Exchange generiert und in Dateien mit dem `.log`-Erweiterung in der`C:\temp\ExchangeLog\`-Verzeichnis an den`ExchangeKinesisDataStream`Kinesis -Daten-Stream in der Region us-east-1 im JSON-Format. Obwohl die Exchange-Protokolle nicht im JSON-Format vorliegen, kann Kinesis Agent für Windows die Protokolle analysieren und in JSON-Format in JSON-Format transformieren. Vor dem Streamen werden Schlüssel-Wert-Paare für die Schlüssel `ComputerName` und `DT` zu jedem JSON-Objekt mit Werten für den Computernamen und das Datum und die Uhrzeit der Datensatzverarbeitung hinzugefügt. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `ExchangeKinesisDataStream`-Stream lesen und verarbeiten.

```
{
  "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

Die folgende Datei streamt Internetinformationsdienste (IIS) für Windows-Protokolldatensätze, die am Standardspeicherort für diese Dateien gespeichert werden, an den`IISKinesisDataStream`Streamen aus Kinesis Data Streams in der Region us-east-1. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `IISKinesisDataStream`-Stream lesen und verarbeiten. IIS ist ein Webserver für Windows.

```
{
  "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` mit Abfrage

Die folgende Dateistreams protokollieren Ereignisse aus dem Windows-Systemereignisprotokoll, die eine Ebene von`Critical`oder .`Error`(kleiner als oder gleich 2) an den`SystemKinesisDataStream`Kinesis -Daten-Stream in der Region us-east-1 im JSON-Format. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `SystemKinesisDataStream`-Stream lesen und verarbeiten.

```
{
  "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`

Die folgende Datei streamt Microsoft Common Language Runtime (CLR) -Ausnahme- und -Sicherheitsereignisse an den`ClrKinesisDataStream`Kinesis -Daten-Stream in der Region us-east-1 im JSON-Format. Eine benutzerdefinierte Anwendung kann die Datensätze aus dem `ClrKinesisDataStream`-Stream lesen und verarbeiten.

```
{
  "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`

Die folgende Datei streamt Leistungsindikatoren für alle geöffneten Dateien, alle Anmeldeversuche seit dem Neustart, die Anzahl der Datenträger-Lesevorgänge pro Sekunde und den Prozentsatz an freiem Speicherplatz an -CloudWatch Metriken in der Region us-east-1. Sie können diese Metriken in CloudWatch grafisch darstellen, Dashboards aus den Diagrammen erstellen und Alarme festlegen, die Benachrichtigungen senden, wenn Schwellenwerte überschritten werden.

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

Streamen aus dem Windows-Anwendungsereignisprotokoll an Senken

Das folgende Beispiel`appsettings.json`Beispiel-Konfigurationsdateien veranschaulichen das Streamen von Windows-Anwendungsereignisprotokollen an verschiedene Senken in Amazon Kinesis Agent für Microsoft Windows. Beispiele für die Verwendung der Senken-Typen `KinesisStream` und `CloudWatch` finden Sie unter Streamen aus verschiedenen Quellen an Kinesis Data Streams.

`KinesisFirehose`

Die folgenden Dateiströme`Critical`oder .`Error`Windows-Anwendung protokollieren Ereignisse in die`WindowsLogFirehoseDeliveryStream`Kinesis Data Firehose Delivery-Stream in der Region us-east-1. Wenn die Verbindung mit Kinesis Data Firehose unterbrochen wird, werden Ereignisse zuerst in die Warteschlange im Arbeitsspeicher gestellt. Danach werden sie, sofern erforderlich, in die Warteschlange in einer Datei auf der Festplatte gestellt, bis die Verbindung wiederhergestellt wurde. Anschließend werden Ereignisse aus der Warteschlange genommen und gefolgt von neuen Ereignissen gesendet.

Sie können Kinesis Data Firehose so konfigurieren, dass die gestreamten Daten an verschiedenen Arten von Speicher- und Analyse-Services basierend auf den Daten-Pipeline-Anforderungen gespeichert werden.

```
{
  "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`

Die folgenden Dateiströme`Critical`oder .`Error`Windows-Anwendungsprotokollereignisse in CloudWatch Logs an -Protokoll-Streams im`MyServiceApplicationLog-Group`-Protokollgruppe. Der Name jedes einzelnen Streams beginnt mit `Stream-`. Er endet mit dem vierstelligen Jahr, zweistelligen Monat und zweistelligen Tag, an dem der Stream erstellt wurde, wobei alle Angaben miteinander verkettet werden (`Stream-20180501` ist z. B. der am 1. Mai 2018 erstellte Stream).

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

Verwenden von Pipes

Die folgende `appsettings.json`-Beispiel-Konfigurationsdatei veranschaulicht die Verwendung Pipe-bezogener Funktionen.

Dieses Beispiel streamt Protokolleinträge aus dem`c:\LogSource\`auf die`ApplicationLogFirehoseDeliveryStream`Kinesis Data Firehose Delivery-Stream. Es enthält nur die Zeilen, die dem regulären Ausdruck entsprechen, der durch das `FilterPattern`-Schlüssel-Wert-Paar angegeben wird. Genauer gesagt werden nur Zeilen in der Protokolldatei, die mit`10`oder .`11`Genauer gesagt werden an Kinesis Data Firehose gestreamt.

```
{
  "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),.*"
    }
  ]
}
```

Verwenden mehrerer Datenquellen und Pipes

Die folgende `appsettings.json`-Beispiel-Konfigurationsdatei veranschaulicht die Verwendung mehrerer Quellen und Pipes.

Dieses Beispiel streamt die Windows-Ereignisprotokolle für Anwendung, Sicherheit und System an den`EventLogStream`Kinesis Data Firehose Delivery-Stream mit drei Quellen, drei Pipes und einer einzelnen Senke.

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

Konfigurieren von Telemetrie

Um eine bessere Unterstützung zu ermöglichen, sammelt standardmäßig Amazon Kinesis Agent für Microsoft Windows Statistiken über den Betrieb des Agenten und sendet sie an AWS. Diese Informationen enthalten personenbezogene Daten. Sie enthalten nicht alle Daten, die Sie sammeln oder an AWS-Services streamen. Wir sammeln alle 60 Minuten ungefähr 1 bis 2 KB dieser Metrikdaten.

Sie können die Erfassung und Übertragung dieser Statistiken deaktivieren. Dazu fügen Sie das folgenden Schlüssel-Wert-Paar auf derselben Ebene wie Quellen, Senken und Pipes zur Konfigurationsdatei `appsettings.json` hinzu:

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

Beispiel: Die folgende Konfigurationsdatei konfiguriert eine Quelle, Senke und Pipe und deaktiviert außerdem Telemetriken:

```
{
  "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"
    }
}
```

Wir sammeln die folgenden Metriken, wenn Telemetrie aktiviert ist:

`ClientId`

Die automatisch zugewiesene ID, wenn die Software installiert ist.

`ClientTimestamp`

Datum und Uhrzeit des Zeitpunkts, als die Telemetrie erfasst wurde.

`OSDescription`

Eine Beschreibung des Betriebssystems.

`DotnetFramework`

Die aktuelle dotnet-Framework-Version.

`MemoryUsage`

Der von Kinesis Agent für Windows verbrauchte Speicherverbrauch (in MB).

`CPUUsage`

Der Prozentsatz Kinesis CPU-Auslastung im Dezimalformat. Beispiel: 0,01 bedeutet 1 %.

`InstanceId`

Die Amazon EC2 Instance-ID, wenn Kinesis Agent für Windows auf einer Amazon EC2 Instance ausgeführt wird.

`InstanceType (string)`

Der Amazon EC2 Instance-Typ, wenn Kinesis Agent für Windows auf einer Amazon EC2 Instance ausgeführt wird.

Außerdem sammeln wir die unter Liste der Kinesis Agent für Windows-Metriken aufgelisteten Metriken.

Tutorial: Streamen von JSON-Protokolldateien mit Kinesis Agent für Windows zu Amazon S3

In diesem Tutorial finden Sie ausführliche Schritte zur Einrichtung einer Daten-Pipeline mit Amazon Kinesis Agent für Microsoft Windows (Kinesis Agent für Windows).

In diesem Tutorial führen Sie die folgenden Schritte durch:

- Verwenden von Kinesis Agent for Windows zum Streamen von Protokolldateien im JSON-Format anAmazon Simple Storage Service (Amazon S3)überAmazon Kinesis Data Firehose. Weitere Informationen zu Kinesis Agent für Windows finden Sie unterWas ist der Amazon Kinesis Agent für Microsoft Windows?.
- Optimieren der Protokolldaten vor dem Streamen durch Objektausstattung. Weitere Informationen finden Sie unter Konfigurieren von Senken-Ausstattungen.
- Verwenden vonAmazon AthenaSuchen Sie nach bestimmten Arten von Protokolldatensätzen.

Prerequisites

Wenn Sie noch kein AWS Konto haben, befolgen Sie die Anweisungen unterEinrichten eines AWS-Kontos, um einen zu bekommen.

Themen

Schritt 1: Konfigurieren von AWS -Services

Führen Sie diese Schritt durch, um Ihre Umgebung auf das Streamen von Protokolldaten an Amazon Simple Storage Service (Amazon S3) mittels Amazon Kinesis Agent für Microsoft Windows vorzubereiten. Weitere Informationen und Voraussetzungen finden Sie unter Tutorial: Streamen von JSON-Protokolldateien an Amazon S3.

Verwenden Sie die AWS Management Console, um AWS Identity and Access Management (IAM), Amazon S3, Kinesis Data Firehose und Amazon Elastic Compute Cloud (Amazon EC2) zu konfigurieren, um die Streaming-Protokolldaten von einer EC2-Instance zu Amazon S3 vorzubereiten.

Themen

Konfigurieren von IAM-Richtlinien und -Rollen

Erstellen Sie die folgende Richtlinie, die Kinesis Agent für Windows zum Streamen von Datensätzen an einen bestimmten Kinesis Data Firehose -Bereitstellungs-Stream autorisiert:

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

Ersetzen*region*Der Name der AWS Region, in der der -Bereitstellungs-Stream von Kinesis Data Firehose erstellt wird (`us-east-1`, zum Beispiel). Ersetzen Sie *`account-id`* durch die 12-stelige Konto-ID für das AWS-Konto, in dem der Bereitstellungs-Stream erstellt wird.

Wählen Sie in der Navigationsleiste-Support, und dannSupport-Center. Ihre aktuell angemeldete 12-stellige Kontonummer (ID) wird imSupport-Center-Navigationsbereich.

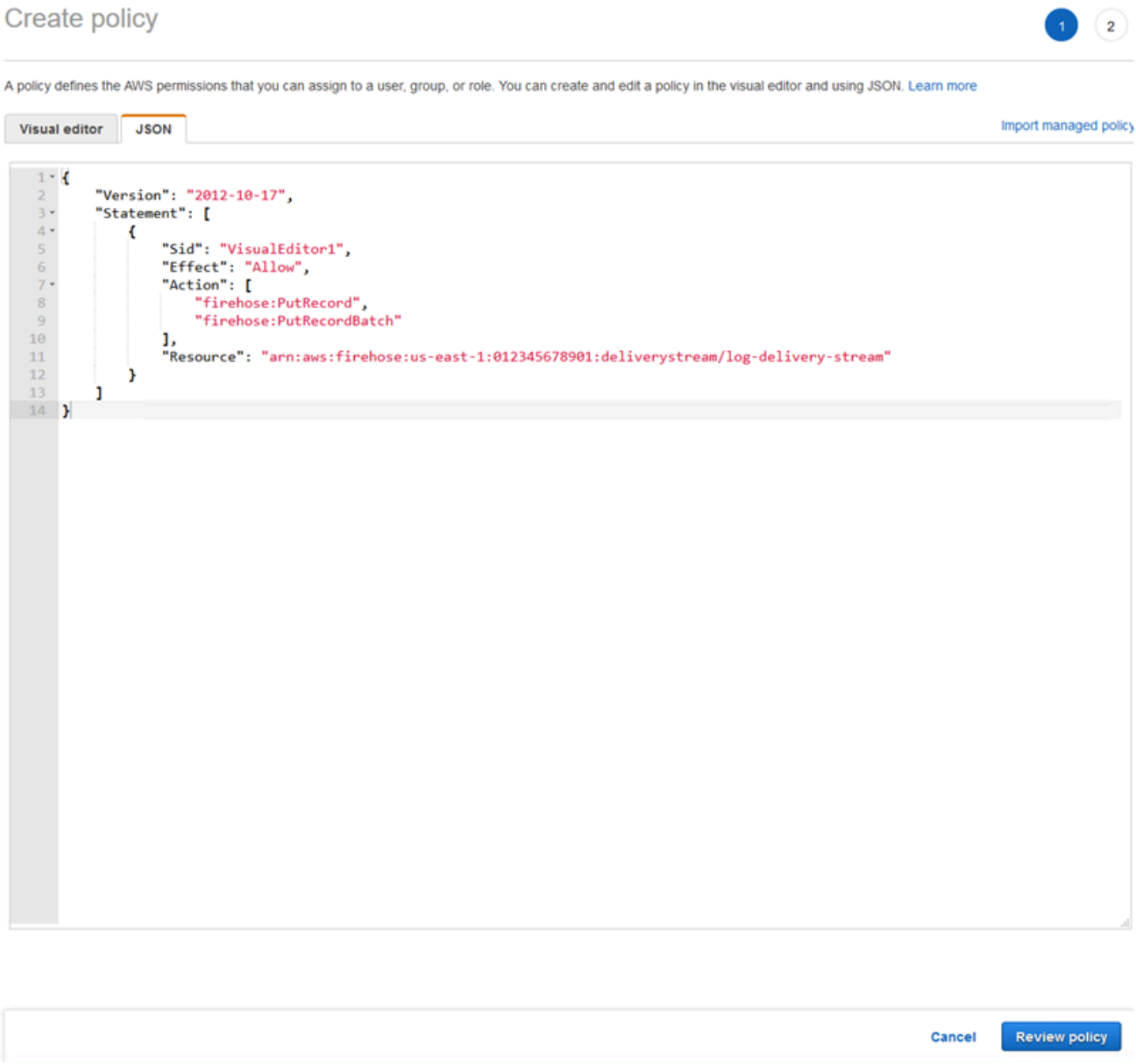
Erstellen Sie die Richtlinie mit dem folgenden Verfahren. Speichern Sie die Richtlinie unter dem Namen `log-delivery-stream-access-policy`.

So erstellen Sie eine Richtlinie mit dem JSON-Richtlinieneditor

1. Melden Sie sich bei der AWS-Managementkonsole an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien) aus.

   Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.
3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie die Registerkarte JSON.
5. Geben Sie ein JSON-Richtliniendokument ein. Weitere Informationen zur IAM-Richtliniensprache finden Sie unterIAM-JSON-RichtlinienreferenzimIAM-Benutzerhandbuch.
6. Wählen Sie, wenn Sie fertig sind, Review policy (Richtlinie überprüfen) aus. Die Richtlinienvalidierung meldet mögliche Syntaxfehler.

   > **Note**
   >
   > Sie können jederzeit zwischen den Registerkarten Visual editor (Visueller Editor) und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oderReview policyimVisual editor (Visueller Editor)Ihre Richtlinie möglicherweise neu, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unterUmstrukturierung einer RichtlinieimIAM-Benutzerhandbuch.
7. Geben Sie auf der Seite Review policy (Richtlinie überprüfen) unter Name einen Namen und unter Description (Beschreibung) (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie unter Summary die Richtlinienzusammenfassung, um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden. Wählen Sie dann Create policy aus, um Ihre Eingaben zu speichern.



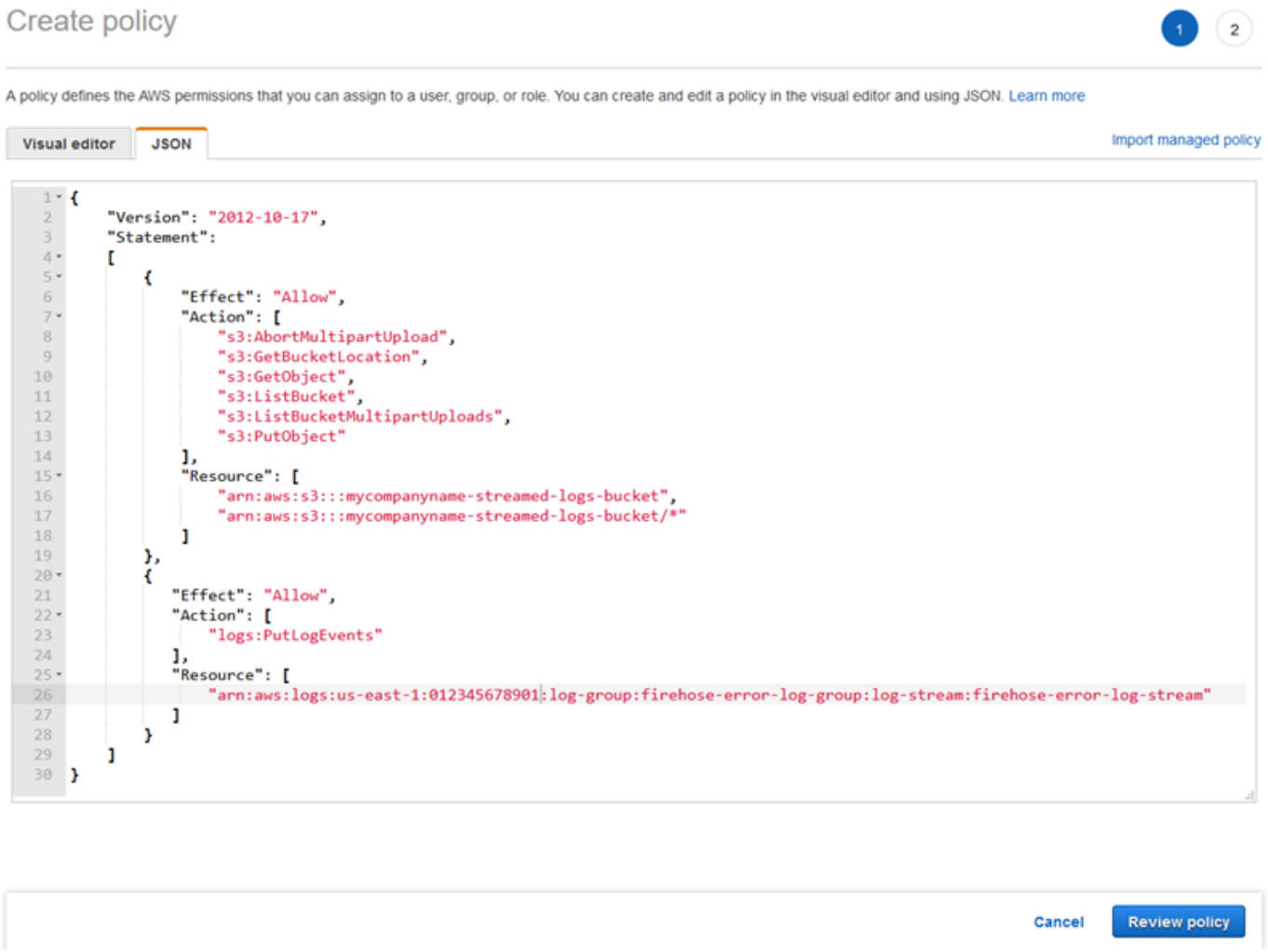
So erstellen Sie die Rolle, die Kinesis Data Firehose Zugriff auf einen S3-Bucket gewährt

1. Erstellen Sie mit dem vorherigen Verfahren eine Richtlinie mit dem Namen `firehose-s3-access-policy`, die mittels des folgenden JSON-Dokuments definiert wird:

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

Ersetzen Sie *bucket-name* durch einen eindeutigen Namen für den Bucket, in dem die Protokolle gespeichert werden. Ersetzen*region*Erstellen Sie durch die AWS Region, in der die Protokollgruppe und der Protokoll-Stream von CloudWatch Logs erstellt werden. Diese dienen zur Protokollierung aller Fehler, die beim Streamen von Daten an Amazon S3 über Kinesis Data Firehose aufreten. Ersetzen Sie *account-id* durch die 12-stellige Konto-ID für das Konto, in dem die Protokollgruppe und der Protokoll-Stream erstellt werden.



2. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles und wählen Sie dann Create role aus.
3. Wählen Sie das SymbolAWS-ServiceKlicken Sie auf und wählen Sie dann die OptionKinesis-Service.
4. Klicken Sie aufKinesis Data FirehoseKlicken Sie auf den Anwendungsfall und wählen Sie dannWeiter: Berechtigungen
5. Geben Sie in das Suchfeld**`firehose-s3-access-policy`**Wählen Sie diese Richtlinie aus und wählen Sie dann aus.Weiter: Prüfen.
6. Geben Sie im Feld Role Name (Rollenname) **`firehose-s3-access-role`** ein.
7. Wählen Sie Create role aus.

So erstellen Sie die Rolle, die dem Instance-Profil für die EC2-Instance zugeordnet werden soll, auf der Kinesis Agent für Windows ausgeführt wird

1. Klicken Sie im Navigationsbereich der IAM-Konsole auf Roles und wählen Sie dann Create role aus.
2. Wählen Sie das SymbolAWS-ServiceKlicken Sie auf, und wählen Sie dann ausEC2.
3. Klicken Sie aufWeiter: Berechtigungen
4. Geben Sie in das Suchfeld **`log-delivery-stream-access-policy`** ein.
5. Wählen Sie die Richtlinie aus und wählen Sie dann aus.Weiter: Prüfen.
6. Geben Sie im Feld Role Name (Rollenname) **`kinesis-agent-instance-role`** ein.
7. Wählen Sie Create role aus.

Erstellen des Amazon S3 Buckets

Erstellen Sie den S3-Bucket, in dem Kinesis Data Firehose die Protokolle streamt.

So erstellen Sie den S3-Bucket für die Speicherung von Protokollen

1. Öffnen Sie die Amazon S3-Konsole unter https://console.aws.amazon.com/s3/.
2. Wählen Sie Create Bucket (Bucket erstellen) aus.
3. Geben Sie im Feld Bucket name (Bucket-Name) den eindeutigen S3-Bucket-Namen ein, den Sie unter Konfigurieren von IAM-Richtlinien und -Rollen ausgewählt haben.
4. Wählen Sie die Region aus, in der der Bucket erstellt werden sollte. Dies ist in der Regel die gleiche Region, in der Sie den -Bereitstellungs-Stream von Kinesis Data Firehose und die Amazon EC2 Instance erstellen möchten.
5. Wählen Sie Create (Erstellen) aus.

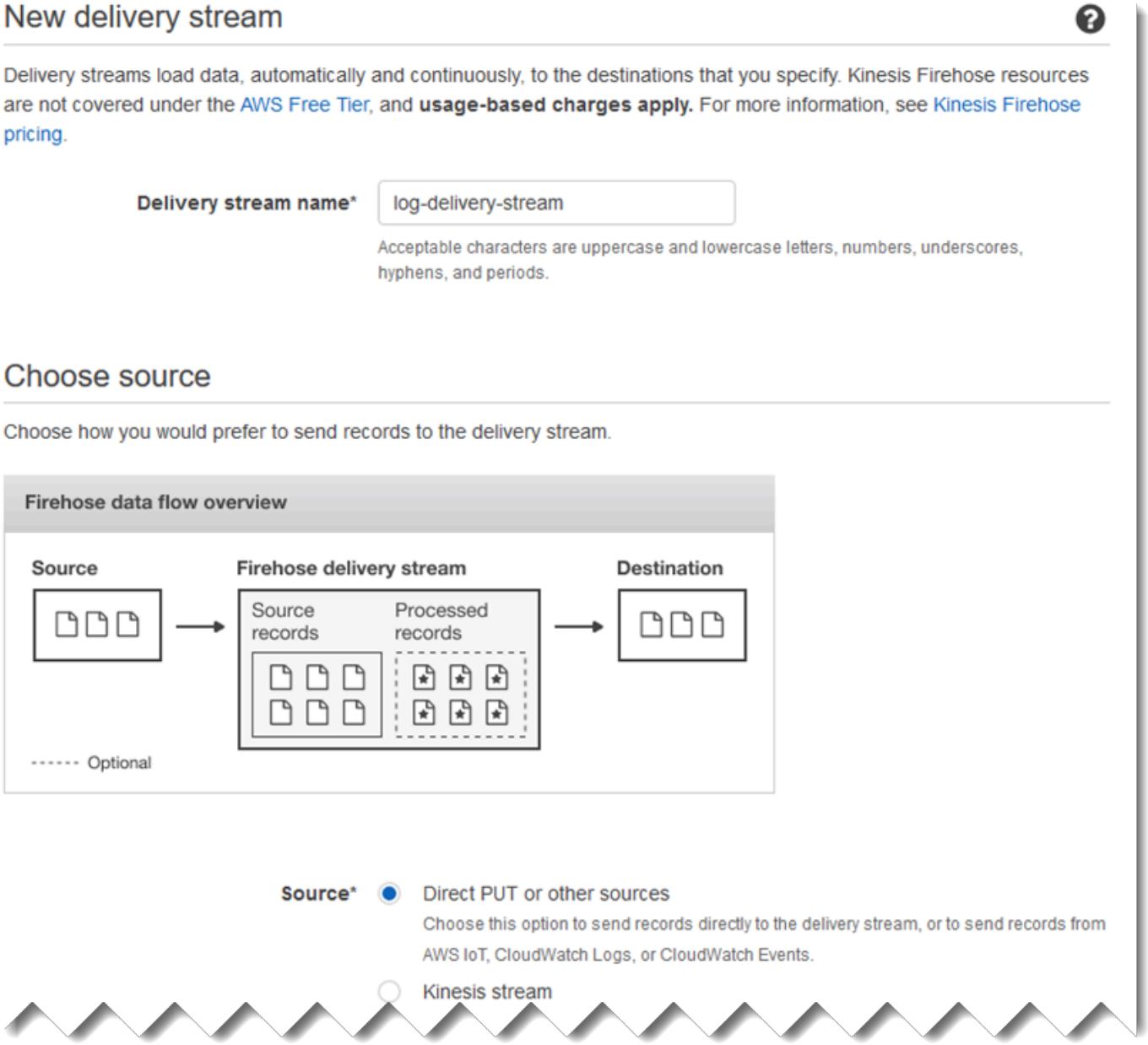
Erstellen des Kinesis Data Firehose Delivery-Stream

Erstellen Sie den Kinesis Data Firehose Delivery-Stream, der gestreamte Datensätze in Amazon S3 speichert.

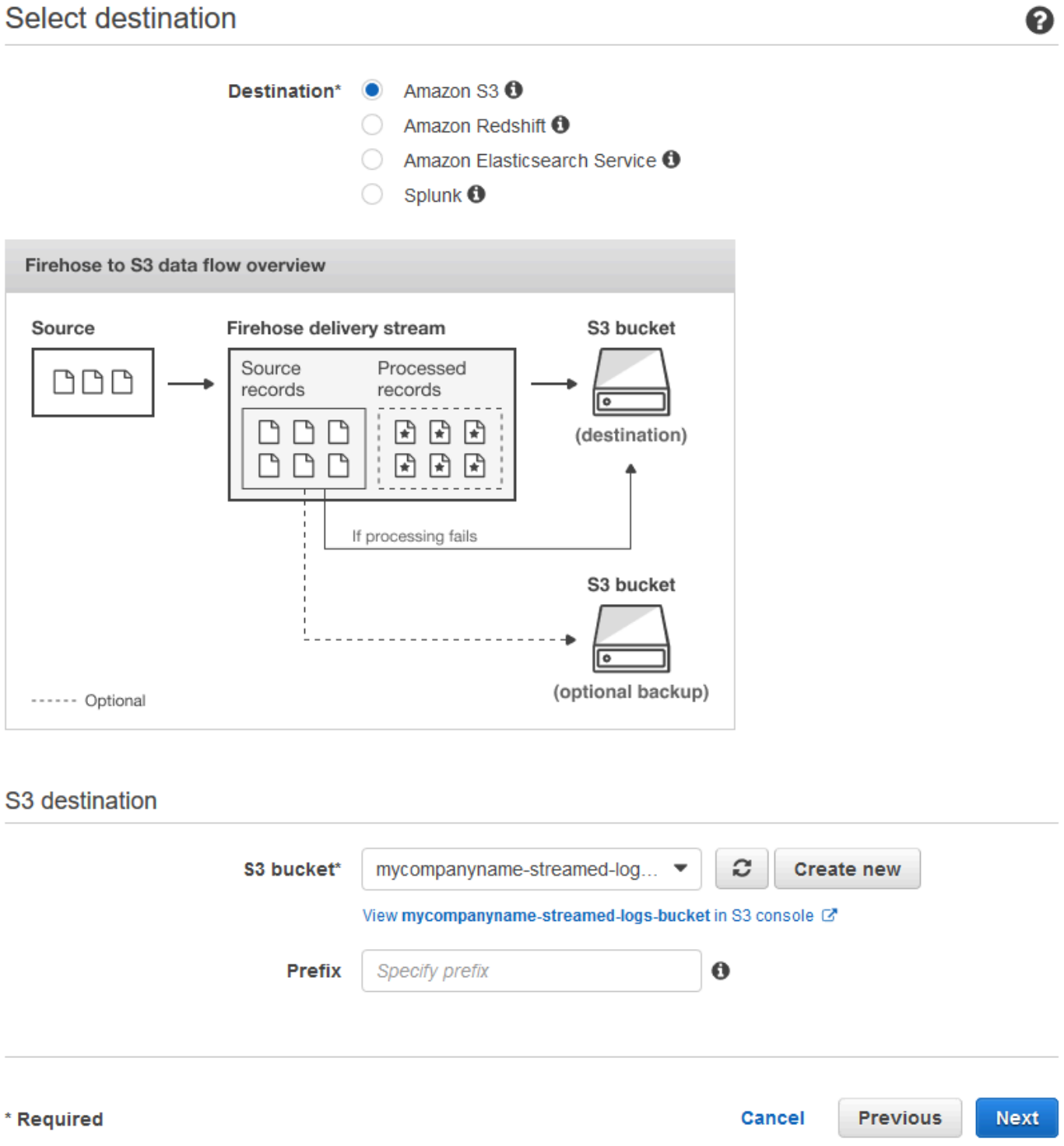
So erstellen Sie den Kinesis Data Firehose Delivery-Stream

1. Öffnen Sie die Kinesis Data Firehose Konsole unterhttps://console.aws.amazon.com/firehose/.
2. Wählen Sie Create Delivery Stream aus.

3. Geben Sie im Feld Delivery stream name (Name des Bereitstellungs-Streams) **`log-delivery-stream`** ein.
4. Wählen Sie für Source (Quelle) die Option Direct PUT or other sources (Direct PUT oder andere Quellen) aus.



5. Wählen Sie Next.
6. Wählen Sie erneut Next (Weiter).
7. Wählen Sie für das ZielAmazon S3.
8. Wählen Sie für S3 bucket (S3-Bucket) den Namen des Buckets aus, den Sie unter Erstellen des Amazon S3 Buckets erstellt haben.



9. Wählen Sie Next.
10. Geben Sie in das Feld Buffer interval (Pufferintervall) **60** ein.
11. Wählen Sie für IAM role (IAM-Rolle) die Option Create new or choose (Neu erstellen oder auswählen) aus.

12. Wählen Sie für IAM Role (IAM-Rolle) `firehose-s3-access-role` aus.
13. Wählen Sie Allow.

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. Wählen Sie Next.
15. Wählen Sie Create Delivery Stream (Bereitstellungs-Stream erstellen) aus.

Erstellen Sie die Amazon EC2 Instance, um Kinesis Agent für Windows auszuführen

Erstellen Sie die EC2-Instance, die Kinesis Agent for Windows zum Streamen von Protokolldatensätzen über Kinesis Data Firehose verwendet.

So erstellen Sie die EC2-Instance

1. Öffnen Sie die Amazon EC2-Konsole unter https://console.aws.amazon.com/ec2/.
2. Befolgen Sie die Anweisungen unter Erste Schritte mit Amazon EC2 Windows-Instances unter Verwendung der folgenden zusätzlichen Schritte:
   - Wählen Sie unter IAM role (IAM-Rolle) für die Instance `kinesis-agent-instance-role` aus.
   - Wenn Sie nicht bereits eine über das öffentliche Internet verbundene Virtual Private Cloud (VPC) besitzen, befolgen Sie die Anweisungen unterEinrichten von Amazon EC2imAmazon EC2 Benutzerhandbuch für Windows-Instances.
   - Erstellen oder verwenden Sie eine Sicherheitsgruppe, die den Zugriff auf die Instance so einschränkt, dass er nur über Ihren Computer oder die Computer Ihrer Organisation möglich ist. Weitere Informationen finden Sie unterEinrichten von Amazon EC2imAmazon EC2 Benutzerhandbuch für Windows-Instances.
   - Wenn Sie ein vorhandenes Schlüsselpaar angeben, stellen Sie sicher, dass Sie Zugriff auf den privaten Schlüssel für das Schlüsselpaar haben. Oder erstellen Sie ein neues Schlüsselpaar und speichern Sie den privaten Schlüssel an einer sicheren Stelle.
   - Bevor Sie fortfahren, warten Sie, bis die Instance ausgeführt wird und zwei von zwei Zustandsprüfungen bestanden hat.
   - Die Instance erfordert eine öffentliche IP-Adresse. Wenn keine zugewiesen wurde, befolgen Sie die Anweisungen unterElastic IP-AdressenimAmazon EC2 Benutzerhandbuch für Windows-Instances.

Nächste Schritte

Schritt 2: Installieren, Konfigurieren und Ausführen von Kinesis Agent für Windows

Schritt 2: Installieren, Konfigurieren und Ausführen von Kinesis Agent für Windows

In diesem Schritt verwenden Sie die AWS Management Console, um eine Remote-Verbindung mit der Instance herzustellen, die Sie unterErstellen Sie die Amazon EC2 Instance, um Kinesis Agent für Windows auszuführen. Dann installieren Sie Amazon Kinesis Agent für Microsoft Windows auf der Instance, erstellen die Konfigurationsdatei für Kinesis Agent für Windows und stellen sie bereit. Starten Sie dann dieAWSkineSiStap-Service.

1. Stellen Sie mittels RDP (Remote Desktop Protocol) eine Remote-Verbindung mit der Instance her. Befolgen Sie hierzu die Anweisungen unterSchritt 2: Herstellen einer Verbindung mit Ihrer InstanceimAmazon EC2 Benutzerhandbuch für Windows-Instances.
2. Deaktivieren Sie auf der Instance mit Microsoft Windows Server Manager die verstärkte Sicherheitskonfiguration von Microsoft Internet Explorer für Benutzer und Administratoren. Weitere Informationen finden Sie unter How To Turn Off Internet Explorer Enhanced Security Configuration auf der Microsoft TechNet-Website.
3. Installieren und konfigurieren Sie Kinesis Agent für Windows auf der Instanz. Weitere Informationen finden Sie unter Installieren von Kinesis Agent für Windows.
4. Erstellen Sie auf der Instance mithilfe von Notepad eine Kinesis Agent for Windows -Konfigurationsdatei. Speichern Sie die Datei unter `%PROGRAMFILES%\Amazon \AWSKinesisTap\appsettings.json`. Fügen Sie der Konfigurationsdatei den folgenden Inhalt hinzu:

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

Diese Datei konfiguriert Kinesis Agent für Windows zum Senden von Protokolldatensätzen im JSON-Format aus Dateien im`c:\logsource\`-Verzeichnis (dassource) in einen Kinesis Data Firehose Delivery-Stream namens`log-delivery-stream`(diesink) enthalten. Bevor die einzelnen Protokolldatensätze an Kinesis Data Firehose gestreamt werden, werden Sie mit zwei zusätzlichen Schlüssel-Wert-Paaren verbessert, die den Namen des Computers und einen Zeitstempel enthalten.

5. Erstellen Sie das Verzeichnis `c:\LogSource\` und erstellen Sie mit Notepad eine `test.log`-Datei in diesem Verzeichnis mit dem folgenden Inhalt:

```
{ "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. Verwenden Sie in einer PowerShell-Sitzung den folgenden Befehl, um den Service AWSKinesisTap zu starten:

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

7. Wechseln Sie mit Datei-Explorer in das Verzeichnis `%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`. Öffnen Sie die neueste Protokolldatei. Die Protokolldatei sollte in etwa folgendermaßen aussehen:

```
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.
```

   Diese Protokolldatei weist darauf hin, dass der Service gestartet wurde und aus dem Verzeichnis `c:\LogSource\` nun Protokolldatensätze erfasst werden. Jede Zeile wird als einzelnes JSON-Objekt analysiert. Jedem Objekt werden Schlüssel-Wert-Paare für den Computernamen und Zeitstempel hinzugefügt. Dann wird es zu Kinesis Data Firehose gestreamt.

8. Navigieren Sie in ein bis zwei Minuten zu dem Amazon S3 Bucket, den Sie in[Erstellen des Amazon S3 Buckets](#)Verwenden der AWS Managementkonsole Stellen Sie sicher, dass Sie in der Konsole die richtige Region ausgewählt haben.

   In diesem Bucket befindet sich ein Ordner für das aktuelle Jahr. Öffnen Sie diesen Ordner, um einen Ordner für den aktuellen Monat anzuzeigen. Öffnen Sie diesen Ordner, um einen Ordner für den aktuellen Tag anzuzeigen. Öffnen Sie diesen Ordner, um einen Ordner für die

aktuelle Stunde (in UTC) anzuzeigen. Öffnen Sie diesen Ordner, um ein oder mehrere Elemente anzuzeigen, die mit dem Namen `log-delivery-stream` beginnen.

9. Öffnen Sie den Inhalt des aktuellen Elements, um zu bestätigen, dass die Protokolldatensätze erfolgreich in Amazon S3 mit den gewünschten Verbesserungen gespeichert wurden. Wenn alles richtig konfiguriert ist, sieht der Inhalt in etwa wie folgt aus:

```
{"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. Weitere Informationen zum Lösen irgendwelcher der folgenden Probleme finden Sie unter Fehlerbehebung bei Amazon Kinesis Agent für Microsoft Windows:

    - Die Kinesis Agent for Windows -Protokolldatei enthält Fehler.
    - Erwartete Ordner oder Elemente in Amazon S3 sind nicht vorhanden.
    - Der Inhalt eines Amazon S3 Artikels ist nicht korrekt.

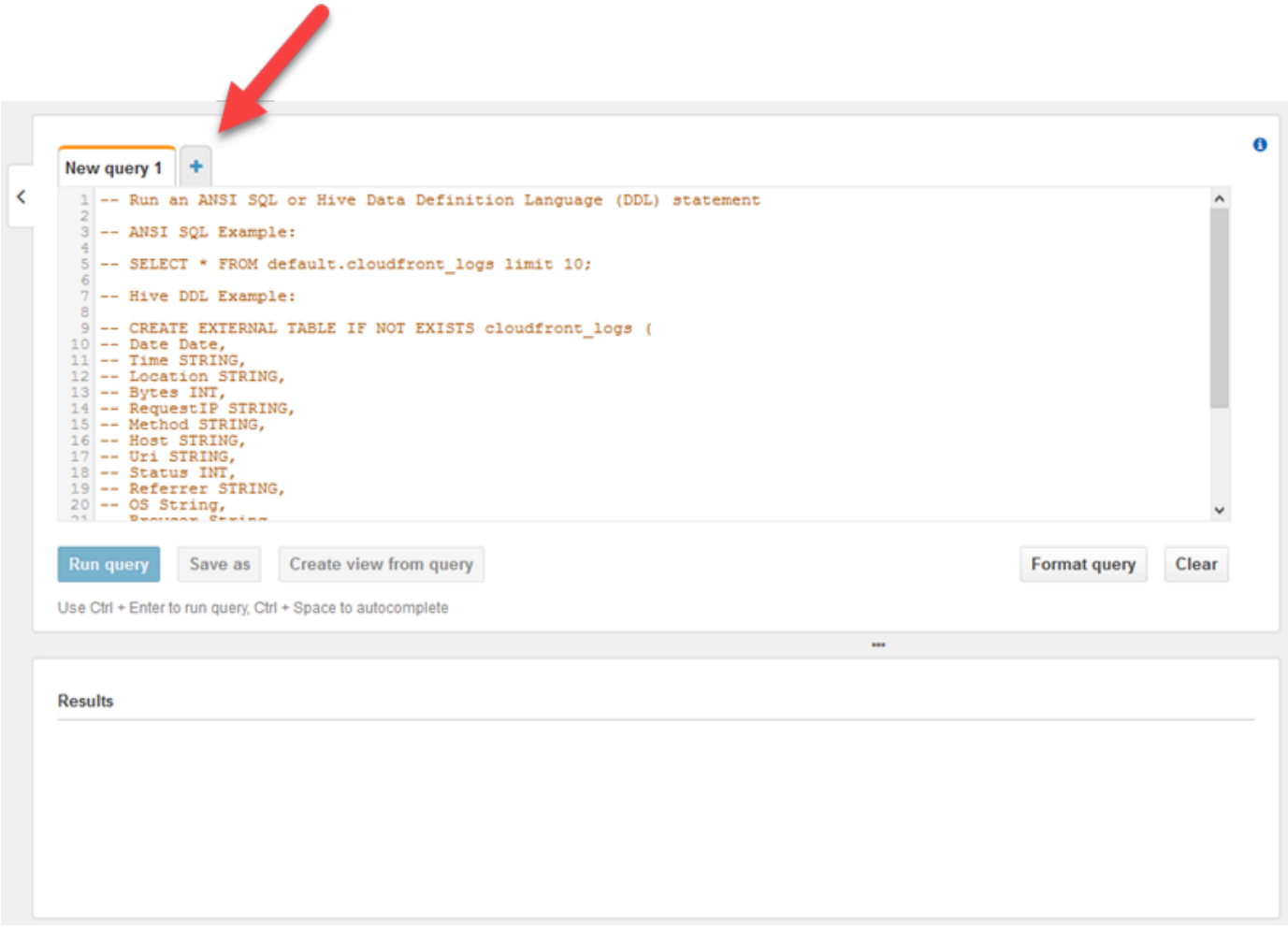
Nächste Schritte

Schritt 3: Abfragen der Protokolldaten in Amazon S3

Schritt 3: Abfragen der Protokolldaten in Amazon S3

Im letzten Schritt dieses Amazon Kinesis Agent für Microsoft Windows-TutorialVerwenden Sie Amazon Athena, um die Protokolldaten abzufragen, die im Amazon Simple Storage Service (Amazon S3) gespeichert sind.

1. Öffnen Sie die Athena-Konsole unter https://console.aws.amazon.com/athena/.
2. Wählen Sie das Pluszeichen (+Klicken Sie auf das Athena Abfragefenster, um ein neues Abfragefenster zu erstellen.



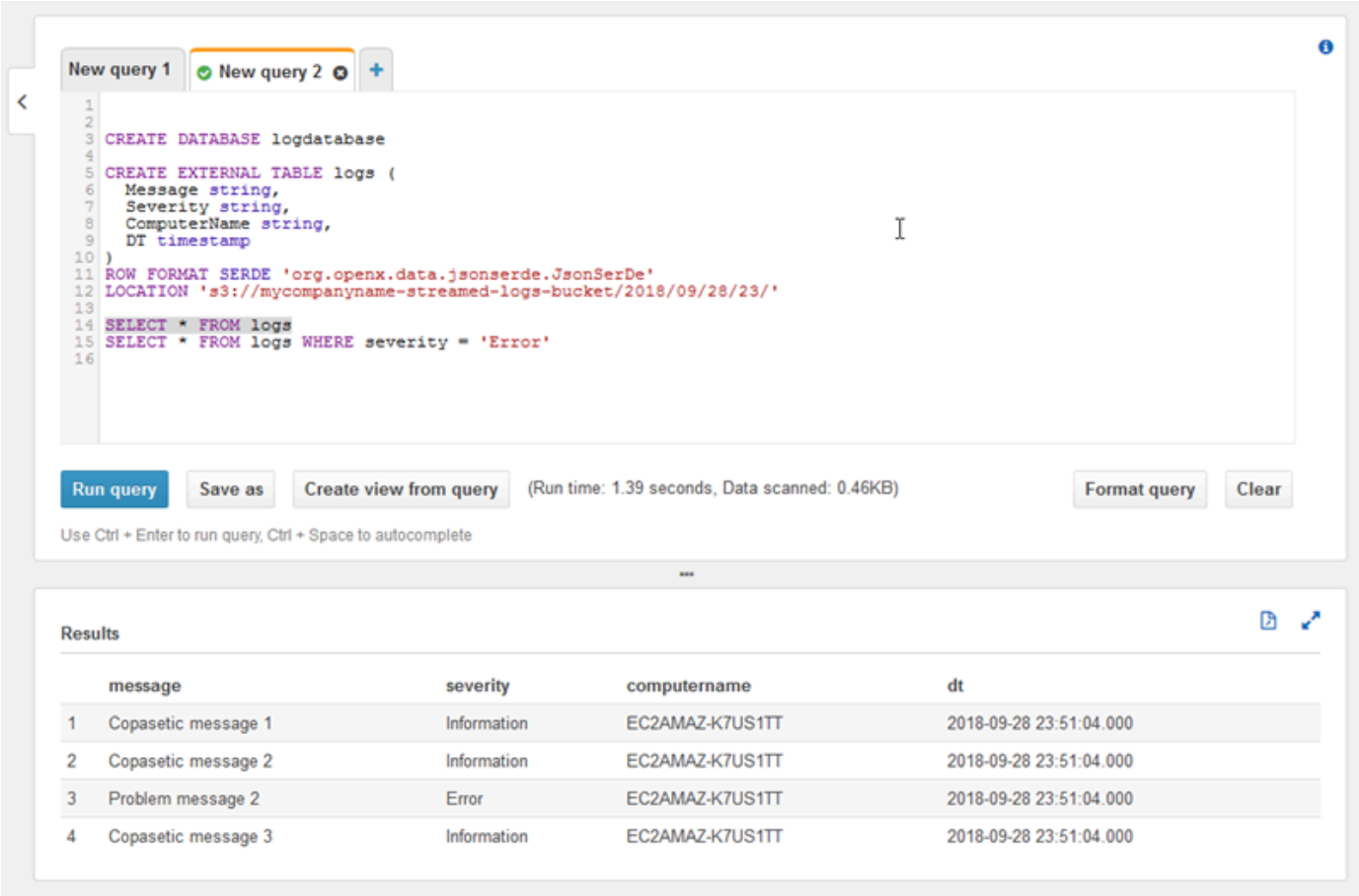
3. Geben Sie den folgenden Text in das Abfragefenster ein:

```
CREATE DATABASE logdatabase

CREATE EXTERNAL TABLE logs (
  Message string,
  Severity string,
```

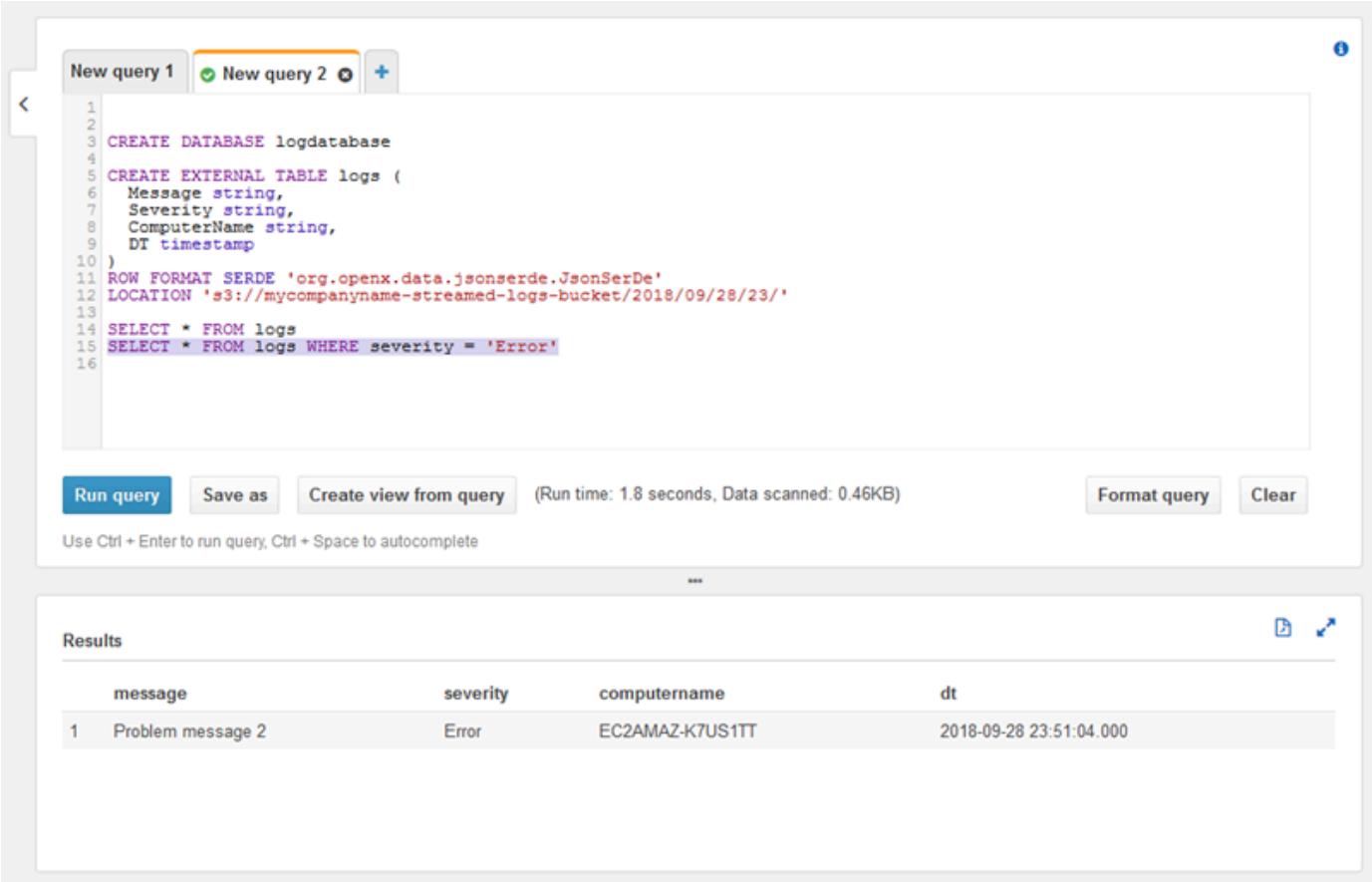
```
  ComputerName string,
  DT timestamp
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://bucket/year/month/day/hour/'

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

Ersetzen Sie *bucket* durch den Namen des Buckets, den Sie unter Erstellen des Amazon S3 Buckets erstellt haben. Ersetzen*year*,*month*,*day*und*hour*Durch das Jahr, den Monat, den Tag und die Stunde, als die Amazon S3 Protokolldatei in koordinierter Weltzeit (UTC) erstellt wurde.

4. Wählen Sie den Text für die `CREATE DATABASE`-Anweisung und klicken Sie dann auf Run query (Abfrage ausführen). Dadurch wird die Protokolldatenbank in Athena erstellt.
5. Wählen Sie den Text für die `CREATE EXTERNAL TABLE`-Anweisung und klicken Sie dann auf Run query (Abfrage ausführen). Dadurch wird eine Athena -Tabelle erstellt, die auf den S3-Bucket mit den Protokolldaten verweist und dabei das JSON-Schema dem Schema für die Athena-Tabelle zuordnet.
6. Wählen Sie den Text für die erste `SELECT`-Anweisung und klicken Sie dann auf Run query (Abfrage ausführen). Dadurch werden Zeilen in der Tabelle angezeigt.



7. Wählen Sie den Text für die zweite `SELECT`-Anweisung und klicken Sie dann auf Run query (Abfrage ausführen). Dadurch werden nur die Zeilen in der Tabelle angezeigt, die Protokolldatensätze mit dem Schweregrad `Error` darstellen. Diese Art von Abfrage sucht interessante Protokolldatensätze in einem potenziell großen Satz von Protokolldatensätzen.



Nächste Schritte

Verwenden Sie die AWS Management Console, um die Ressourcen zu bereinigen, die während des Tutorials erstellt wurden:

1. Beenden Sie die EC2-Instance (siehe Schritt 3 unter Erste Schritte mit Amazon EC2 Windows-Instances).

   > **Important**
   >
   > Wenn Sie eine Instanz gestartet haben, die sich nicht imKostenloses Kontingent für AWSWenn Sie sie beenden, fallen Ihnen für die Instance Gebühren an, bis Sie sie beenden.

2. Löschen Sie den Kinesis Data Firehose Delivery-Stream.

a. Öffnen Sie die Kinesis Data Firehose Konsole unterhttps://console.aws.amazon.com/firehose/.
b. Wählen Sie den von Ihnen erstellten Bereitstellungs-Stream aus.
c. Wählen Sie Delete.
d. Wählen Sie Delete Delivery Stream (Bereitstellungs-Stream löschen) aus.

3. Löschen Sie den S3-Bucket. Detaillierte Anweisungen finden Sie unterWie lösche ich einen S3-Bucket?imUser Guide for Amazon Simple Storage Service Console.

Weitere Informationen finden Sie unter den folgenden Themen:

- Konfigurieren von Amazon Kinesis Agent für Microsoft Windows
- Was ist Amazon Kinesis Data Firehose?
- Was ist Amazon S3?
- Was ist Amazon Athena?

Fehlerbehebung bei Amazon Kinesis Agent für Microsoft Windows

Befolgen Sie die folgenden Anweisungen, um Probleme bei der Verwendung von Amazon Kinesis Agent für Microsoft Windows zu diagnostizieren und zu beheben.

Themen

Es werden keine Daten von Desktops oder Servern an erwartete AWS-Services gestreamt

Symptoms

Wenn Sie Protokolle, Ereignisse und Metriken überprüfen, die von verschiedenen AWS -Services gehostet werden, die so konfiguriert sind, dass sie Datenströme von Kinesis Agent für Windows empfangen, werden von Kinesis Agent für Windows keine Daten gestreamt.

Causes

Für dieses Problem gibt es verschiedene mögliche Ursachen:

- Eine Quelle, Senke oder Pipe ist falsch konfiguriert.
- Die Authentifizierung für Kinesis Agent für Windows ist falsch konfiguriert.
- Die Autorisierung für Kinesis Agent für Windows ist falsch konfiguriert.
- Ein regulärer Ausdruck, der in einer `DirectorySource`-Deklaration bereitgestellt wird, ist fehlerhaft.
- Für eine `DirectorySource`-Deklaration wird ein nicht vorhandenes Verzeichnis angegeben.

- Für AWS -Services werden ungültige Werte bereitgestellt, die dann Anforderungen von Kinesis Agent für Windows ablehnen.
- Eine Senke verweist auf eine Ressource, die in der angegebenen oder implizierten AWS Region nicht vorhanden ist.
- Für eine `WindowsEventLogSource`-Deklaration wird eine ungültige Abfrage angegeben.
- Für das `InitialPosition`-Schlüssel-Wert-Paar für eine Quelle wird ein ungültiger Wert angegeben.
- Die `appsettings.json`-Konfigurationsdatei entspricht nicht dem JSON-Schema für die Datei.
- Die Daten streamen an eine andere Region als die, die in der AWS Management Console ausgewählt ist.
- Kinesis Agent für Windows ist nicht richtig installiert oder wird nicht ausgeführt.

Resolutions

Um Probleme mit nicht gestreamten Daten zu beheben, führen Sie die folgenden Schritte durch:

1. Überprüfen Sie die Kinesis Agent für Windows-Protokolle im`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`-Verzeichnis. Suchen Sie nach der Zeichenfolge `ERROR`.
   a. Wenn eine Quelle oder Senke nicht geladen wurde, gehen Sie wie folgt vor:
      i. Überprüfen Sie die Fehlermeldung und suchen Sie die `Id` der Quelle oder Senke.
      ii. Überprüfen Sie die dieser `Id` entsprechende Quell- oder Senken-Deklaration in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` auf Fehler im Zusammenhang mit der gefundenen Fehlermeldung. Weitere Informationen finden Sie unter Konfigurieren von Amazon Kinesis Agent für Microsoft Windows.
      iii. Korrigieren Sie die Probleme in der Konfigurationsdatei im Zusammenhang mit dem Fehler.
      iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.
   b. Wenn die Fehlermeldung besagt, dass für eine Pipe keine `SourceRef` oder `SinkRef` gefunden wurde, gehen Sie wie folgt vor:
      i. Notieren Sie die `Id` der Pipe.
      ii. Überprüfen Sie die Pipe-Deklaration in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`, die der angegebenen `Id` entspricht. Stellen Sie sicher, dass die Werte der `SourceRef`- und `SinkRef`-Schlüssel-Wert-Paare richtig geschriebene `Id`s für die Quell- und Senken-Deklarationen sind, auf die Sie verweisen

möchten. Korrigieren Sie alle Tipp- oder Rechtschreibfehler. Wenn in der Konfigurationsdatei eine Quell- oder Senken-Deklaration fehlt, fügen Sie die Deklaration hinzu. Weitere Informationen finden Sie unter Konfigurieren von Amazon Kinesis Agent für Microsoft Windows.

iii. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

c. Wenn die Fehlermeldung besagt, dass ein bestimmter IAM-Benutzer oder eine bestimmte Rolle von nicht berechtigt ist, bestimmte Operationen auszuführen, gehen Sie wie folgt vor:

i. Stellen Sie sicher, dass der richtige IAM-Benutzer oder die richtige Rolle von Kinesis Agent für Windows verwendet. Wenn dies nicht der Fall ist, überprüfen SieSenken-Sicherheitskonfigurationund passen Sie an, wie Kinesis Agent für Windows authentifiziert wird, um sicherzustellen, dass der richtige IAM-Benutzer oder die richtige Rolle von verwendet wird.

ii. Wenn der richtige IAM-Benutzer oder die richtige Rolle verwendet wird, untersuchen Sie mithilfe der AWS Management Console die Richtlinien, die dem Benutzer oder der Rolle zugeordnet sind. Stellen Sie sicher, dass der Benutzer oder die Rolle über alle die in der Fehlermeldung erwähnten Berechtigungen für alle AWS verfügt, auf die Kinesis Agent für Windows zugreift. Weitere Informationen finden Sie unter Konfigurieren der Autorisierung.

iii. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Sicherheitsprobleme behoben wurden.

d. Wenn die Fehlermeldung beim Analysieren eines regulären Ausdrucks, der in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` enthalten ist, auf einen Argumentfehler hinweist, gehen Sie wie folgt vor:

i. Überprüfen Sie den regulären Ausdruck in der Konfigurationsdatei.

ii. Überprüfen Sie die Syntax des regulären Ausdrucks. Es gibt mehrere Websites, die Sie zur Überprüfung regulärer Ausdrücke verwenden können, oder verwenden Sie die folgenden Befehlszeilen, um reguläre Ausdrücke für eine `DirectorySource`-Quell-Deklaration zu überprüfen:

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

Ersetzen Sie *`sourceId`* durch den Wert des `Id`-Schlüssel-Wert-Paares der `DirectorySource`-Quell-Deklaration mit einem falschen regulären Ausdruck.

iii. Nehmen Sie alle erforderlichen Korrekturen am regulären Ausdruck in der Konfigurationsdatei vor, sodass er gültig ist.

iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

e. Wenn die Fehlermeldung beim Analysieren eines regulären Ausdrucks, der nicht in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` enthalten ist, auf einen Argumentfehler hinweist und sich auf eine bestimmte Senke bezieht, gehen Sie wie folgt vor:

i. Suchen Sie die Senken-Deklaration in der Konfigurationsdatei.

ii. Stellen Sie sicher, dass die Schlüssel-Wert-Paare, die sich speziell auf einen AWS-Service beziehen, Namen verwenden, die den Validierungsregeln für diesen Service entsprechen. Beispiel #Gruppennamen von CloudWatch Logs dürfen nur einen bestimmten Zeichensatz enthalten, der der mit dem regulären Ausdruck angegeben wird`[\.\-_/#A-Za-z0-9]+`.

iii. Korrigieren Sie alle ungültigen Namen in den Schlüssel-Wert-Paaren für die Senken-Deklaration und stellen Sie sicher, dass diese Ressourcen in AWS ordnungsgemäß konfiguriert sind.

iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

f. Wenn die Fehlermeldung besagt, dass eine Quelle oder Senke aufgrund eines Nullparameters oder eines fehlenden Parameters nicht geladen werden kann, gehen Sie wie folgt vor:

i. Merken Sie sich die `Id` der Quelle oder Senke.

ii. Suchen Sie die Quell- oder Senken-Deklaration, die der gemerkten `Id` in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` entspricht.

iii. Überprüfen Sie die Schlüssel-Wert-Paare, die in der Quell- oder Senken-Deklaration bereitgestellt werden, im Vergleich mit den Anforderungen der Quell- oder Senken-Typen unter Konfigurieren von Amazon Kinesis Agent für Microsoft Windows für den relevanten Senken-Typ. Fügen Sie alle fehlenden erforderlichen Schlüssel-Wert-Paare zur Quell- oder Senken-Deklaration hinzu.

iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

g. Wenn die Fehlermeldung besagt, dass ein Verzeichnisname ungültig ist, führen Sie die folgenden Schritte durch:

   i. Suchen Sie den ungültigen Verzeichnisnamen in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

   ii. Überprüfen Sie, ob dieses Verzeichnis vorhanden ist und ob es die Protokolldateien enthält, die gestreamt werden sollen.

   iii. Korrigieren Sie Tippfehler oder Fehler des in der Konfigurationsdatei angegebenen Verzeichnisnamens.

   iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

h. Wenn die Fehlermeldung besagt, dass eine Ressource nicht vorhanden ist:

   i. Suchen Sie die Ressource-Referenz für Ressourcen, die nicht in einer Senken-Deklaration enthalten sind, in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

   ii. Suchen Sie mithilfe der AWS Management Console die Ressource in der richtigen AWS-Region, die in der Senken-Deklaration verwendet werden sollte. Vergleichen Sie sie mit der Angabe in der Konfigurationsdatei.

   iii. Ändern Sie die Senken-Deklaration in der Konfigurationsdatei auf den richtigen Ressourcennamen und die richtige Region.

   iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

i. Wenn die Fehlermeldung besagt, dass eine Abfrage für eine bestimmte `WindowsEventLogSource` ungültig ist, führen Sie die folgenden Schritte durch:

   i. Suchen Sie in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` die `WindowsEventLogSource`-Deklaration mit derselben `Id` wie in der Fehlermeldung.

   ii. Stellen Sie sicher, dass der Wert des `Query`-Schlüssel-Wert-Paares in der Quell-Deklaration mit dem unter Event queries and Event XML übereinstimmt.

   iii. Nehmen Sie die gewünschten Änderungen an der Abfrage vor, damit sie die Compliance-Anforderungen erfüllt.

   iv. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

j. Wenn die Fehlermeldung besagt, dass eine anfängliche Position ungültig ist, führen Sie die folgenden Schritte durch:

i. Suchen Sie in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` die Quell-Deklaration mit derselben `Id` wie in der Fehlermeldung.

ii. Ändern Sie den Wert des `InitialPosition`-Schlüssel-Wert-Paares in der Quell-Deklaration, sodass er den zulässigen Werten entspricht, wie unter Lesezeichen-Konfiguration beschrieben.

iii. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

2. Stellen Sie sicher, dass die Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` dem JSON-Schema entspricht.

   a. Rufen Sie in einem Befehlszeilenfenster die folgenden Zeilen auf:

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

   b. Korrigieren Sie alle Probleme, die bezüglich der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` erkannt wurden.

   c. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend in der neuesten Protokolldatei, ob die Konfigurationsprobleme behoben wurden.

3. Ändern Sie die Protokollierungsebene, um detailliertere Protokollierungsinformationen zu erhalten.

   a. Ersetzen Sie die Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\nlog.xml` durch den folgenden Inhalt:

   ```
   <?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. Stoppen und starten Sie den `AWSKinesisTap`-Service. Überprüfen Sie anschließend die neueste Protokolldatei auf zusätzliche Meldungen im Protokoll, die bei der Diagnose und Lösung des Problems hilfreich sein könnten.

4. Stellen Sie sicher, dass Sie in der AWS Management Console Ressourcen in der richtigen Region anzeigen.
5. Stellen Sie sicher, dass der Kinesis Agent für Windows -Agent installiert ist und ausgeführt wird.
   a. Wählen Sie in Windows Start und navigieren Sie dann zu Systemsteuerung, Verwaltung, Services.
   b. Suchen Sie den Service AWSKinesisTap.
   c. Wenn der AWSKinesis Agent für Windows nicht zu sehen ist, installieren Sie Kinesis Agent für Windows gemäß den Anweisungen unterErste Schritte mit Amazon Kinesis Agent für Microsoft Windows.
   d. Wenn der Service zu sehen ist, bestimmen Sie, ob er derzeit ausgeführt wird. Wenn er nicht ausgeführt wird, öffnen Sie das Kontextmenü (Rechtsklick) für den Service und wählen Sie Start.

e. Überprüfen Sie, ob der Dienst gestartet wurde, indem Sie die neueste Protokolldatei im Verzeichnis `%PROGRAMDATA%\Amazon\AWSKinesisTap\logs` untersuchen.

Gilt für

Diese Informationen gelten für Kinesis Agent für Windows, Version 1.0.0.115 und höher.

Erwartete Daten fehlen manchmal

Symptoms

Kinesis Agent für Windows streamt Daten die meiste Zeit erfolgreich, gelegentlich kann es aber vorkommen, dass Daten fehlen.

Causes

Für dieses Problem gibt es verschiedene mögliche Ursachen:

- Die Funktion zum Setzen eines Lesezeichens wird nicht verwendet.
- Die Datenraten-Limits für AWS -Services werden basierend auf der aktuellen Konfiguration dieser Services überschritten.
- Die API-Aufrufraten-Limits für AWS -Services werden basierend auf dem aktuellen`appsettings.json`-Konfigurationsdatei und die AWS Kontobeschränkungen.

Resolutions

Um Probleme mit fehlenden Daten zu beheben, führen Sie die folgenden Schritte durch:

1. Erwägen Sie, die Funktion zum Setzen von Lesezeichen zu verwenden, die unter Lesezeichen-Konfiguration dokumentiert wird. Damit wird sichergestellt, dass alle Daten letztendlich gesendet werden, selbst wenn Kinesis Agent für Windows angehalten und gestartet wird.
2. Verwenden Sie die integrierten Metriken von Kinesis Agent für Windows, um Probleme zu ermitteln:
   a. Aktivieren Sie das Streamen von Kinesis Agent für Windows-Metriken, wie unter beschriebenKonfigurieren von Kinesis Agent für Windows-Metrik-Pipes.

b. Tritt eine erhebliche Anzahl von nicht wiederherstellbaren Fehlern für eine oder mehrere Senken auf, bestimmen Sie, wie viele Bytes oder Datensätze pro Sekunde gesendet werden. Bestimmen Sie anschließend, ob der Wert innerhalb der Limits liegt, die für diese AWS-Services in der Region und in dem Konto konfiguriert sind, wo die Daten gestreamt werden.

c. Wenn Limits überschritten werden, reduzieren Sie entweder die Rate oder die Menge der zu sendenden Daten, fordern Sie Erhöhungen des Limits an oder erhöhen Sie das Sharding, sofern zutreffend.

d. Nachdem Sie Anpassungen vorgenommen haben, fahren Sie mit der Überwachung der integrierten -Metriken von Kinesis Agent für Windows fort, um sicherzustellen, dass die Situation behoben wurde.

Weitere Informationen zu Kinesis Data Streams Limits finden Sie unterLimKinesis Data Streams LimitsimKinesis Data Streams Entwicklerhandbuch. Weitere Informationen zu Kinesis Data Firehose hose-Limits finden Sie unterAmazon Kinesis Data Firehose -Limits.

Gilt für

Diese Informationen gelten für Kinesis Agent für Windows, Version 1.0.0.115 und höher.

Daten treffen im falschen Format ein

Symptoms

Die Daten treffen beim AWS -Service im falschen Format ein.

Causes

Für dieses Problem gibt es verschiedene mögliche Ursachen:

- Der Wert für das `Format`-Schlüssel-Wert-Paar für eine Senken-Deklaration in der Konfigurationsdatei `appsettings.json` ist falsch.
- Der Wert für das `RecordParser`-Schlüssel-Wert-Paar in einer `DirectorySource`-Deklaration ist falsch.
- Die regulären Ausdrücke in einer `DirectorySource`-Deklaration, die den `Regex`-Datensatz-Parser verwendet, sind falsch.

Resolutions

Um Probleme mit inkorrekter Formatierung zu lösen, führen Sie die folgenden Schritte durch:

1. Überprüfen Sie die Senken-Deklarationen in der Konfigurationsdatei `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.
2. Stellen Sie sicher, dass der korrekte Wert des `Format`-Schlüssel-Wert-Paars für jede Senken-Deklaration angegeben wird. Weitere Informationen finden Sie unter Senken-Deklarationen.
3. Wenn Quellen mit `DirectorySource`-Deklarationen über Pipes mit Senken verbunden sind, die für das `Format`-Schlüssel-Wert-Paar die Werte `xml` oder `json` angeben, stellen Sie sicher, dass diese Quellen einen der folgenden Werte für das `RecordParser`-Schlüssel-Wert-Paar angeben:
   - `SingleLineJson`
   - `Regex`
   - `SysLog`
   - `Delimited`

   Andere Datensatz-Parser sind nur textbasiert und funktionieren nicht richtig mit Senken, für die eine XML- oder JSON-Formatierung erforderlich ist.
4. Wenn Protokolldatensätze von dem `DirectorySource`-Quelltyp nicht korrekt analysiert werden, rufen Sie die folgenden Zeilen in einem Befehlszeilenfenster auf, um den Zeitstempel und die Schlüssel-Wert-Paare des regulären Ausdrucks zu überprüfen, die in der `DirectorySource`-Deklaration angegeben werden:

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

   Ersetzen Sie *`sourceId`* durch den Wert des `Id`-Schlüssel-Wert-Paares der `DirectorySource`-Quell-Deklaration, die nicht korrekt zu funktionieren scheint. Korrigieren Sie alle von `ktdiag.exe` gemeldeten Probleme.

Gilt für

Diese Informationen gelten für Kinesis Agent für Windows, Version 1.0.0.115 und höher.

Leistungsprobleme

Symptoms

Anwendungen und -Services haben erhöhte Latenzen, nachdem Kinesis Agent für Windows installiert und gestartet wurde.

Causes

Für dieses Problem gibt es verschiedene mögliche Ursachen:

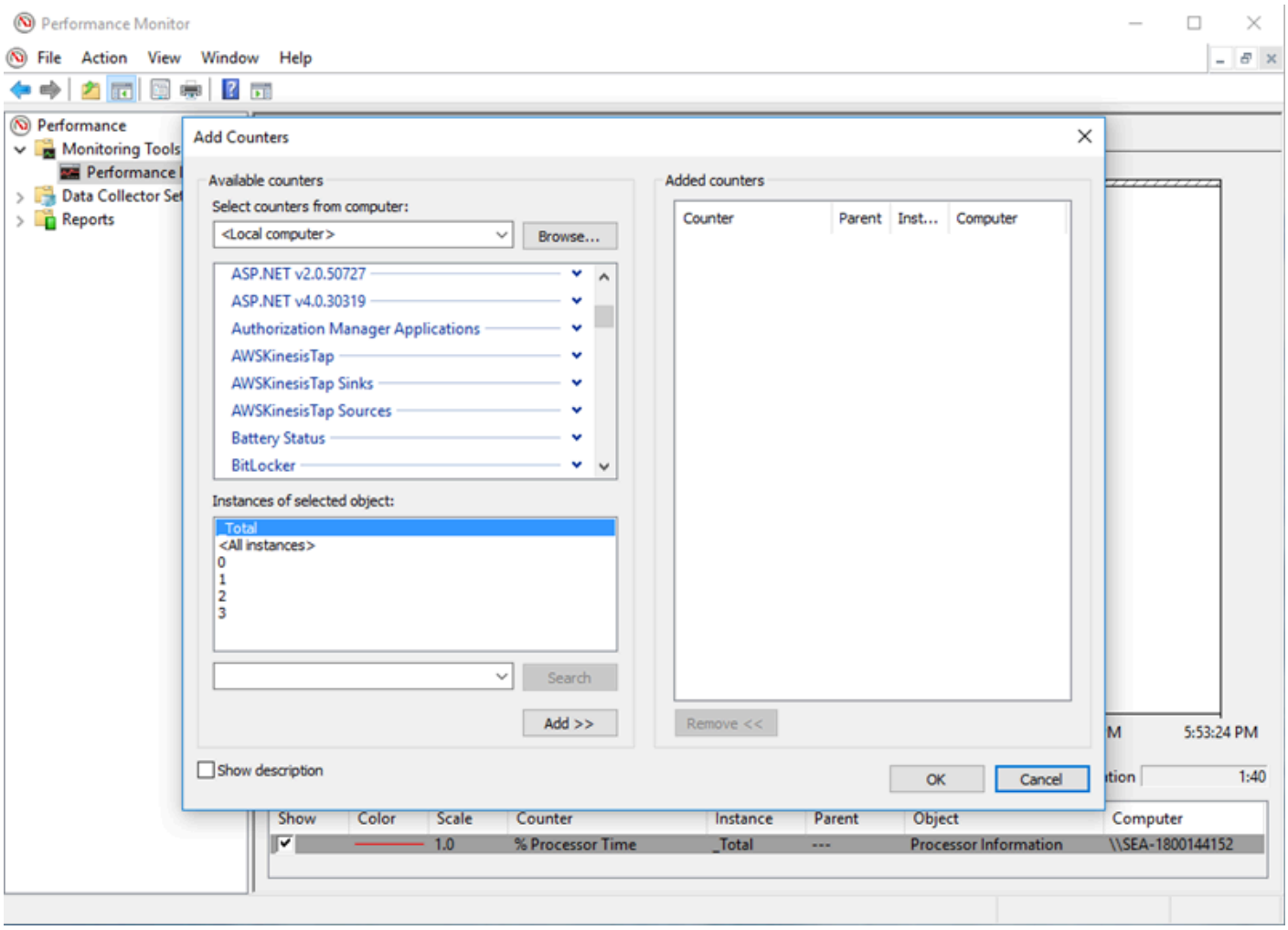
- Der Computer, auf dem Kinesis Agent für Windows ausgeführt wird, verfügt nicht über ausreichend Kapazität zum Streamen der gewünschten Datenmenge.
- Unnötige Daten werden an einen oder mehrere AWS -Services gestreamt.
- Kinesis Agent für Windows streamt Daten an AWS -Services, die nicht für eine solche hohe Datenrate konfiguriert sind.
- Kinesis Agent für Windows ruft Vorgänge auf AWS -Services in einem Konto auf, in dem das Limit der API-Aufrufrate zu niedrig ist.

Resolutions

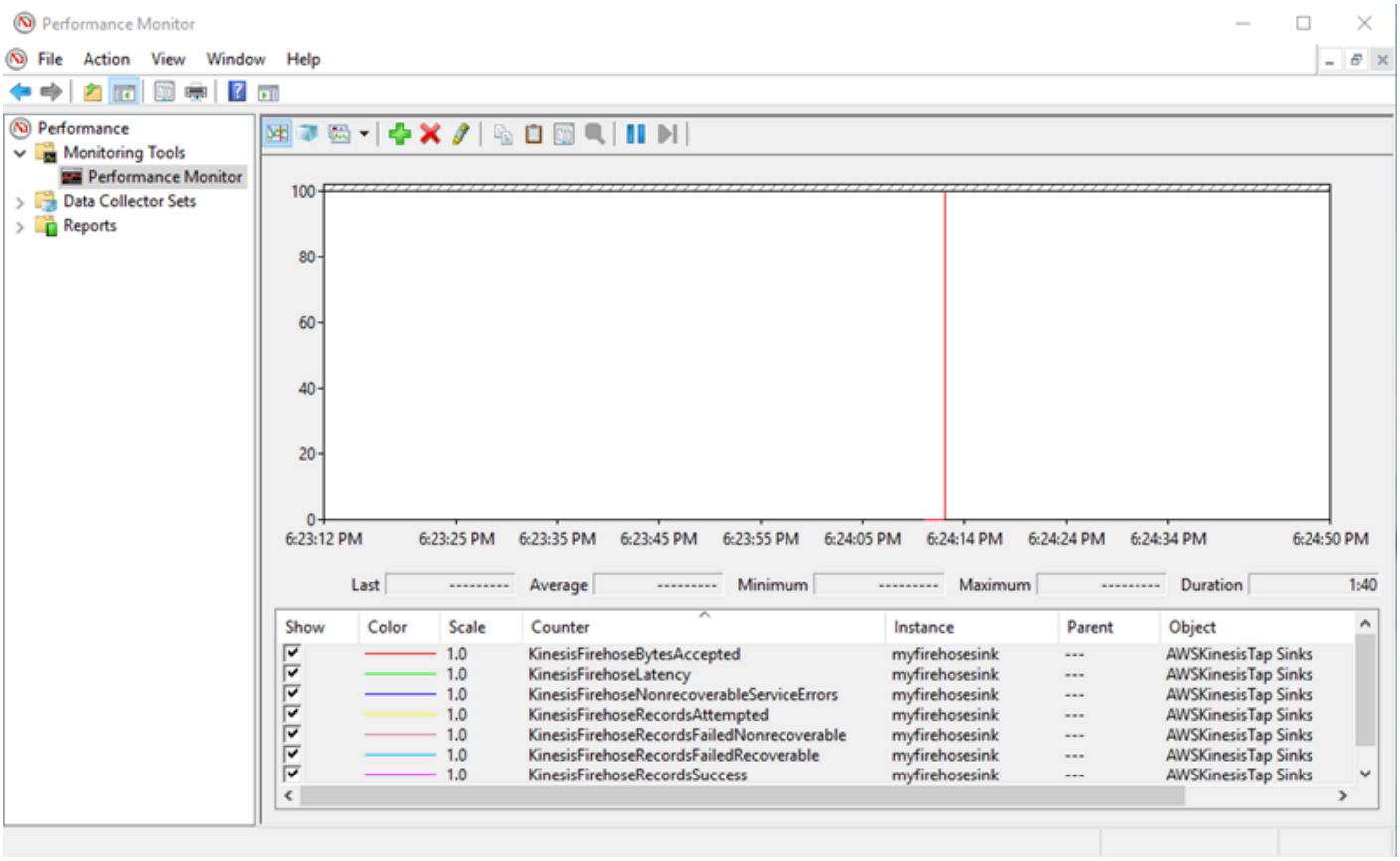
Um Leistungsprobleme zu beheben, führen Sie die folgenden Schritte durch:

1. Verwenden Sie die Windows-Anwendung zur Ressourcenüberwachung, um die Arbeitsspeicher-, CPU-, Festplatten- und Netzwerknutzung zu überprüfen. Wenn Sie mit Kinesis Agent für Windows große Datenmengen streamen möchten, müssen Sie einem Computer in einigen dieser Bereiche je nach Konfiguration möglicherweise höhere Kapazitäten zur Verfügung stellen.
2. Möglicherweise können Sie die Menge der protokollierten Daten mithilfe von Filtern verringern:
   - Siehe `Query`-Schlüssel-Wert-Paar unter WindowsEventLogSource-Konfiguration.
   - Siehe Pipeline-Filterung unter Konfigurieren von Pipes.
   - Weitere Informationen finden Sie unter Amazon CloudWatch Metrik-Filterung unterKonfiguration der CloudWatch Senken-Konfiguration) enthalten.
3. Verwenden Sie die Windows-Leistungsüberwachungsanwendung, um Kinesis Agent für Windows-Metriken anzuzeigen oder diese Metriken an CloudWatch zu streamen (siheIntegrierte Quelle von Kinesis Agent für Windows -Metriken: Quelle) enthalten. In der Windows-Leistungsüberwachungsanwendung können Sie Zähler für Kinesis Agent für Windows Senks

und -Quellen hinzufügen. Sie werden unter den Indikatorkategorien AWSKinesisTap und AWSKinesisTap Sources aufgelistet.



Wenn Sie z. B. -Leistungsprobleme von Kinesis Data Firehose diagnostizieren, fügen Sie dasKinesis Firehose-Senke-Leistungsindikatoren.



Wenn eine große Anzahl von behebbaren Fehlern auftritt, überprüfen Sie die neuesten Kinesis Agent für Windows-Protokolle in der`%PROGRAMDATA%\Amazon\AWSKInesisTap\logs`-Verzeichnis. Wenn bei `KinesisStream`- oder `KinesisFirehose`-Senken eine Drosselung auftritt, gehen Sie wie folgt vor:

- Wenn der Grund für die Drosselung zu schnell gestreamte Daten sind, erwägen Sie, die Anzahl der Shards für den Kinesis Data Stream zu erhöhen. Weitere Informationen finden Sie unterResharding, Skalierung und ParallelverarbeitungimKinesis Data Streams Entwicklerhandbuch.
- Erwägen Sie, das Limit der API-Aufrufe für Kinesis Data Streams oder die Puffergröße für die Senke zu erhöhen, wenn die API-Aufrufe gedrosselt werden. Weitere Informationen finden Sie unterLimKinesis Data Streams LimitsimKinesis Data Streams Entwicklerhandbuch.
- Wenn Daten zu schnell gestreamt werden, erwägen Sie, eine Erhöhung des Ratenlimits für den -Bereitstellungs-Stream von Kinesis Data Firehose zu beantragen. Oder wenn die API-Aufrufe gedrosselt werden, beantragen Sie eine Erhöhung des API-Aufruf-Limits (siehe Amazon Kinesis Data Firehose-Limits) oder erhöhen Sie die Puffergröße für die Senke.
- Nachdem Sie die Anzahl der Shards für einen Kinesis Data Streams am oder das Ratenlimit für einen -Bereitstellungs-Stream von Kinesis Data Firehose erhöht haben, ändern Sie den

Kinesis Agent für Windows`appsettings.json`-Konfigurationsdatei, um die Datensätze pro Sekunde oder die Bytes pro Sekunde für die Senke zu erhöhen. Andernfalls kann Kinesis Agent für Windows die erhöhten Limits nicht nutzen.

Gilt für

Diese Informationen gelten für Kinesis Agent für Windows, Version 1.0.0.115 und höher.

Kein Speicherplatz mehr

Symptoms

Kinesis Agent für Windows wird auf einem Computer mit sehr wenig Speicherplatz auf einem oder mehreren Festplattenlaufwerken ausgeführt.

Causes

Für dieses Problem gibt es verschiedene mögliche Ursachen:

- Die -Protokollierungs-Konfigurationsdatei für Kinesis Agent für Windows ist falsch.
- Die persistente Kinesis Agent für Windows -Warteschlange ist falsch konfiguriert.
- Der Speicherplatz wird von einer anderen Anwendung oder einem anderen Service belegt.

Resolutions

Um Probleme mit dem Speicherplatz zu lösen, führen Sie die folgenden Schritte durch:

- Wenn der Speicherplatz auf dem Datenträger mit den -Kinesis Agent für Windows-Protokolldateien niedrig ist, untersuchen Sie das Verzeichnis der Protokolldatei (in der Regel`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`) enthalten. Stellen Sie sicher, dass eine angemessene Anzahl von Protokolldateien beibehalten wurde und dass die Protokolldateien eine angemessene Größe aufweisen. Sie können den Speicherort, die Beibehaltung und die Ausführlichkeit der Kinesis Agent für Windows-Protokolle steuern, indem Sie die`%PROGRAMFILES%\Amazon\AWSKinesisTap\Nlog.xml`-Konfigurationsdatei.
- Wenn die Senken-Warteschlangenfunktion aktiviert ist, überprüfen Sie die Senken-Deklarationen, die diese Funktion verwenden. Stellen Sie sicher, dass das`QueuePath`-Schlüssel-Wert-Paar auf

ein Laufwerk mit ausreichendem Speicherplatz für die maximale Anzahl von Stapeln verweist, die mit dem `QueueMaxBatches`-Schlüssel-Wert-Paar angegeben wird. Wenn dies nicht möglich ist, reduzieren Sie den Wert des `QueueMaxBatches`-Schlüssel-Wert-Paares, sodass die Daten problemlos in den verbleibenden Speicherplatz für das angegebene Festplattenlaufwerk passen.

- Suchen Sie mit dem Windows-Explorer die Dateien, die den Speicherplatz belegen, und übertragen oder löschen Sie überschüssige Dateien. Ändern Sie die Konfiguration der Anwendungen oder Services, von denen große Mengen an Speicherplatz belegt werden.

Gilt für

Diese Informationen gelten für Kinesis Agent für Windows, Version 1.0.0.115 und höher.

Tools zur Fehlerbehebung

Zusätzlich zur Überprüfung der -Konfigurationsdatei können Sie die`ktdiag.exe`Tool, das einige weitere Funktionen für die Diagnose und Behebung von Problemen bei der Konfiguration und Nutzung von Kinesis Agent für Windows bietet. Das Tool `ktdiag.exe`befindet sich im Verzeichnis `%PROGRAMFILES%\Amazon\AWSKinesisTap`.

- Wenn Sie der Meinung sind, dass Protokolldateien mit einem bestimmten Dateimuster in ein Verzeichnis geschrieben, von Kinesis Agent für Windows aber nicht verarbeitet werden, verwenden Sie das`/w`, um zu überprüfen, ob diese Änderungen erkannt werden. Angenommen, Sie erwarten, dass Protokolldateien mit dem Namensmuster `*.log` in das Verzeichnis `c:\foo` geschrieben werden. Sie können in diesem Fall den Schalter `/w` beim Ausführen des Tools `ktdiag.exe` unter Angabe des Verzeichnisses und des Dateimusters verwenden:

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

  Wenn Protokolldateien geschrieben werden, können Sie eine Ausgabe ähnlich der folgenden sehen:

  ```
  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
```

Wenn keine solche Ausgabe zu sehen ist, ist beim Schreiben der Protokolle ein Anwendungs- oder Serviceproblem aufgetreten oder liegt anstelle eines Problems mit dem Kinesis Agent für Windows ein Problem mit der Sicherheitskonfiguration vor. Wenn trotz einer solchen Ausgabe die Protokolle anscheinend immer noch nicht zu verarbeiten scheint, finden Sie weitere Informationen unterEs werden keine Daten von Desktops oder Servern an erwartete AWS-Services gestreamt.

- Manchmal werden Protokolle nur gelegentlich geschrieben, es wäre aber nützlich zu überprüfen, ob Kinesis Agent für Windows ordnungsgemäß funktioniert. Mit dem Schalter `/log4net` können Sie eine Anwendung simulieren, die Protokolle mithilfe der `Log4net`-Bibliothek schreibt, z. B.:

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

Hiermit wird eine `Log4net`-artige Protokolldatei in die Protokolldatei `c:\foo\log2.log` geschrieben und es werden fortgesetzt neue Protokolleinträge hinzugefügt, bis ein Schlüssel gedrückt wird. Mithilfe zusätzlicher Schalter, die optional nach dem Dateinamen angegeben werden, können Sie mehrere Optionen konfigurieren:

Sperren: `-lm`, `-li` oder `-le`

Sie können einen der folgenden Schalter für die Sperrung angeben, die steuern, wie die Protokolldatei gesperrt wird:

`-lm`

Für die Protokolldatei wird ein minimales Maß an Sperrung verwendet, um maximalen Zugriff auf die Protokolldatei zu ermöglichen.

`-li`

Nur Threads innerhalb desselben Prozesses können gleichzeitig auf das Protokoll zugreifen.

`-le`

Es kann nur jeweils ein Thread auf das Protokoll zugreifen. Dies ist die Standardeinstellung.

`-tn:`*`Millisekunden`*

Gibt die Anzahl von *`Millisekunden`* zwischen dem Schreiben von Protokolleinträgen an. Der Standardwert ist 1000 Millisekunden (1 Sekunde).

`-sm:`*`Bytes`*

Gibt die Anzahl der *`Bytes`* für jeden Protokolleintrag an. Der Standardwert lautet 1000 Bytes.

`-bk:`*`Zahl`*

Gibt die *`Anzahl`* der Protokolleinträge an, die jeweils geschrieben werden soll. Der Standardwert ist 1.

- Manchmal ist es sinnvoll, eine Anwendung simulieren, die Schreibvorgänge für das Windows-Ereignisprotokoll durchführt. Verwenden Sie den Schalter `/e`, um Protokolleinträge eines Windows-Ereignisprotokolls zu schreiben, z. B.:

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

Dadurch werden Protokolleinträge in das Ereignisprotokoll der Windows-Anwendung geschrieben, bis eine Taste gedrückt wird. Sie können nach dem Namen des Protokolls optional die folgenden zusätzlichen Optionen angeben:

`-tn:`*`Millisekunden`*

Gibt die Anzahl von *`Millisekunden`* zwischen dem Schreiben von Protokolleinträgen an. Der Standardwert ist 1000 Millisekunden (1 Sekunde).

`-sm:`*`Bytes`*

Gibt die Anzahl der *`Bytes`* für jeden Protokolleintrag an. Der Standardwert lautet 1000 Bytes.

`-bk:`*`Zahl`*

Gibt die *`Anzahl`* der Protokolleinträge an, die jeweils geschrieben werden soll. Der Standardwert ist 1.

Erstellen von Kinesis Agent für Windows-Plugins

In den meisten Situationen ist das Erstellen eines Amazon Kinesis Agent für Microsoft Windows Plugins nicht erforderlich. Kinesis Agent für Windows ist vielseitig konfigurierbar und enthält leistungsstarke Quellen und Senken, wie z. B.`DirectorySource`und`KinesisStream`, die für die meisten Szenarien ausreichen. Weitere Informationen zu den vorhandenen Quellen und Senken finden Sie unter Konfigurieren von Amazon Kinesis Agent für Microsoft Windows.

Unter ungewöhnliche Umständen ist es möglicherweise erforderlich, Kinesis Agent für Windows mit einem benutzerdefinierten Plug-In zu erweitern. Einige dieser Fälle sind die folgenden:

- Paketerstellung einer komplexen `DirectorySource`-Deklaration unter Verwendung der Datensatz-Parser `Regex` oder `Delimited`, sodass sie in vielen verschiedenen Konfigurationsdatensätzen angewendet werden kann.
- Erstellen einer neuen Quelle, die nicht dateibasiert ist oder die Analysefunktionen überschreitet, die durch vorhandene Datensatz-Parser geboten werden.
- Erstellen einer Senke für einen AWS-Service, der derzeit nicht unterstützt wird.

Themen

Erste Schritte mit Kinesis Agent für Windows-Plugins

Es gibt keine speziellen Informationen zu benutzerdefinierten Plug-Ins. Alle vorhandenen Quellen und Senken verwenden dieselben Mechanismen, die benutzerdefinierte Plug-Ins zum Laden verwenden, wenn Kinesis Agent für Windows gestartet wird, und instanziieren relevante Plug-Ins, nachdem die`appsettings.json`-Konfigurationsdatei.

Beim Starten von Kinesis Agent für Windows erfolgt die folgende Reihenfolge:

1. Kinesis Agent für Windows scannt Komponenten im Installationsverzeichnis (`%PROGRAMFILES%\Amazon\AWSKinesisTap`) für Klassen, die die`IFactory<T>`-

Schnittstelle, die in der`Amazon.KinesisTap.Core`Baugruppe Diese Schnittstelle wird in definiert`Amazon.KinesisTap.Core\Infrastructure\IFactory.cs`im Kinesis Agent für Windows-Quellcode.

2. Kinesis Agent für Windows lädt die Komponenten, die diese Klassen enthalten, und ruft die`RegisterFactory`-Methode für diese Klassen.
3. Kinesis Agent für Windows lädt die`appsettings.json`-Konfigurationsdatei. Für jede Quelle und Senke in der Konfigurationsdatei werden die Schlüssel-Wert-Paare `SourceType` und `SinkType` geprüft. Wenn Factories mit demselben Namen wie die Werte der Schlüssel-Wert-Paare `SourceType` und `SinkType` registriert sind, wird für diese Factories die Methode `CreateInstance` aufgerufen. Der Methode `CreateInstance` werden die Konfiguration und andere Informationen als ein `IPluginContext`-Objekt übergeben. Die Methode `CreateInstance` ist für das Konfigurieren und Initialisieren des Plug-Ins bestimmt.

Damit ein Plug-In korrekt ausgeführt wird, muss eine registrierte Factory-Klasse vorhanden sein, die das Plug-In erstellt, und die Plug-In-Klasse selbst muss definiert sein.

Der -Quellcode von Kinesis Agent für Windows befindet sich unterhttps://github.com/awslabs/kinesis-agent-windows.

Implementieren von Kinesis Agent für Windows Plugin-Fabriken

Führen Sie die folgenden Schritt durch, um eine Kinesis g-In-Factory zu implementieren.

So erstellen Sie eine Kinesis Agent für Windows-Plugin-Factory

1. Erstellen Sie ein C#-Bibliotheksprojekt für .NET Framework 4.6.
2. Fügen Sie einen Verweis auf die Komponente `Amazon.KinesisTap.Core` hinzu. Diese Baugruppe befindet sich im`%PROGRAMFILES%\Amazon\AWSKinesisTap`-Verzeichnis nach der Installation von Kinesis Agent für Windows.
3. Installieren Sie mithilfe von `NuGet` das Paket `Microsoft.Extensions.Configuration.Abstractions`.
4. Installieren Sie mithilfe von `NuGet` das Paket `System.Reactive`.
5. Installieren Sie mithilfe von `NuGet` das Paket `Microsoft.Extensions.Logging`.
6. Erstellen Sie eine Factory-Klasse, mit der entweder `IFactory<IEventSource>` für Quellen oder `IFactory<IEventSink>` für Senken implementiert wird. Fügen Sie die `RegisterFactory`- oder `CreateInstance`-Methoden hinzu.

Beispielsweise erstellt der folgende Code eine -Plug-In-Factory von Kinesis Agent für Windows, mit der eine Quelle erstellt wird, mit der zufällige Daten generiert werden:

```
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);
            }
        }
    }
}
```

Die `switch`-Anweisung wird in der Methode `CreateInstance` verwendet, falls Sie die Factory irgendwann zum Erstellen verschiedener Arten von Instances erweitern möchten.

Um eine Senken-Factory zu erstellen, mit der eine aktionslose Senke erstellt wird, verwenden Sie eine Klasse ähnlich der folgenden:

```
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}.");
            }
        }
    }
}
```

Implementieren von Kinesis Agent für Windows Plugin-Quellen

Führen Sie die folgenden Schritt durch, um Kinesis lug-In-Quelle zu implementieren.

So erstellen Sie eine Kinesis Agent für Windows-Plugin-Quelle

1. Fügen Sie eine Klasse hinzu, mit der die `IEventSource<out T>`-Schnittstelle zu dem zuvor für die Quelle erstellten Projekt hinzufügt wird.

   Verwenden Sie z. B. den folgenden Code, um eine Quelle zu definieren, mit der zufällige Daten generiert werden:

```
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();
        }
    }
}
```

In diesem Beispiel erbt die Klasse `RandomSource` von der Klasse `EventSource<T>`, da sie die Eigenschaft `Id` bereitstellt. Obwohl dieses Beispiel nicht das Setzen von Lesezeichen unterstützt, ist diese Basisklasse auch für die Implementierung dieser Funktionalität nützlich. Envelopes bieten eine Möglichkeit zum Speichern von Metadaten und Verpacken beliebiger Daten zum Streamen an Senken. Die Klasse `RandomData` wird im nächsten Schritt definiert und stellt den Typ des Ausgabeobjekts aus dieser Quelle dar.

2. Fügen Sie eine Klasse zu dem zuvor definierten Projekt hinzu, das die Daten enthält, die aus der Quelle gestreamt werden.

Beispiel: Ein Container für zufällige Daten könnte wie folgt definiert werden:

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

3. Kompilieren Sie das zuvor definierte Projekt.
4. Kopieren Sie die Komponente in das Installationsverzeichnis für Kinesis Agent für Windows.
5. Erstellen oder aktualisieren`appsettings.json`Führen Sie die -Konfigurationsdatei durch, und platzieren Sie sie im Installationsverzeichnis für Kinesis Agent für Windows.
6. Kinesis Agent für Windows beenden und starten.
7. Überprüfen Sie die aktuelle Kinesis Agent für Windows-Protokolldatei (normalerweise im`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`Führen Sie das -Verzeichnis durch, um sicherzustellen, dass keine Probleme mit dem benutzerdefinierten Quell-Plug-In vorliegen.
8. Stellen Sie sicher, dass die Daten bei dem gewünschten AWS-Service eintreffen.

Ein Beispiel für die Erweiterung der`DirectorySource`Verwenden Sie die -Funktionalität, um die Analyse eines bestimmten Protokollformats zu implementieren, finden Sie unter`Amazon.KinesisTap.Uls\UlsSourceFactory.cs`und`Amazon.KinesisTap.Uls\UlsLogParser.cs`im Kinesis Agent für Windows-Quellcode.

Ein Beispiel für die Erstellung einer Quelle, die die Funktionalität zum Setzen von Lesezeichen bereitstellt, finden Sie unter`Amazon.KinesisTap.Windows\WindowsSourceFactory.cs`und`Amazon.KinesisTap.Windows\EventLogSource.cs`im Kinesis Agent für Windows-Quellcode.

Implementieren von Kinesis Agent für Windows Plugin Senks

Führen Sie die folgenden Schritt durch, um eine Kinesis g-In-Senke zu implementieren.

So erstellen Sie eine Kinesis Agent für Windows-Plugin-Senke

1. Fügen Sie zu dem zuvor definierten Projekt eine Klasse hinzu, mit der die `IEventSink`-Schnittstelle implementiert wird.

   Beispiel: Der folgende Code implementiert eine Senke, die nichts anderes macht, als den Eingang von Datensätzen zu protokollieren, die dann verworfen werden.

```
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.");
        }
    }
}
```

   In diesem Beispiel erbt die `NullSink`-Senken-Klasse von der `EventSink`-Klasse, da sie die Möglichkeit bietet, Datensätze in verschiedene Serialisierungsformate, wie z. B. JSON und XML, zu transformieren.

2. Kompilieren Sie das zuvor definierte Projekt.

3. Kopieren Sie die Komponente in das Installationsverzeichnis für Kinesis Agent für Windows.
4. Erstellen oder aktualisieren`appsettings.json`Führen Sie die -Konfigurationsdatei durch, die von der neuen Senke Gebrauch macht, und platzieren Sie sie im Installationsverzeichnis für Kinesis Agent für Windows. Angenommen, Sie möchten die benutzerdefinierten Plug-Ins `RandomSource` und `NullSink` nutzen. In diesem Fall könnten Sie die folgende `appsettings.json`-Konfigurationsdatei verwenden:

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

   Diese Konfiguration erstellt eine Quelle, die eine Instance von `RandomData` sendet, bei der `RandomValue` alle 10 Sekunden auf eine Zufallszahl zwischen 0 und 50 eingestellt wird. Die hiermit erstellte Senke transformiert die eingehenden `RandomData`-Instances in das JSON-Format, protokolliert das JSON-Format und verwirft die Instances. Achten Sie darauf, beide Factories, die Quell-Klasse `RandomSource` und die Senken-Klasse `NullSink`, im zuvor definierten Projekt einzuschließen, um die Beispiel-Konfigurationsdatei verwenden zu können.
5. Kinesis Agent für Windows beenden und starten.

6. Überprüfen Sie die aktuelle Kinesis Agent für Windows-Protokolldatei (normalerweise im`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`Führen Sie die folgenden Schritt durch, um sicherzustellen, dass es keine Probleme mit dem benutzerdefinierten Senken-Plug-In gibt.
7. Stellen Sie sicher, dass die Daten bei dem gewünschten AWS-Service eintreffen. Da das Beispiel `NullSink` nicht an einen AWS-Service gestreamt wird, können Sie die richtige Operation der Senke anhand von Protokollmeldungen überprüfen, aus denen hervorgeht, dass die Datensätze empfangen wurden.

   Beispielsweise kann die angezeigte Protokolldatei wie folgt aussehen:

```
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}.
```

Wenn Sie eine Senke erstellen, die auf AWS-Services zugreift, sind bestimmte Basisklassen möglicherweise hilfreich. Für eine Spüle, die die`AWSBufferedEventSink`Basisklasse finden Sie unter`Amazon.KinesisTap.AWS\CloudWatchLogsSink.cs`im Quellcode für Kinesis Agent für Windows.

Benutzerhandbuch für Amazon Kinesis Agent für Microsoft Windows

API-Version: 15.10.2018

In der folgenden Tabelle sind Änderungen amBenutzerhandbuch für Amazon Kinesis Agent für Microsoft Windows(dieses Dokument).

Update-Historie-Änderung	Update-Historie-Beschreibung	Update-Historie-Datum
Größeres Dokumentationsupdate	Zusätzliche Anweisungen für die MSI-Installation. Die DirectorySource-Konfiguration wurde aktualisiert und WindowSeventLogPollingSource hinzugefügt. Für die Konfiguration der Sink wurde eine lokale Dateisystem-Synchronisierungskonfiguration hinzugefügt; ProfileRefreshingAWScredentialProvider; Informationen zu Textdekorationen, Auflösen von Variablen in Senken-Attributen, Konfigurieren regionaler STS-Endpunkte für Senken, Konfigurieren von VPC Endpunkten und Konfigurieren alternativer Proxyserver. Für Pipes wurden Konfigurationsattribute hinzugefügt.	23. Februar 2021
Aktualisierung der Dokumentation	Aktualisierte Thematik, um anzuzeigen, dass bei den Spezifikationen des Amazon-	7. November 2018

	S3-Speicherorts zwischen Groß-	
Erste Veröffentlichung, Version 1.0.0.0.0.0.0.0.	Erste Version des Kinesis Agent für Windows-B enutzerhandbuchs.	5. November 2018

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar im AWS-Referenzhandbuch.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.