aws

Guide de l'utilisateur

Amazon Kinesis Agent pour les instances Microsoft Windows

Amazon Kinesis Agent pour les instances Microsoft Windows: Guide de l'utilisateur

Table of Contents

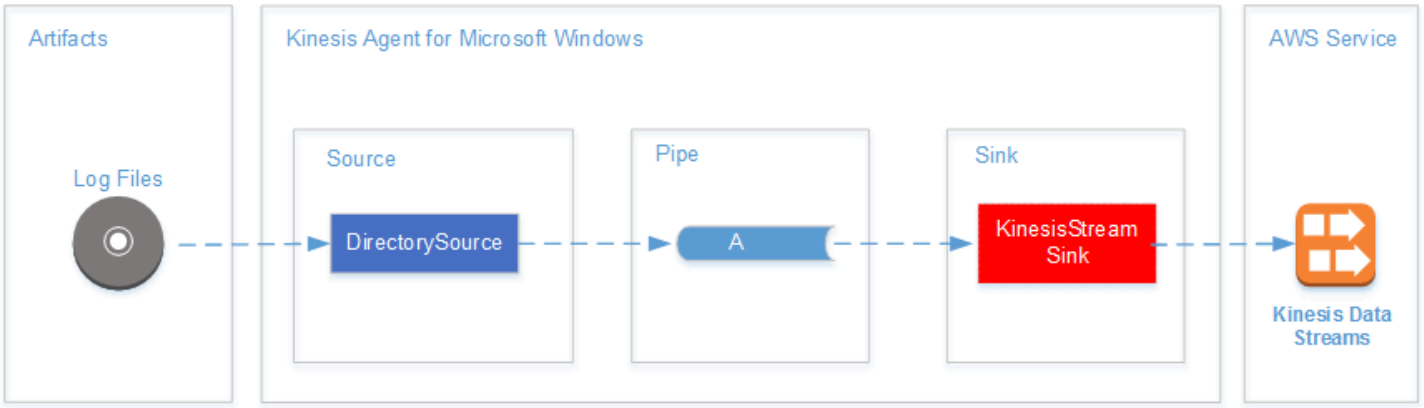
Présentation de Amazon Kinesis Agent pour Microsoft Windows ?

Amazon Kinesis Agent for Microsoft Windows (Kinesis Agent for Windows) est un agent configurable et extensible. Il s'exécute sur des flottes d'ordinateurs de bureau et de serveurs Windows, sur site ou dans le cloud AWS Cloud. Kinesis Agent for Windows collecte, analyse, transforme et diffuse de manière efficace et fiable des journaux, des événements et des métriques pour différents services AWS, notammentKinesis Data Streams,Kinesis Data Firehose,Amazon CloudWatch, etCloudWatch Logs.

À partir de ces services, vous pouvez ensuite stocker, analyser et visualiser les données à l'aide d'une grande variété d'autres services AWS, notamment :

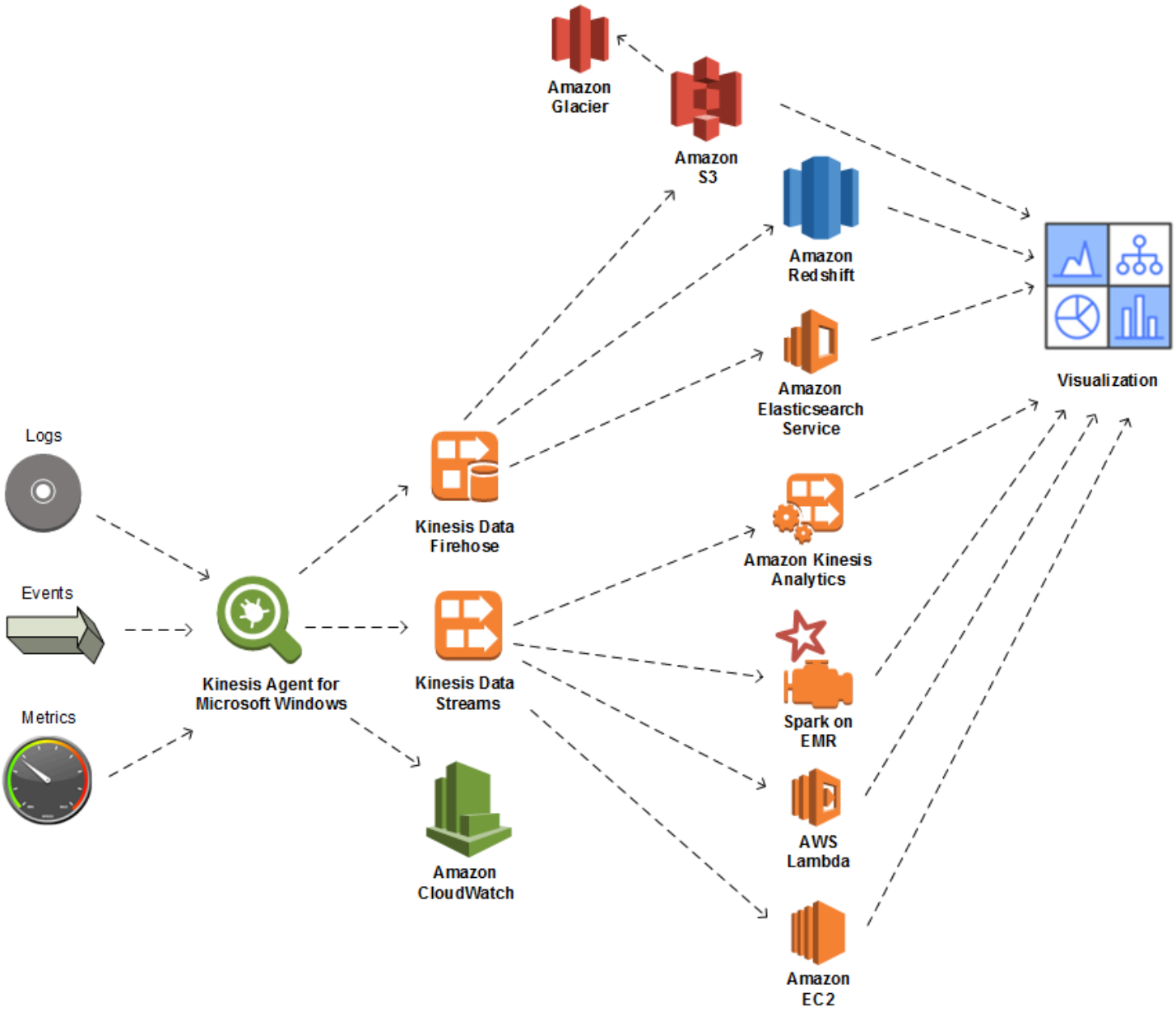
- Amazon Simple Storage Service (Amazon S3)
- Amazon Redshift
- Amazon Elasticsearch Service (Amazon ES)
- Kinesis Data Analytics
- Amazon QuickSight
- Amazon Athena
- Kibana

Le schéma suivant illustre une configuration simple de Kinesis Agent for Windows qui diffuse des fichiers journaux vers Kinesis Data Streams.



Pour plus d'informations sur les sources, les canaux et les récepteurs, consultez Amazon Kinesis Agent pour Microsoft Windows Concepts.

Le schéma suivant illustre certaines façons de créer des canaux de données personnalisés en temps réel à l'aide d'infrastructures de traitement de flux. Ces frameworks incluent Kinesis Data Analytics, Apache Spark on Amazon EMR et AWS Lambda.



Rubriques

À propos d'AWS

Amazon Web Services (AWS) est un ensemble de services d'infrastructure numériques que vous pouvez utiliser lors du développement d'applications. Les services incluent le calcul, le stockage, les base de données, l'analyse et la synchronisation de l'application (messagerie et file d'attente). AWS utilise un modèle de tarification à l'utilisation. Seuls les services que vous, ou vos applications, vous sont facturés. De plus, afin de rendre ses services plus accessibles pour le prototypage et l'expérimentation, AWS propose un niveau d'offre gratuite. Dans cette offre, les services sont gratuits en-dessous d'un certain niveau d'utilisation. Pour de plus amples informations sur les coûts AWS etOffre gratuite, consultezMise en route avec le Centre de ressources. Pour créer un compte AWS, ouvrez la page d'accueil AWS et inscrivez-vous.

Que pouvez-vous faire avec Kinesis Agent for Windows ?

Kinesis Agent pour Windows fournit les fonctions et capacités suivantes :

Collecte de données de journaux, d'événements et de métriques

Kinesis Agent for Windows collecte, analyse, transforme et diffuse des journaux, des événements et des métriques de flottes de serveurs et d'ordinateurs de bureau vers un ou plusieurs services AWS. Le format de la charge utile reçue par les services peut être différent de celui de la source d'origine. Par exemple, un journal peut être stocké dans un format de texte particulier (comme le format syslog) sur un serveur. Kinesis Agent for Windows peut collecter et analyser ce texte, et éventuellement le transformer au format JSON, par exemple, avant qu'il soit diffusé vers AWS. Cette opération facilite le traitement par certains services AWS qui utilisent le format JSON. Les données diffusées vers Kinesis Data Streams peuvent être traitées en continu par Kinesis Data Analytics afin de générer des métriques supplémentaires et des métriques agrégées, ce qui peut alimenter des tableaux de bord dynamiques. Vous pouvez stocker les données à l'aide de différents services AWS (comme Amazon S3), en fonction de la manière dont les données sont utilisées en aval dans un pipeline de données.

Intégration aux services AWS

Vous pouvez configurer Kinesis Agent for Windows afin qu'il envoie des fichiers journaux, des événements et des métriques vers plusieurs services AWS :

- Kinesis Data Firehose— Stockez facilement des données diffusées dans Amazon S3, Amazon Redshift, Amazon ES ouSplunkPour de plus amples analyses.
- Kinesis Data Streams— Traitez les données diffusées à l'aide d'applications personnalisées hébergées dans Kinesis Data Analytics ou Apache Spark surAmazon EMR. Ou utilisez du code personnalisé s'exécutant surAmazon EC2ou des fonctions sans serveur personnalisées s'exécutant dansAWS Lambda.
- CloudWatch— Affichez les métriques diffusées dans des graphiques, que vous pouvez combiner dans des tableaux de bord. Définissez ensuite des alarmes CloudWatch qui sont déclenchées par des valeurs de métrique qui ne respectent pas les seuils prédéfinis.
- CloudWatch Logs— Stockez les journaux et les événements diffusés, puis affichez-les et recherchez-les dans AWS Management Console, ou traitez-les plus en aval dans un pipeline de données.

Installation et configuration rapides

Vous pouvez installer et configurer Kinesis Agent for Windows en seulement quelques étapes. Pour plus d'informations, consultez Installation de Kinesis Agent pour Windows et Configuration d'Amazon Kinesis Agent pour Microsoft Windows. Un fichier de configuration déclarative simple spécifie les éléments suivants :

- Les sources et les formats des journaux, des événements et des métriques à collecter.
- Les transformations à appliquer aux données collectées. D'autres données peuvent être incluses, et les données existantes peuvent être transformées et filtrées.
- Les destinations où sont envoyées les données finales, ainsi que la mise en mémoire tampon, le partitionnement et le format pour les charges utiles de diffusion.

Kinesis Agent for Windows est fourni avec des analyseurs intégrés pour les fichiers journaux générés par les services d'entreprise Microsoft courants, tels que :

- Microsoft Exchange
- SharePoint
- Contrôleurs de domaine Active Directory
- Serveurs DHCP

Aucune administration continue

Kinesis Agent for Windows s'adapte automatiquement aux différentes situations sans perdre aucune donnée. C'est le cas notamment de la rotation des journaux, de la récupération après le redémarrage et des interruptions temporaires de réseau ou de service. Vous pouvez configurer Kinesis Agent for Windows afin qu'il soit automatiquement mis à jour vers de nouvelles versions. Aucune intervention de l'opérateur n'est requise dans ces situations.

Extension à l'aide d'une architecture ouverte

Si les capacités déclaratives et les plug-ins intégrés ne suffisent pas pour surveiller les systèmes d'ordinateurs de bureau ou de serveurs, vous pouvez étendre Kinesis Agent for Windows en créant des plug-ins. Les nouveaux plug-ins activent de nouvelles sources et destinations pour les journaux, les événements et les métriques. Le code source pour Kinesis Agent for Windows est disponible à l'adressehttps://github.com/awslabs/kinesis-agent-windows.

Benefits

Kinesis Agent for Windows effectue la collecte des données initiales, la transformation et la diffusion des journaux, des événements et des métriques pour les pipelines de données. La génération de ces pipelines de données présente de nombreux avantages :

Analyse et visualisation

L'intégration de Kinesis Agent for Windows avec Kinesis Data Firehose et ses capacités de transformation facilitent l'intégration avec différents services d'analyse et de visualisation :

- Amazon QuickSight— Service d'informatique décisionnelle basé sur le cloud et capable d'intégrer de nombreuses sources différentes. Kinesis Agent pour Windows peut transformer des données et les diffuser sur Amazon S3 et Amazon Redshift via Kinesis Data Firehose. Ce processus permet de découvrir des informations précises grâce aux données à l'aide des visualisations Amazon QuickSight.
- athena— Service de requête interactif qui active l'interrogation de données basée sur SQL. Kinesis Agent pour Windows peut transformer et diffuser des données vers Amazon S3 via Kinesis Data Firehose. athena peut ensuite exécuter de manière interactive des requêtes SQL sur ces données afin d'inspecter et d'analyser rapidement des journaux et des événements.
- Kibana— Outil de visualisation de données open source ; Kinesis Agent pour Windows peut transformer et diffuser des données vers Amazon ES via Kinesis Data Firehose. Vous pouvez ensuite utiliser Kibana pour explorer ces données. Créez et ouvrez différentes visualisations, y compris des histogrammes, des graphiques linéaires, des diagrammes à secteurs, des tableaux heat-map et des graphiques géospatiaux.

Security

Un pipeline d'analyse des données de journaux et d'événements qui inclut Kinesis Agent for Windows peut détecter les failles de sécurité dans les entreprises et envoyer des alertes en cas de faille, ce qui peut vous aider à bloquer ou arrêter les attaques.

Performances des applications

Kinesis Agent for Windows peut collecter des données de journaux, d'événements et de métriques relatives aux performances de l'application ou du service. Un pipeline de données complet peut ensuite analyser ces données. Cette analyse vous aide à améliorer les performances et la fiabilité des applications et des services grâce à la détection et au signalement des défauts qui, sans elle, pourraient passer inaperçus. Par exemple, vous pouvez détecter les modifications importantes au niveau des heures d'exécution des appels d'API de service. Lorsque cette fonctionnalité est liée à un déploiement, elle vous aide à localiser et à résoudre les nouveaux problèmes de performances avec les nouveaux services en votre possession.

Opérations de service

Un pipeline de données peut analyser les données collectées pour prédire les problèmes opérationnels potentiels et apporter des informations sur la façon d'éviter les interruptions de service. Par exemple, vous pouvez analyser des journaux, des événements et des métriques pour déterminer l'utilisation des capacités actuelle et prévue, afin de pouvoir mettre en service une capacité supplémentaire avant que les utilisateurs de service soient affectés. En cas d'interruption de service, vous pouvez analyser les données afin de déterminer l'impact sur les clients pendant la panne.

Auditing

Un pipeline de données peut traiter les journaux, les événements et les métriques collectés et transformés par Kinesis Agent for Windows. Vous pouvez ensuite procéder à l'audit de ces données traitées à l'aide de différents services AWS. Par exemple, Kinesis Data Firehose peut recevoir un flux de données de Kinesis Agent for Windows, qui stocke les données dans Amazon S3. Vous pouvez ensuite contrôler ces données en exécutant des requêtes SQL interactives à l'aide de Athena.

Archiving

Souvent, les données opérationnelles les plus importantes sont les données récemment collectées. Toutefois, l'analyse de données d'applications et de services collectées sur plusieurs années peut également être utile, pour la planification à long terme par exemple. La conservation de grandes quantités de données peut s'avérer coûteuse. Kinesis Agent pour Windows peut collecter, transformer et stocker des données dans Amazon S3 via Kinesis Data Firehose. Par conséquent,Amazon S3 Glacierpermet de réduire les coûts de l'archivage des données plus anciennes.

Alerting

Kinesis Agent pour Windows transmet les mesures vers CloudWatch. À votre tour, vous pouvez créer des alarmes CloudWatch pour envoyer une notification viaAmazon Simple Notification Service (Amazon SNS)lorsqu'une mesure enfreint systématiquement un seuil spécifique. Les ingénieurs disposent ainsi d'une meilleure connaissance des problèmes opérationnels avec leurs applications et services.

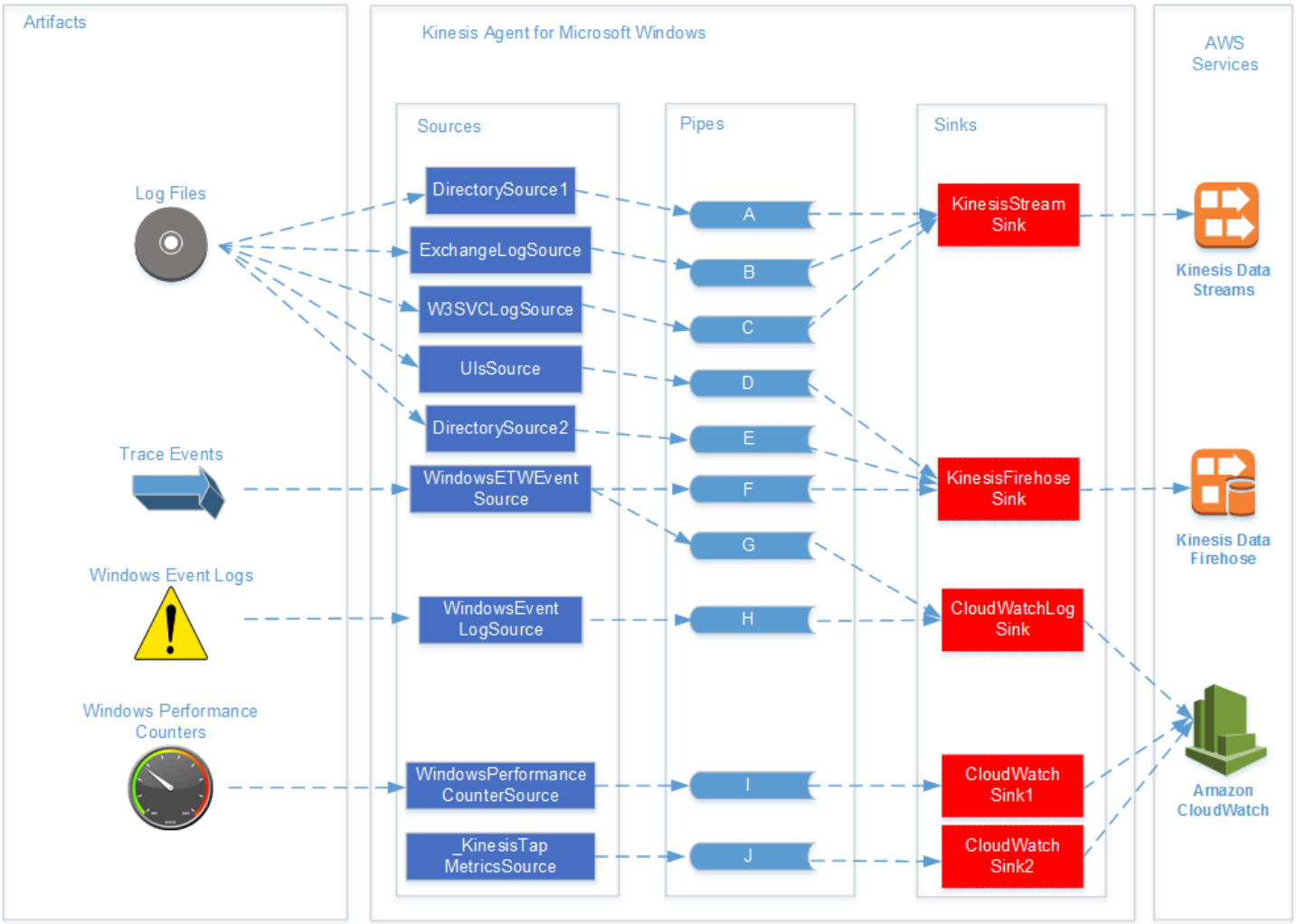
Mise en route avec Kinesis Agent pour Windows

Pour en savoir plus sur Kinesis Agent for Windows, nous vous recommandons de commencer par les sections suivantes :

- Amazon Kinesis Agent pour Microsoft Windows Concepts
- Démarrez avec l'agent Amazon Kinesis

Amazon Kinesis Agent pour Microsoft Windows Concepts

Si vous comprenez les concepts clés d'Amazon Kinesis Agent for Microsoft Windows (Kinesis Agent for Windows) vous pourrez plus facilement collecter des données sur les flottes d'ordinateurs de bureau et de serveurs et les diffuser vers le reste du pipeline de données afin qu'elles soient traitées.



Ce diagramme de pipeline de données illustre les composants et processus suivants :

Les serveurs et les bureaux ont des artefacts tels que les fichiers journaux, les événements et les mesures qui sont rassemblés par un ou plusieurs Kinesis Agent pour Windowssources. Les données peuvent éventuellement être transformées, par exemple, d'un format texte de fichier plat en objet.

Les données (sous forme d'objet ou de texte) peuvent ensuite circuler dans un ou plusieurs Kinesis Agent pour WindowsCanaux. Un canal connecte une source à un agent Kinesis pour Windowssink. Le canal peut éventuellement filtrer les données inutiles.

Un récepteur peut éventuellement transformer les données analysées en objets au format JSON ou XML. Le récepteur envoie les données vers un service AWS spécifique, tel que Kinesis Data Streams, Kinesis Data Firehose ou Amazon CloudWatch.

En utilisant plusieurs canaux, une seule source peut envoyer les mêmes données vers plusieurs récepteurs (observez par exemple les canaux F et G dans le diagramme). En utilisant plusieurs canaux, plusieurs sources peuvent diffuser des données vers un seul récepteur (observez par exemple les canaux A, B et C dans le diagramme). Il est également possible d'utiliser plusieurs canaux pour diffuser des données à partir de plusieurs récepteurs vers plusieurs sources. Les sources, les récepteurs et les canaux ont des types, et plusieurs sources, récepteurs ou canaux peuvent être du même type.

Pour obtenir des exemples de fichiers de configuration déclarant des sources, des récepteurs et des canaux, consultez Exemples de configuration de l'agent Kinesis pour Windows.

Rubriques

Pipelines de données

AData pipelineest utilisé pour collecter, traiter, visualiser et éventuellement générer des alarmes pour les applications et les services. Kinesis Agent for Windows s'intègre aux pipelines de données au début, où les journaux, les événements et les mesures sont rassemblés à partir de flottes d'ordinateurs de bureau ou de serveurs. Kinesis Agent for Windows transmet les données collectées aux différents services AWS qui forment le reste du pipeline de données. Un pipeline de données a un objectif, par exemple celui de visualiser l'état d'un service spécifique en temps réel afin d'aider les ingénieurs à exploiter plus efficacement ce service. Un pipeline de données de l'état d'un service peut effectuer les opérations suivantes :

- Signaler des problèmes aux ingénieurs avant qu'ils n'affectent l'expérience des clients des services.
- Aider les ingénieurs à gérer efficacement le coût du service en fournissant les tendances de l'utilisation des ressources. Ces tendances leur permettent d'ajuster de façon appropriée

les niveaux des ressources, ou même de mettre en œuvre des scénarios de mise à l'échelle automatique.

- Éclairer la cause première des problèmes qui sont signalés par les clients du service. Cela permet de résoudre plus rapidement ces problèmes et réduit les coûts de support.

Pour obtenir un exemple détaillé de la création d'un pipeline de données à l'aide de Kinesis Agent pour Windows, consultezDidacticiel : Diffuser les fichiers journaux JSON vers Amazon S3 à l'aide de Kinesis Agent pour Windows.

Sources

Agent Kinesis pour Windowssourcerassemble des journaux, des événements ou des mesures. Une source collecte un type spécifique de données à partir d'un producteur spécifique de ces données, en fonction du type de la source. Par exemple, le type `DirectorySource` collecte les fichiers journaux de répertoires spécifiques dans le système de fichiers. Si les données ne sont pas déjà structurées (comme dans le cas de certains types de fichiers journaux), une source peut être utile pour l'analyse de la représentation textuelle dans une certaine forme structurée. Chaque source correspond à unDéclaration de sourcedans Kinesis Agent pour Windows`appsettings.json`Fichier de configuration de. La déclaration de source fournit des détails essentiels pour configurer la source afin de l'adapter en fonction des exigences spécifiques de collecte des données. Les types de détails qui peuvent être configurés varient en fonction du type de source. Par exemple, le type de source `DirectorySource` nécessite de spécifier le répertoire dans lequel se trouvent les fichiers journaux.

Pour plus d'informations sur les types de source et les déclarations de source, consultez Déclarations de sources.

Sinks

Agent Kinesis pour Windowssinkprend les données collectées par une source Kinesis Agent pour Windows et les transmet à l'un des nombreux services AWS possibles qui forment le reste du pipeline de données. Chaque récepteur correspond à unDéclaration de récepteurdans Kinesis Agent pour Windows`appsettings.json`Fichier de configuration de. La déclaration de récepteur fournit des détails essentiels pour configurer le récepteur afin de l'adapter en fonction des exigences spécifiques de diffusion des données. Les types de détails qui peuvent être configurés varient en fonction du type de récepteur. Par exemple, certains types de récepteurs autorisent une déclaration de récepteur pour spécifier une sérialisation `Format` spécifique pour les données fournies. Lorsque

cette option est spécifiée dans la déclaration de récepteur, la sérialisation des données collectées a lieu avant la diffusion des données vers le service AWS associé au récepteur.

Pour plus d'informations sur les types de récepteur et les déclarations de récepteur, consultez Déclarations de récepteurs.

Pipes

Agent Kinesis pour WindowsCanauxconnecte la sortie d'une source Kinesis Agent pour Windows à l'entrée d'un récepteur Kinesis Agent pour Windows. Il peut éventuellement transformer les données lors de leur passage dans le canal. Chaque canal correspond à une déclaration de canal spécifique dans l'Agent de Kinesis pour Windows`appsettings.json`Fichier de configuration de. La déclaration de canal fournit des détails essentiels pour configurer le récepteur, comme la source et le récepteur du canal.

Pour plus d'informations sur les types de canal et les déclarations de canal, consultez Déclarations de canal.

Démarrez avec l'agent Amazon Kinesis

Vous pouvez utiliser Amazon Kinesis Agent for Microsoft Windows (Kinesis Agent for Windows) pour collecter, analyser, transformer et diffuser des journaux, événements et métriques de votre flotte Windows et les diffuser vers différents services AWS. Les informations suivantes contiennent des prérequis et les instructions étape par étape pour installer et configurer Kinesis Agent pour Windows.

Rubriques

Prerequisites

Avant d'installer Kinesis Agent pour Windows, veillez à disposer des prérequis suivants :

- Connaissance des concepts Kinesis Agent for Windows. Pour plus d'informations, consultez Amazon Kinesis Agent pour Microsoft Windows Concepts.
- Un compte AWS permettant d'utiliser les différents services AWS liés à votre pipeline de données. Pour plus d'informations sur la création et la configuration d'un compte AWS, consultez Configuration d'un compte AWS.
- Microsoft .NET Framework 4.6 ou une version ultérieure sur chaque ordinateur de bureau ou serveur qui exécutera Kinesis Agent pour Windows. Pour plus d'informations, consultez Install the .NET Framework for developers dans la documentation Microsoft .NET.

  Pour déterminer la version la plus récente de .NET Framework installée sur un ordinateur de bureau ou un serveur, utilisez le script PowerShell suivant :

```
[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
```

- Vous disposez des flux qui vous permettront d'envoyer des données à partir de Kinesis Agent for Windows (si vous utilisez Amazon Kinesis Data Streams). Créez les flux à l'aide de l'outilConsole Kinesis Data Streams, leAWS CLI, ouOutils AWS pour Windows PowerShell. Pour de plus amples informations, veuillez consulterCréation et mise à jour des flux de données dans leAmazon Kinesis Data Streams.
- Vous disposez des flux de diffusion Firehose qui vous permettront d'envoyer des données à partir de Kinesis Agent pour Windows (si vous utilisez Amazon Kinesis Data Firehose). Créez des flux de livraison à l'aide duKinesis Data Firehose, leAWS CLI, ouOutils AWS pour Windows PowerShell. Pour plus d'informations, consultez Création d'un flux de diffusion Amazon Kinesis Data Firehose dans le Guide du développeur Amazon Kinesis Data Firehose.

Configuration d'un compte AWS

Si vous n'avez pas de compte AWS, complétez les étapes suivantes pour en créer un.

Pour créer un compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

Pour créer un administrateur pour vous-même et ajouter l'utilisateur à un groupe d'administrateurs (console)

1. Connectez-vous à la IAM console (Console IAM) en tant que propriétaire du compte en choisissant Root user (Utilisateur racine) et en entrant l'adresse e-mail de votre compte AWS. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > Nous vous recommandons vivement de respecter la bonne pratique qui consiste à avoir recours à**`Administrator`**Utilisateur IAM qui suit et verrouille en sécurité les informations d'identification de l'utilisateur racine. Connectez-vous en tant qu'utilisateur racine pour effectuer certaines tâches de gestion des comptes et des services.

2. Dans le panneau de navigation, choisissez Utilisateurs, puis Add user (Ajouter un utilisateur).
3. Dans User name (Nom d'utilisateur), entrez **`Administrator`**.
4. Activez la case à cocher en regard de l'accès à AWS Management Console. Puis, sélectionnez Mot de passe personnalisé, et saisissez votre nouveau mot de passe dans la zone de texte.
5. Par défaut, AWS oblige le nouvel utilisateur à créer un nouveau mot de passe lors de sa première connexion. Décochez la case en regard de User must create a new password at next sign-in (L'utilisateur doit créer un nouveau mot de passe à sa prochaine connexion) pour autoriser le nouvel utilisateur à réinitialiser son mot de passe une fois qu'il s'est connecté.
6. ChoisissezSuivant: Autorisations.
7. Sous Set permissions (Accorder des autorisations), choisissez Add user to group (Ajouter un utilisateur au groupe).
8. Choisissez Create group.
9. Dans la boîte de dialogue Create group (Créer un groupe), pour Group name (Nom du groupe), tapez **`Administrators`**.
10. ChoisissezStratégies de filtre, puisGestion d'AWS - fonction de travailpour filtrer le contenu de la table.
11. Dans la liste des stratégies, cochez la case AdministratorAccess. Choisissez ensuite Create group.

    > **Note**
    >
    > Vous devez activer l'accès de l'utilisateur et du rôle IAM à la facturation avant de pouvoir utiliser les autorisations `AdministratorAccess` pour accéder à la console AWS Billing and Cost Management. Pour ce faire, suivez les instructions de l'étape 1 du didacticiel portant sur comment déléguer l'accès à la console de facturation.

12. De retour dans la liste des groupes, activez la case à cocher du nouveau groupe. Choisissez Refresh si nécessaire pour afficher le groupe dans la liste.
13. ChoisissezSuivant: Tags (Balises).
14. (Facultatif) Ajoutez des métadonnées à l'utilisateur en associant les balises sous forme de paires clé-valeur. Pour de plus amples informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des utilisateurs et des rôles IAM dans le Guide de l'utilisateur IAM.
15. ChoisissezSuivant: VérificationPour afficher la liste des membres du groupe à ajouter au nouvel utilisateur. Une fois que vous êtes prêt à continuer, choisissez Create user.

Vous pouvez utiliser ce même processus pour créer d'autres groupes et utilisateurs et pour accorder l'accès aux ressources de votre compte AWS à vos utilisateurs. Pour en savoir plus sur l'utilisation des stratégies permettant de limiter les autorisations d'accès des utilisateurs à certaines ressources AWS, veuillez consulter Gestion des accès et Exemples de stratégies.

Pour vous inscrire sur AWS et créer un compte d'administrateur

1. Si vous n'avez pas de comptehttps://aws.amazon.com/. Choisissez Créer un compte AWS, puis suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code PIN en utilisant le clavier numérique du téléphone.
2. Connectez-vous à AWS Management Console et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
3. Dans le volet de navigation, choisissez Groups, puis Create New Group.
4. Dans Nom du groupe, entrez un nom pour le groupe, par exemple **`Administrators`**, puis choisissez Étape suivante.
5. Dans la liste des stratégies, sélectionnez la case à cocher en regard de la stratégie AdministratorAccess. Vous pouvez utiliser le menu Filter et la zone Search pour filtrer la liste de stratégies.
6. Choisissez Next Step. Choisissez Créer un groupe. Votre nouveau groupe apparaît ensuite sous Nom du groupe.
7. Dans le volet de navigation, choisissez Users, puis Create New Users.
8. Dans la zone 1, saisissez un nom d'utilisateur, décochez la case en regard de Générez une clé d'accès pour chaque utilisateur, puis choisissez Créer.
9. Dans la liste des utilisateurs, sélectionnez le nom (et non pas la case à cocher) de l'utilisateur que vous venez de créer. Vous pouvez utiliser la zone Recherche pour rechercher le nom utilisateur.
10. Choisissez l'onglet Groupes, puis sélectionnez Ajouter un utilisateur aux groupes.
11. Cochez la case en regard du groupe d'administrateurs, puis choisissez Ajouter aux groupes.
12. Choisissez l'onglet Informations d'identification de sécurité. Sous Sign-In Credentials, choisissez Manage Password.
13. Sélectionnez Affecter un mot de passe personnalisé, saisissez un mot de passe dans les zones Mot de passe et Confirmer le mot de passe, puis choisissez Appliquer.

Installation de Kinesis Agent pour Windows

Vous pouvez installer Kinesis Agent pour Windows sous Windows trois façons :

- Installez à l'aide de MSI (package d'installation Windows).
- Installez àAWS Systems Manager, un ensemble de services permettant d'administrer les serveurs et les ordinateurs de bureau.
- Exécuter un script PowerShell.

> **Note**
>
> Les instructions suivantes utilisent occasionnellement les termes KinesisTap et AWSKinesisTap. Ces termes ont la même signification que Kinesis Agent pour Windows, mais vous devez les spécifier tels quels lorsque vous exécutez ces instructions.

Installer Kinesis Agent pour Windows à l'aide de MSI

Vous pouvez télécharger le dernier package Kinesis Agent pour Windows MSI à partir duréférentiel kinesis-agent-windows sur GitHub. Après avoir téléchargé le MSI, utilisez Windows pour le lancer et suivez les invites du programme d'installation. Après l'installation, vous pouvez désinstaller comme n'importe quelle application Windows.

Vous pouvez également utiliser le kitmsiexecÀ partir de l'invite de commande Windows pour installer en mode silencieux, activer la journalisation et désinstaller, tel qu'illustré dans les exemples suivants. Remplacez*AWSKinesisTap.1.1.216.4.msi* `with the appropriate version of Kinesis Agent for Windows for your application.`

Pour installer Kinesis Agent pour Windows en mode silencieux :

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

Pour consigner les messages d'installation à des fins de dépannage dans un fichier nommé***logfile.log*** :

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

Pour désinstaller Kinesis Agent pour Windows à l'aide de l'invite de commandes :

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

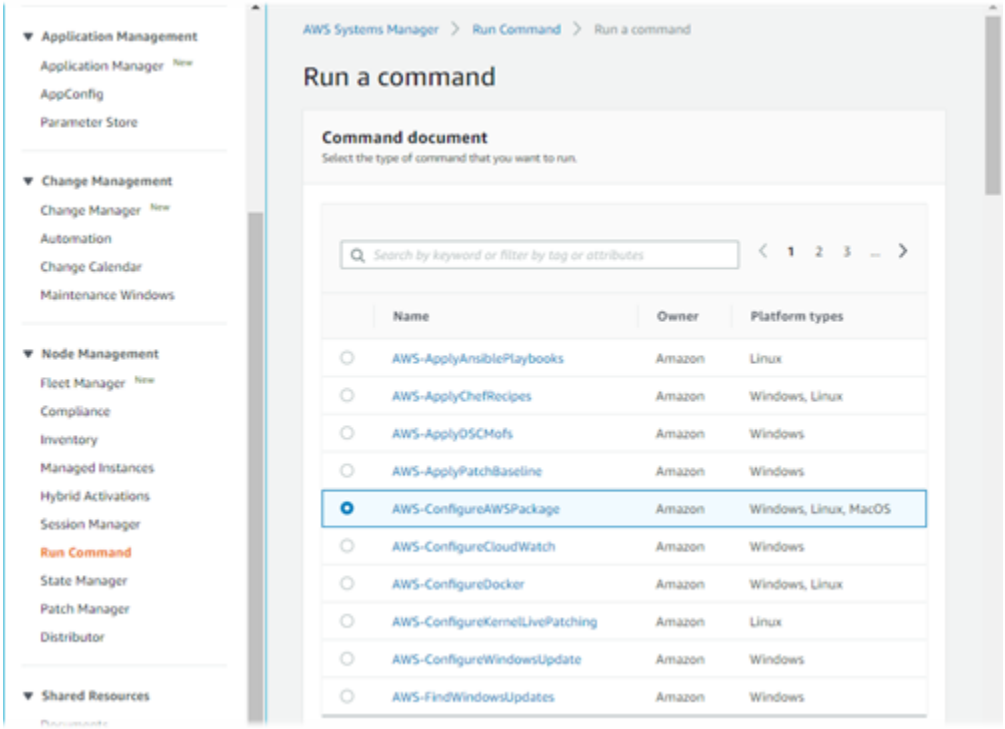
Installer Kinesis Agent pour Windows à l'aide d'AWS Systems Manager

Suivez ces étapes pour installer Kinesis Agent pour Windows à l'aide de la fonctionnalité Exécuter la commande Pour en savoir plus sur la fonctionnalité Exécuter la commande, consultezAWS Systems Managerdans leAWS Systems Manager. En plus d'utiliser la commande d'exécution de Systems Manager, vous pouvez également utiliser Systems ManagerFenêtres de maintenanceandState Managerpour automatiser le déploiement de Kinesis Agent pour Windows au fil du temps.

> **Note**
>
> L'installation de Systems Manager pour Kinesis Agent for Windows est disponible dans les régionsAWS Systems ManagerÀ l'exception des éléments suivants :
>
> - cn-north-1
> - cn-northwest-1
> - Toutes les régions AWS GovCloud

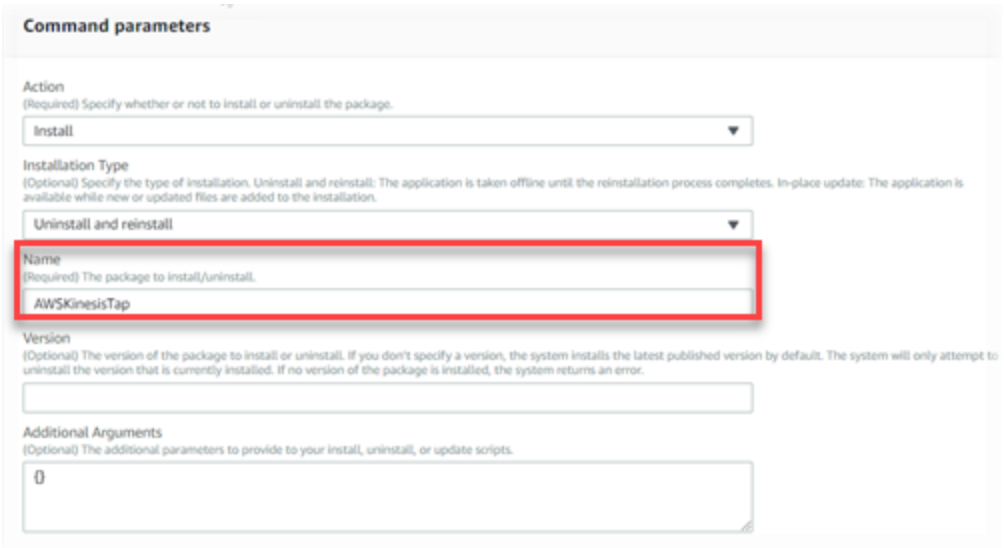
Pour installer Kinesis Agent pour Windows à l'aide du Systems Manager

1. Vérifiez que la version 2.58.0 ou une version ultérieure de l'agent SSM est installée sur les instances dans lesquelles vous souhaitez installer Kinesis Agent pour Windows. Pour de plus amples informations, veuillez consulterInstallation et configuration de l'agent SSM sur les instances Windowsdans leAWS Systems Manager.
2. Ouvrez la console AWS Systems Manager à l'adressehttps://console.aws.amazon.com/systems-manager/.
3. Dans le volet de navigation,Gestion des nœuds, choisissezFonctionnalité Exécuter la commande, puisFonctionnalité Exécuter la commande.
4. À partir duDocument de commande, sélectionnezAWS-ConfigureAWSPackagedocument.



5. UNDERParamètres de commande, pourNom, saisissezAWSKineSISTAP. Conservez les valeurs par défaut des autres paramètres.

> **Note**
>
> quitterVersionPour spécifier la dernière version du package AWSKineSistap. Éventuellement, vous pouvez saisir une version spécifique à installer.



6. UNDERCibles, spécifiez les instances sur lesquelles exécuter la commande. Vous pouvez choisir de spécifier des instances en fonction des balises associées aux instances, vous pouvez choisir des instances manuellement ou vous pouvez spécifier un groupe de ressources qui inclut des instances.
7. Conservez les valeurs par défaut de tous les autres paramètres et choisissezRun (Exécuter Lambda).

Installer Kinesis Agent pour Windows à l'aide de PowerShell

Utilisez un éditeur de texte pour copier les commandes suivantes dans un fichier et l'enregistrer en tant que script PowerShell. Nous utilisons`InstallKinesisAgent.ps1`Dans l'exemple suivant.

```
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
}
```

Ouvrez une fenêtre d'invite de commande de niveau élevé. Dans le répertoire où le fichier a été téléchargé, utilisez la commande suivante pour exécuter le script :

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

Pour installer une version spécifique de Kinesis Agent pour Windows, ajoutez le kit-`version`Option :

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

Remplacez*`version`*avec un numéro de version de Kinesis Agent pour Windows valide. Pour plus d'informations sur la version, consultezréférentiel kinesis-agent-windows sur GitHub.

Il existe de nombreux outils de déploiement capables d'exécuter des scripts PowerShell à distance. Ils peuvent être utilisés pour automatiser l'installation de Kinesis Agent for Windows sur des flottes de serveurs ou des ordinateurs de bureau.

Configuration et démarrage de Kinesis Agent pour Windows

Après l'installation de Kinesis Agent pour Windows, vous devez configurer et démarrer l'agent. Après cette opération, aucune autre intervention ne devrait être nécessaire.

Pour configurer et démarrer Kinesis Agent

1. Créez et déployez un fichier de configuration Kinesis Agent for Windows. Ce fichier configure les sources, les récepteurs et les canaux, ainsi que d'autres éléments de la configuration globale.

   Pour en savoir plus sur la configuration de Kinesis Agent for Windows, consultezConfiguration d'Amazon Kinesis Agent pour Microsoft Windows.

   Pour obtenir des exemples complets de fichiers de configuration à personnaliser et installer, consultez Exemples de configuration de l'agent Kinesis pour Windows.
2. Ouvrez une fenêtre d'invite de commande PowerShell avec élévation de privilèges et démarrez Kinesis Agent pour Windows à l'aide de la commande PowerShell suivante :

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

Configuration d'Amazon Kinesis Agent pour Microsoft Windows

Avant de démarrer Amazon Kinesis Agent pour Microsoft Windows, vous devez créer un fichier de configuration et le déployer. Le fichier de configuration fournit les informations nécessaires pour collecter, transformer et diffuser les données entre les serveurs et les ordinateurs de bureau Windows et les différents services AWS. Les fichiers de configuration définissent les ensembles de sources, récepteurs et canaux qui connectent les sources aux récepteurs, ainsi que les transformations facultatives.

Le fichier de configuration de l'Agent Kinesis pour Windows est nomméappsettings.json. Déployez ce fichier sur %PROGRAMFILES%\Amazon\AWSKinesisTap.

Rubriques

Structure de la configuration de base

La structure de base du fichier de configuration de Amazon Kinesis Agent for Microsoft Windows est un document JSON suivant :

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

- La valeur de Sources est une ou plusieurs Déclarations de sources.

- La valeur de `Sinks` est une ou plusieurs Déclarations de récepteurs.
- La valeur de `Pipes` est une ou plusieurs Déclarations de canal.

Pour plus d'informations sur les concepts de source, canal (pipe) et récepteur (sink) de Kinesis Agent (sink), consultezAmazon Kinesis Agent pour Microsoft Windows Concepts.

L'exemple suivant est un`appsettings.json`Configurez Kinesis Agent pour Windows pour diffuser les événements des journaux d'application Windows sur Kinesis Data Firehose.

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

Pour plus d'informations sur chaque type de déclaration, consultez les sections suivantes :

- Déclarations de sources
- Déclarations de récepteurs
- Déclarations de canal

Sensibilité à la casse de la configuration

Les fichiers au format JSON sont généralement sensibles à la casse et vous devez supposer que toutes les clés et valeurs des fichiers de configuration de Kinesis Agent for Windows le sont également. Certaines clés et valeurs du fichier de configuration `appsettings.json` ne sont pas sensibles à la casse. Par exemple :

- Valeur de la paire clé-valeur `Format` pour les récepteurs. Pour plus d'informations, consultez Déclarations de récepteurs.
- Valeur de la paire clé-valeur `SourceType` pour les sources, de la paire clé-valeur `SinkType` pour les récepteurs et de la paire clé-valeur `Type` pour les canaux et les plug-ins.
- Valeur de la paire clé-valeur `RecordParser` pour la source `DirectorySource`. Pour plus d'informations, consultez Configuration de DirectorySource.
- Valeur de la paire clé-valeur `InitialPosition` pour les sources. Pour plus d'informations, consultez Configuration des signets.
- Préfixes pour les substitutions de variables. Pour plus d'informations, consultez Configuration des substitutions de variables de récepteur.

Déclarations de sources

Dans Amazon Kinesis Agent pour Microsoft Windows,Déclarations de sourcesDécrire l'emplacement et la nature des données de journaux, d'événements et de métriques qui doivent être collectées. Elles spécifient également, le cas échéant, des informations destinées à l'analyse des données afin que ces dernières puissent être transformées. Les sections suivantes décrivent les configurations des types de sources intégrés qui sont disponibles dans Kinesis Agent pour Windows. Étant donné que l'Agent Kinesis pour Windows est extensible, vous pouvez ajouter des types de sources personnalisés. Chaque type de source nécessite généralement des paires clé-valeur spécifiques dans les objets de configuration qui sont pertinents pour ce type de source.

Toutes les déclarations de sources doivent contenir au moins les paires clé-valeur suivantes :

`Id`

Chaîne unique qui identifie un objet source particulier dans le fichier de configuration.

`SourceType`

Nom du type de source pour cet objet source. Le type de source spécifie l'origine des données de journal, d'événement ou de métrique qui sont collectées par cet objet source. Il détermine également les autres aspects de la source pouvant être déclarés.

Pour obtenir des exemples de fichiers de configuration complets utilisant différents types de déclarations de sources, consultez Diffusion à partir de diverses sources vers les Kinesis Data Streams.

Rubriques

Configuration de DirectorySource

Overview

Le type de source `DirectorySource` recueille les journaux des fichiers qui sont stockés dans le répertoire spécifié. Étant donné que les fichiers journaux ont de nombreux formats différents, la déclaration `DirectorySource` vous permet de spécifier le format des données dans le fichier journal. Vous pouvez ensuite transformer le contenu du journal dans un format standard tel que JSON ou XML avant de le diffuser vers différents services AWS.

Voici un exemple de déclaration `DirectorySource` :

```
{
    "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"
}
```

Toutes les déclarations `DirectorySource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"DirectorySource"` (obligatoire).

`Directory`

Chemin du répertoire contenant les fichiers journaux (obligatoire).

`FileNameFilter`

Limite le cas échéant l'ensemble de fichiers dans le répertoire dans lequel les données de journal sont collectées sur la base d'un modèle d'attribution de noms de fichiers contenant des caractères génériques. Si vous disposez de plusieurs modèles de nom de fichier journal, cette fonctionnalité vous permet d'utiliser un`DirectorySource`, comme illustré dans l'exemple suivant.

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

Les administrateurs système compressent parfois les fichiers journaux avant de les archiver. Si vous spécifiez`"*.*"`in`FileNameFilter`, les fichiers compressés connus sont désormais exclus. Cette fonctionnalité empêche`.zip`,`.gz`, et`.bz2`d'être diffusés accidentellement. Si cette paire clé-valeur n'est pas spécifiée, les données de tous les fichiers du répertoire sont collectées par défaut.

`IncludeSubdirectories`

Spécifie de surveiller les sous-répertoires à une profondeur arbitraire limitée par le système d'exploitation. Cette fonctionnalité est utile pour surveiller les serveurs Web avec plusieurs sites Web. Vous pouvez également utiliser l'`IncludeDirectoryFilter`pour surveiller uniquement certains sous-répertoires spécifiés dans le filtre.

`RecordParser`

Spécifie la façon dont le type de source `DirectorySource` doit analyser les fichiers journaux qui se trouvent dans le répertoire spécifié. Cette paire clé-valeur est obligatoire et les valeurs valides sont les suivantes :

- `SingleLine`— Chaque ligne du fichier journal est un enregistrement de journal.
- `SingleLineJson`— Chaque ligne du fichier journal est un enregistrement de journal au format JSON. Cet analyseur est utile lorsque vous souhaitez ajouter des paires clé-valeur supplémentaires au JSON à l'aide d'un élément ObjectDecoration. Pour plus d'informations, consultez Configuration des décorations de récepteurs. Pour obtenir un exemple utilisant l'analyseur d'enregistrements `SingleLineJson`, consultez Didacticiel : Diffuser les fichiers journaux JSON vers Amazon S3 à l'aide de Kinesis Agent pour Windows.
- `Timestamp`— Une ou plusieurs lignes peuvent inclure un enregistrement de journal. L'enregistrement de journal commence par un horodatage. Cette option requiert la spécification de la paire clé-valeur `TimestampFormat`.
- `Regex`— Chaque enregistrement commence par du texte qui correspond à une expression régulière particulière. Cette option requiert la spécification de la paire clé-valeur `Pattern`.
- `SysLog`— Indique que que le fichier journal est écrit dans l'syslogformat standard. Le fichier journal est analysé dans les enregistrements en fonction de cette spécification.
- `Delimited`— Version plus simple de l'analyseur d'enregistrements Regex dans laquelle les éléments de données des enregistrements de journaux sont séparés par un délimiteur cohérent. Cette option est plus facile à utiliser et s'exécute plus rapidement que l'analyseur Regex. Il est préférable de l'utiliser lorsqu'elle est disponible. Lorsque vous utilisez cette option, vous devez spécifier la paire clé-valeur `Delimiter`.

`TimestampField`

Spécifie le champ JSON qui contient l'horodatage de l'enregistrement. Est uniquement utilisé avec l'élément `RecordParser` `SingleLineJson`. La paire clé-valeur est facultative. Si cet élément n'est pas spécifié, Kinesis Agent pour Windows utilise l'heure à laquelle l'enregistrement a été

lu comme horodatage. La spécification de cette paire clé-valeur permet à de générer Kinesis de latence plus précises.

`TimestampFormat`

Spécifie comment analyser la date et l'heure associées à l'enregistrement. La valeur est la chaîne `epoch` ou une chaîne de format date/heure .NET. Si la valeur est `epoch`, l'heure est analysée en fonction de l'heure UNIX Epoch. Pour plus d'informations sur l'heure UNIX Epoch, consultez Heure UNIX. Pour plus d'informations sur les chaînes de format date/heure .NET, consultezChaînes de format de date et d'heure personnalisées dans la documentation Microsoft .NET). Cette paire clé-valeur est uniquement obligatoire si l'analyseur d'enregistrements `Timestamp` est spécifié ou si l'analyseur d'enregistrements `SingleLineJson` est spécifié en même temps que la paire clé-valeur `TimestampField`.

`Pattern`

Spécifie une expression régulière qui doit correspondre à la première ligne d'un enregistrement comportant potentiellement plusieurs lignes. Cette paire clé-valeur est uniquement obligatoire pour l'analyseur d'enregistrements `Regex`.

`ExtractionPattern`

Spécifie une expression régulière qui doit utiliser des groupes nommés. L'enregistrement est analysé à l'aide de cette expression régulière et les groupes nommés forment les champs de l'enregistrement analysé. Ces champs sont ensuite utilisés comme base pour construire des objets ou des documents JSON ou XML qui sont ensuite diffusés par les récepteurs vers différents services AWS. Cette paire clé-valeur est facultative et est disponible avec l'`Regex`analyseur d'enregistrement et l'analyseur d'horodatage.

Le nom de groupe `Timestamp` fait l'objet d'un traitement spécial, car il indique à l'analyseur `Regex` le champ qui contient la date et l'heure de chaque enregistrement dans chaque fichier journal.

`Delimiter`

Spécifie le caractère ou la chaîne qui sépare chaque élément de chaque enregistrement de journal. Cette paire clé-valeur doit être (et peut uniquement être) utilisée avec l'analyseur d'enregistrements `Delimited`. Utilisez la séquence de deux caractères `\t` pour représenter le caractère de tabulation.

`HeaderPattern`

Spécifie une expression régulière correspondant à la ligne du fichier journal qui contient l'ensemble des en-têtes de l'enregistrement. Si le fichier journal ne contient aucune information d'en-tête, utilisez la paire clé-valeur `Headers` pour spécifier les en-têtes implicites. La paire clé-valeur `HeaderPattern` est facultative et est uniquement valide pour l'analyseur d'enregistrements `Delimited`.

> **Note**
>
> Une entrée d'en-tête vide (longueur 0) pour une colonne provoque le filtrage des données de cette colonne dans la sortie finale de la sortie analysée `DirectorySource`.

`Headers`

Spécifie les noms des colonnes de données analysées à l'aide du délimiteur spécifié. Cette paire clé-valeur est facultative et est uniquement valide pour l'analyseur d'enregistrements `Delimited`.

> **Note**
>
> Une entrée d'en-tête vide (longueur 0) pour une colonne provoque le filtrage des données de cette colonne dans la sortie finale de la sortie analysée `DirectorySource`.

`RecordPattern`

Spécifie une expression régulière qui identifie les lignes du fichier journal contenant des données d'enregistrement. En dehors de la ligne d'en-tête facultative identifiée par `HeaderPattern`, les lignes qui ne correspondent pas à l'élément `RecordPattern` spécifié sont ignorées pendant le traitement. Cette paire clé-valeur est facultative et est uniquement valide pour l'analyseur d'enregistrements `Delimited`. Si cet élément n'est pas fourni, par défaut, toute ligne qui ne correspond pas à l'élément facultatif `HeaderPattern` ou `CommentPattern` est considérée comme une ligne contenant des données d'enregistrement analysables.

`CommentPattern`

Spécifie une expression régulière qui identifie les lignes du fichier journal devant être exclues avant l'analyse des données du fichier journal. Cette paire clé-valeur est facultative et est uniquement valide pour l'analyseur d'enregistrements `Delimited`. Si cet élément n'est pas

fourni, par défaut, toute ligne qui ne correspond pas à l'élément facultatif `HeaderPattern` est considérée comme une ligne contenant des données d'enregistrement analysables, sauf si `RecordPattern` est spécifié.

`TimeZoneKind`

Indique si l'horodatage dans le fichier journal doit être considéré dans le fuseau horaire local ou dans le fuseau horaire en temps universel coordonné (UTC). Cette option est facultative et a pour valeur par défaut l'heure UTC. Les seules valeurs valides pour cette paire clé-valeur sont `Local` ou `UTC`. L'horodatage n'est jamais modifié si `TimeZoneKind` n'est pas spécifié ou si la valeur est UTC. L'horodatage est converti en UTC lorsque le paramètre`TimeZoneKind`Valeur est`Local`et que le récepteur recevant l'horodatage est CloudWatch Logs ou que l'enregistrement analysé est envoyé à d'autres récepteurs. Les dates et heures qui sont intégrées dans les messages ne sont pas converties.

`SkipLines`

Lorsque cet élément est spécifié, il contrôle le nombre de lignes ignorées au début de chaque fichier journal avant l'exécution de l'analyse des enregistrements. Cet élément est facultatif et sa valeur par défaut est 0.

Encodage

Par défaut, Kinesis Agent pour Windows peut détecter automatiquement l'encodage à partir du signet d'octème. Cependant, l'encodage automatique peut ne pas fonctionner correctement sur certains formats Unicode plus anciens. L'exemple suivant spécifie le codage requis pour diffuser un journal Microsoft SQL Server.

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

Pour obtenir la liste des noms d'encodages, consultezListe des encodagesdans la documentation Microsoft .NET.

ExtractionRegexOptions

Vous pouvez utiliser`ExtractionRegexOptions`Pour simplifier les expressions régulières. La paire clé-valeur est facultative. La valeur par défaut est `"None"`.

L'exemple suivant spécifie que l'`"."`correspond à n'importe quel caractère, y compris`\r\n`.

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

Pour obtenir la liste des champs possibles pour ExtractionRegexOptions, consultez l'Énumération RegexOptionsdans la documentation Microsoft .NET.

Analyseur d'enregistrements **Regex**

Vous pouvez analyser les journaux de texte non structuré à l'aide de l'analyseur d'enregistrements `Regex`, ainsi qu'avec les paires clé-valeur `TimestampFormat`, `Pattern` et `ExtractionPattern`. Par exemple, supposons que votre fichier journal ressemble à ce qui suit :

```
[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'
```

Vous pouvez spécifier l'expression régulière suivante pour la paire clé-valeur `Pattern` afin de faciliter la décomposition du fichier journal en enregistrements de journaux individuels :

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

Cette expression régulière correspond à la séquence suivante :

1. Début de la chaîne évaluée
2. Un ou plusieurs caractères alphabétiques entre crochets
3. Horodatage entre crochets L'horodatage correspond à la séquence suivante :
   a. Année sur quatre chiffres
   b. Barre oblique
   c. Mois sur deux chiffres
   d. Barre oblique
   e. Jour sur deux chiffres
   f. Caractère espace
   g. Heures sur deux chiffres

h. Deux-points
i. Minutes sur deux chiffres
j. Deux-points
k. Secondes sur deux chiffres
l. Point
m. Millisecondes sur trois chiffres

Vous pouvez spécifier le format suivant pour la paire clé-valeur `TimestampFormat`afin de convertir l'horodatage textuel en date et heure :

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

Vous pouvez utiliser l'expression régulière suivante pour extraire les champs de l'enregistrement de journal via la paire clé-valeur `ExtractionPattern`.

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

Cette expression régulière correspond aux groupes suivants en séquence :

1. `Severity`— Un ou plusieurs caractères alphabétiques entre crochets.
2. `TimeStamp`— Consultez la description précédente pour l'horodatage.
3. Trois séquences anonymes de zéro ou plusieurs caractères entre crochets sont ignorées.
4. `SubSystem`— Un ou plusieurs caractères alphabétiques entre crochets.
5. `Module`— Un ou plusieurs caractères alphabétiques entre crochets.
6. Une séquence anonyme de zéro ou plusieurs caractères entre crochets est ignorée.
7. Un espace anonyme est ignoré.
8. `Message`— Zéro ou plusieurs caractères entre guillemets simples.

La déclaration de sources suivante combine ces expressions régulières et le format de date/heure pour fournir à Kinesis Agent for Windows les instructions complètes d'analyse de ce type de fichier journal.

```
{
    "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**
>
> Les barres obliques inverses dans les fichiers au format JSON doivent être placés dans une séquence d'échappement avec une barre oblique inverse supplémentaire.

Pour plus d'informations sur les expressions régulières, consultez Langage des expressions régulières - Aide-mémoire dans la documentation Microsoft .NET.

Analyseur d'enregistrements **`Delimited`**

Vous pouvez utiliser l'analyseur d'enregistrements `Delimited` pour analyser des fichiers journaux et des fichiers de données semi-structurés dans lesquels il y a une séquence de caractères constante séparant chaque colonne de données dans chaque ligne de données. Par exemple, les fichiers CSV, utilisent une virgule pour séparer chaque colonne de données et les fichiers TSV utilisent une tabulation.

Supposons que vous souhaitiez analyser un fichier journal Microsoft NPS Database Format généré par un serveur de stratégies réseau. Ce type de fichier peut se présenter comme suit :

```
"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,,,,
```

L'exemple de fichier de configuration `appsettings.json` suivant comprend une déclaration `DirectorySource` qui utilise l'analyseur d'enregistrements `Delimited` pour analyser ce texte dans une représentation d'objet. Ensuite, il diffuse des données au format JSON vers 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"
        }
    ]
}
```

Les données au format JSON diffusées vers Kinesis Data Firehose se présentent comme suit :

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


Analyseur d'enregistrements **`SysLog`**

Pour l'analyseur d'enregistrements `SysLog`, la sortie analysée à partir de la source inclut les informations suivantes :

Attribut	Type	Description
`SysLogTimeStamp`	Chaîne	Date et heure d'origine du fichier journal au format syslog.
`Hostname`	Chaîne	Nom de l'ordinateur sur lequel le fichier journal au format syslog réside.
`Program`	Chaîne	Nom de l'application ou du service qui a généré le fichier journal.
`Message`	Chaîne	Message de journal généré par l'application ou le service.
`TimeStamp`	Chaîne	Date et heure d'analyse au format ISO 8601.

Vous trouverez ci-dessous un exemple de données SysLog transformées en JSON :

```
{
    "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

Voici un récapitulatif des paires clé-valeur disponibles pour la source `DirectorySource` et les éléments `RecordParser` associés à ces paires clé-valeur.

Nom de clé	RecordParser	Remarques
`SourceType`	Obligatoire pour tous	Doit avoir la valeur `Directory Source`

Nom de clé	RecordParser	Remarques
`Directory`	Obligatoire pour tous	
`FileNameFilter`	Facultatif pour tous	
`RecordParser`	Obligatoire pour tous	
`TimestampField`	Facultatif pour `SingleLineJson`	
`TimestampFormat`	Obligatoire pour `Timestamp` et obligatoire pour `SingleLineJson` si `TimestampField` est spécifié	
`Pattern`	Obligatoire pour `Regex`	
`ExtractionPattern`	Facultatif pour `Regex`	Obligatoire pour le `xml` si le récepteur spécifie le format `json` ou `Regex`
`Delimiter`	Obligatoire pour `Delimited`	
`HeaderPattern`	Facultatif pour `Delimited`	
`Headers`	Facultatif pour `Delimited`	
`RecordPattern`	Facultatif pour `Delimited`	
`CommentPattern`	Facultatif pour `Delimited`	

Nom de clé	RecordParser	Remarques
`TimeZoneKind`	Facultatif pour `Regex`, `Timestamp` , `SysLog` et `SingleLineJson` quand un champ d'horodatage est identifié	
`SkipLines`	Facultatif pour tous	

Configuration de ExchangeLogSource

Le type `ExchangeLogSource` est utilisé pour collecter les journaux à partir de Microsoft Exchange. Exchange génère des journaux dans différents types de formats. Ce type de source peut tous les analyser. Bien qu'il soit possible de les analyser en utilisant le type `DirectorySource` avec l'analyseur d'enregistrements `Regex`, il est beaucoup plus simple d'utiliser `ExchangeLogSource`. En effet, vous n'avez pas besoin de concevoir et de fournir des expressions régulières pour les formats de fichier journal. Voici un exemple de déclaration `ExchangeLogSource` :

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

Toutes les déclarations Exchange peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"ExchangeLogSource"` (obligatoire).

`Directory`

Chemin du répertoire contenant les fichiers journaux (obligatoire).

`FileNameFilter`

Limite le cas échéant l'ensemble de fichiers dans le répertoire dans lequel les données de journal sont collectées sur la base d'un modèle d'attribution de noms de fichiers contenant des caractères

génériques. Si cette paire clé-valeur n'est pas spécifiée, par défaut, les données de journal de tous les fichiers du répertoire sont collectées.

`TimestampField`

Nom de la colonne contenant la date et l'heure de l'enregistrement. Cette paire clé-valeur est facultative et n'a pas besoin d'être spécifiée si le nom du champ est `date-time` ou `DateTime`. Sinon, elle est obligatoire.

Configuration de W3SVCLogSource

Le type `W3SVCLogSource` est utilisé pour collecter les journaux provenant d'Internet Information Services (IIS) pour Windows.

Voici un exemple de déclaration `W3SVCLogSource` :

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

Toutes les déclarations `W3SVCLogSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"W3SVCLogSource"` (obligatoire).

`Directory`

Chemin du répertoire contenant les fichiers journaux (obligatoire).

`FileNameFilter`

Limite le cas échéant l'ensemble de fichiers dans le répertoire dans lequel les données de journal sont collectées sur la base d'un modèle d'attribution de noms de fichiers contenant des caractères génériques. Si cette paire clé-valeur n'est pas spécifiée, par défaut, les données de journal de tous les fichiers du répertoire sont collectées.

Configuration de UlsSource

Le type `UlsSource` est utilisé pour collecter les journaux à partir de Microsoft SharePoint. Voici un exemple de déclaration `UlsSource` :

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

Toutes les déclarations `UlsSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"UlsSource"` (obligatoire).

`Directory`

Chemin du répertoire contenant les fichiers journaux (obligatoire).

`FileNameFilter`

Limite le cas échéant l'ensemble de fichiers dans le répertoire dans lequel les données de journal sont collectées sur la base d'un modèle d'attribution de noms de fichiers contenant des caractères génériques. Si cette paire clé-valeur n'est pas spécifiée, par défaut, les données de journal de tous les fichiers du répertoire sont collectées.

Configuration de WindowsEventLogSource

Le type `WindowsEventLogSource` est utilisé pour collecter des événements à partir du service de journal des événements Windows. Voici un exemple de déclaration `WindowsEventLogSource` :

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

Toutes les déclarations `WindowsEventLogSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"WindowsEventLogSource"` (obligatoire).

`LogName`

Les événements sont collectés à partir du journal spécifié. Les valeurs courantes incluent `Application`, `Security` et `System`, mais vous pouvez spécifier n'importe quel nom de journal d'événements Windows valide. Cette paire clé-valeur est requise.

`Query`

(Facultatif) Limite les événements en sortie à partir de `WindowsEventLogSource`. Si cette paire clé-valeur n'est pas spécifiée, par défaut, tous les événements sont générés en sortie. Pour plus d'informations sur la syntaxe de cette valeur, consultez Event Queries et Event XML dans la documentation Windows. Pour plus d'informations sur les définitions de niveau de journalisation, consultez Event Types dans la documentation Windows.

`IncludeEventData`

(Facultatif) Active la collecte et la diffusion des données d'événements spécifiques au fournisseur associées à des événements provenant du journal des événements Windows spécifié lorsque la valeur de cette paire clé-valeur est `"true"`. Seules les données d'événement qui peuvent être sérialisées avec succès sont incluses. Cette paire clé-valeur est facultative et, si elle n'est pas spécifiée, les données d'événement spécifiques au fournisseur ne sont pas collectées.

> **Note**
>
> L'inclusion des données d'événement peut augmenter de manière significative la quantité de données diffusées à partir de cette source. La taille maximale d'un événement peut être de 262 143 octets en incluant les données d'événement.

La sortie analysée à partir de `WindowsEventLogSource` contient les informations suivantes :

Attribut	Type	Description
`EventId`	Int	Identifiant du type d'événement.
`Description`	Chaîne	Texte qui décrit les détails de l'événement.

Attribut	Type	Description
`LevelDisplayName`	Chaîne	Catégorie d'événement (Erreur, Avertissement, Information, Audit réussi, Échec de l'audit).
`LogName`	Chaîne	Lieu d'enregistrement de l'événement (les valeurs courantes sont `Application` , `Security` et `System`, mais il existe de nombreuses possibilités).
`MachineName`	Chaîne	Ordinateur ayant enregistré l'événement.
`ProviderName`	Chaîne	Application ou service ayant enregistré l'événement.
`TimeCreated`	Chaîne	Moment où l'événement s'est produit au format ISO 8601.
`Index`	Int	Emplacement de l'entrée dans le journal.
`UserName`	Chaîne	Auteur de l'entrée s'il est connu.

Attribut	Type	Description
`Keywords`	Chaîne	Type d'événement. Les valeurs standard incluent `AuditFailure` (événements d'audit de sécurité ayant échoué), `AuditSuccess` (événements d'audit de sécurité ayant réussi), `Classic` (événements déclenchés via la fonction `RaiseEvent`), `Correlation Hint` (événements de transfert), `SQM` (événements de mécanisme de qualité de service), `WDI Context` (événements de contexte de l'infrastructure de diagnostics Windows) et `WDI Diag` (événements de diagnostic de l'infrastructure de diagnostics Windows).
`EventData`	Liste d'objets	Données supplémentaires spécifiques au fournisseur et facultatives concernant l'événement de journal. Cet élément est uniquement inclus si la valeur de la paire clé-valeur `IncludeEventData` est `"true"`.

Voici un exemple d'événement transformé au format JSON :

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

Configuration de WindowsEventLogPollingSource

`WindowsEventLogPollingSource`utilise un mécanisme basé sur l'interrogation pour rassembler tous les nouveaux événements du journal des événements qui correspondent aux paramètres configurés. L'intervalle d'interrogation est mis à jour dynamiquement entre 100 ms et 5000 ms en fonction du nombre d'événements recueillis lors du dernier sondage. Voici un exemple de déclaration `WindowsEventLogPollingSource` :

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

Toutes les déclarations `WindowsEventLogPollingSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale "`WindowsEventLogPollingSource`" (obligatoire).

`LogName`

Spécifie le journal. Les options valides sont`Application`,`Security`,`System`, ou d'autres journaux valides.

`IncludeEventData`

Facultatif. Quand`true`, spécifie que EventData supplémentaires lorsqu'il est diffusé en format JSON et XML est inclus. La valeur par défaut est `false`.

`Query`

Facultatif. Les journaux d'événements Windows prennent en charge l'interrogation d'événements à l'aide d'expressions XPath, que vous pouvez spécifier en utilisant`Query`. Pour de plus amples informations, veuillez consulterRequêtes d'événement et XML d'événementdans la documentation Microsoft.

`CustomFilters`

Facultatif. Liste des filtres séparés par un point-virgule (`;`). Les filtres suivants peuvent être spécifiés.

`ExcludeOwnSecurityEvents`

Exclut les événements de sécurité générés par Kinesis Agent pour Windows lui-même.

Configuration de WindowsETWEventSource

Le type `WindowsETWEventSource` est utilisé pour collecter des suivis d'événements d'application et de service à l'aide d'une fonctionnalité nommée Event Tracing for Windows (ETW). Pour plus d'informations, consultez Event tracing dans la documentation Windows.

Voici un exemple de déclaration `WindowsETWEventSource` :

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

Toutes les déclarations `WindowsETWEventSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"WindowsETWEventSource"` (obligatoire).

`ProviderName`

Spécifie le fournisseur d'événements à utiliser pour collecter les événements de suivi. Il doit s'agir d'un nom de fournisseur ETW valide pour un fournisseur installé. Pour déterminer quels sont les

fournisseurs installés, exécutez la commande suivante dans une fenêtre d'invite de commande Windows :

```
logman query providers
```

`TraceLevel`

Spécifie les catégories d'événements de suivi qui doivent être collectées. Les valeurs autorisées incluent `Critical`, `Error`, `Warning`, `Informational` et `Verbose`. La signification exacte dépend du fournisseur ETW qui est sélectionné.

`MatchAnyKeyword`

Cette valeur est un nombre de 64 bits, dans lequel chaque bits représente un mot-clé. Chaque mot-clé décrit une catégorie d'événements à collecter. Pour connaître les mots-clés pris en charge et leurs valeurs, ainsi que leur lien avec `TraceLevel`, consultez la documentation de ce fournisseur. Par exemple, pour obtenir des informations sur le fournisseur ETW CLR, consultez Niveaux et mots clés ETW du CLR dans la documentation Microsoft .NET Framework.

Dans l'exemple précédent, 32768 (0x00008000) représente l'élément `ExceptionKeyword` du fournisseur ETW CLR qui demande au fournisseur de collecter des informations sur les exceptions déclenchées. Bien que le format JSON ne prenne pas en charge en mode natif les constantes hexadécimales, vous pouvez les spécifier pour `MatchAnyKeyword` en les plaçant dans une chaîne. Vous pouvez également spécifier plusieurs constantes séparées par des virgules. Par exemple, utilisez la commande suivante pour spécifier à la fois `ExceptionKeyword` et `SecurityKeyword` (0x00000400) :

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

Pour assurer que tous les mots-clés spécifiés sont activés pour un fournisseur, plusieurs valeurs de mots-clés sont combinées en utilisant OR et transmises à ce fournisseur.

La sortie de l'élément `WindowsETWEventSource` contient les informations suivantes pour chaque événement :

Attribut	Type	Description
`EventName`	Chaîne	Type d'événement qui s'est produit.
`ProviderName`	Chaîne	Fournisseur ayant détecté l'événement.
`FormattedMessage`	Chaîne	Résumé textuel de l'événement.
`ProcessID`	Int	Processus ayant signalé l'événement.
`ExecutingThreadID`	Int	Thread du processus ayant signalé l'événement.
`MachineName`	Chaîne	Nom de l'ordinateur de bureau ou du serveur qui signale l'événement.
`Payload`	Table de hachage	Table avec une clé de type chaîne et n'importe quel type d'objet comme valeur. La clé est le nom de l'élément de charge utile et la valeur est la valeur de l'élément de charge utile. La charge utile dépend du fournisseur.

Voici un exemple d'événement transformé au format JSON :

```
{
     "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
      }
}
```

Configuration de WindowsPerformanceCounterSource

Le type `WindowsPerformanceCounterSource` collecte les métriques de compteur de performances à partir de Windows. Voici un exemple de déclaration `WindowsPerformanceCounterSource` :

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

Toutes les déclarations `WindowsPerformanceCounterSource` peuvent fournir les paires clé-valeur suivantes :

`SourceType`

Doit avoir pour valeur la chaîne littérale `"WindowsPerformanceCounterSource"` (obligatoire).

`Categories`

Spécifie un ensemble de groupes de métriques de compteur de performances à collecter à partir de Windows. Chaque groupe de métriques contient les paires clé-valeur suivantes :

`Category`

Spécifie l'ensemble de métriques de compteur à collecter (obligatoire).

`Instances`

Spécifie l'ensemble d'objets d'intérêt lorsqu'il y a un ensemble unique de compteurs de performances par objet. Par exemple, lorsque la catégorie est `LogicalDisk`, il y a un ensemble de compteurs de performances par unité de disque. La paire clé-valeur est facultative. Vous pouvez utiliser les caractères génériques `*` et `?` pour représenter plusieurs instances. Pour regrouper les valeurs de toutes les instances, spécifiez `_Total`.

Vous pouvez également utiliser`InstanceRegex`, qui accepte les expressions régulières qui contiennent le`*`caractère générique faisant partie du nom de l'instance.

`Counters`

Spécifie les métriques à collecter pour la catégorie spécifiée. Cette paire clé-valeur est requise. Vous pouvez utiliser les caractères génériques `*` et `?` pour représenter plusieurs compteurs. Vous pouvez spécifier `Counters` en utilisant uniquement le nom ou en utilisant le nom et l'unité. Si les unités de compteur ne sont pas spécifiées, Kinesis Agent pour Windows tente de déduire les unités à partir du nom. Si ces conclusions sont incorrectes, l'unité peut être explicitement spécifiée. Vous pouvez modifier les noms `Counter` si vous le souhaitez. La représentation la plus complexe d'un compteur est un objet avec les paires clé-valeur suivantes :

`Counter`

Nom du compteur. Cette paire clé-valeur est requise.

`Rename`

Nom du compteur à présenter au récepteur. La paire clé-valeur est facultative.

`Unit`

Signification de la valeur qui est associée au compteur. Pour obtenir la liste complète des noms d'unité valides, consultez la documentation sur les unités dansMetricDatumdans leRéférence d'API Amazon CloudWatch.

Voici un exemple de spécification de compteur complexe :

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

`WindowsPerformanceCounterSource`peut uniquement être utilisé avec un récepteur qui spécifie un récepteur Amazon CloudWatch. Utilisez un récepteur distinct si les métriques prédéfinies Kinesis Agent for Windows sont également diffusées vers CloudWatch. Examinez le journal Kinesis Agent pour Windows après le démarrage du service pour déterminer quelles sont les unités qui ont été déduites pour les compteurs lorsque les unités n'ont pas été spécifiées dans le champ`WindowsPerformanceCounterSource`Déclarations. Utilisez PowerShell pour déterminer les noms valides pour les catégories, instances et compteurs.

Pour afficher des informations sur toutes les catégories, y compris les compteurs associés aux ensembles de compteurs, exécutez cette commande dans une fenêtre PowerShell :

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

Pour déterminer quelles sont les instances disponibles pour chacun des compteurs de l'ensemble de compteurs, exécutez une commande similaire à ce qui suit dans une fenêtre PowerShell :

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

La valeur du paramètre `Counter` doit être l'un des chemins d'accès d'un membre `PathsWithInstances` répertoriés au cours du précédent appel de la commande `Get-Counter -ListSet`.

Source des métriques prédéfinies de Kinesis Agent pour Windows

En plus des sources de métriques ordinaires telles que le`WindowsPerformanceCounterSource`type (voirConfiguration de WindowsPerformanceCounterSource), le type de récepteur CloudWatch peut recevoir des métriques à partir d'une source spéciale qui collecte des métriques sur Kinesis Agent pour Windows lui-même. Les métriques Kinesis Agent pour Windows sont également disponibles dans le manuel`KinesisTap`catégorie des compteurs de performances Windows.

La .`MetricsFilter`clé-valeur des déclarations de récepteurs CloudWatch spécifie les métriques qui sont diffusées vers CloudWatch à partir de la source de métriques Kinesis Agent pour Windows intégrée. La valeur est une chaîne qui contient une ou plusieurs expressions de filtre séparées par des points-virgules. Par exemple :

`"MetricsFilter": "ExpressionFiltre1;ExpressionFiltre2"`

Une métrique qui correspond à une ou plusieurs expressions de filtre est diffusée vers CloudWatch.

Les métriques d'instance unique sont de nature globale et ne sont pas liées à une source particulière ou à un récepteur particulier. Les métriques d'instances multiples sont des dimensions basées sur la déclaration source ou de récepteurs `Id`. Chaque type de source ou de récepteur peut avoir un ensemble de métriques différent.

Pour obtenir la liste des noms de métriques prédéfinies Kinesis Agent pour Windows, consultezListe des mesures Kinesis Agent pour Windows.

Pour les métriques d'instance unique, l'expression de filtre est le nom de la métrique, par exemple :

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

Pour les métriques d'instances multiples, l'expression de filtre se présente sous la forme du nom de la métrique, d'un point (`.`), puis de l'élément `Id` de la déclaration source ou de récepteurs qui a généré cette métrique. Supposons par exemple qu'il y a une déclaration de récepteurs avec un élément `Id` ayant pour valeur `MyFirehose` :

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

Vous pouvez utiliser des modèles de caractères génériques spéciaux conçus pour faire une distinction entre les métriques d'instance unique et les métriques d'instances multiples.

- L'astérisque (`*`) correspond à zéro ou plusieurs caractères, à l'exception des points (`.`).
- Le point d'interrogation (`?`) correspond à un caractère, à l'exception des points.
- Tous les autres caractères correspondent uniquement à eux-mêmes.
- `_Total` est un jeton spécial qui provoque l'agrégation de toutes les valeurs d'instances multiples correspondantes au sein de la dimension.

L'exemple suivant représente toutes les métriques d'instance unique :

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

Étant donné qu'un astérisque ne correspond pas à un point, seules les métriques d'instance unique sont incluses.

L'exemple suivant représente toutes les métriques d'instances multiples :

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

L'exemple suivant représente toutes les métriques (instance unique ou instances multiples) :

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

L'exemple suivant regroupe toutes les métriques d'instances multiples pour l'ensemble des sources et des récepteurs :

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

L'exemple suivant regroupe toutes les métriques Kinesis Data Firehose pour tous les récepteurs Kinesis Data Firehose :

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

L'exemple suivant représente toutes les métriques d'erreur d'instance unique et d'instances multiples :

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

L'exemple suivant représente toutes les métriques d'erreur irrécupérable agrégées pour toutes les sources et tous les récepteurs :

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

Pour plus d'informations sur la façon de spécifier un canal utilisant la source de métriques prédéfinies Kinesis Agent pour Windows, consultezConfiguration de Kinesis Agent pour les canaux métriques Windows.

Liste des mesures Kinesis Agent pour Windows

Voici une liste des métriques d'instance unique et d'instances multiples qui sont disponibles pour Kinesis Agent pour Windows.

Métriques d'instance unique

Les métriques d'instance unique disponibles sont les suivantes :

`KinesisTapBuildNumber`

Numéro de version de Kinesis Agent pour Windows.

`PipesConnected`

Nombre de pipelines ayant connecté leur source à leur récepteur avec succès.

`PipesFailedToConnect`

Nombre de pipelines n'ayant pas réussi à connecter leur source à leur récepteur.

`SinkFactoriesFailedToLoad`

Nombre de types de récepteurs n'ayant pas pu être chargés dans l'Agent Kinesis pour Windows.

`SinkFactoriesLoaded`

Nombre de types de récepteurs ayant été chargés dans Kinesis Agent pour Windows avec succès.

`SinksFailedToStart`

Nombre de récepteurs dont le démarrage a échoué, généralement en raison de déclarations de récepteurs incorrectes.

`SinksStarted`

Nombre de récepteurs ayant démarré avec succès.

`SourcesFailedToStart`

Nombre de sources dont le démarrage a échoué, généralement en raison de déclarations de sources incorrectes.

`SourcesStarted`

Nombre de sources ayant démarré avec succès.

`SourceFactoriesFailedToLoad`

Nombre de types de sources n'ayant pas pu être chargés dans Kinesis Agent for Windows.

`SourceFactoriesLoaded`

Nombre de types de sources ayant été chargés dans Kinesis Agent pour Windows.

Métriques d'instances multiples

Les métriques d'instances multiples disponibles sont les suivantes :

Métriques DirectorySource

`DirectorySourceBytesRead`

Nombre d'octets ayant été lus au cours de l'intervalle pour cet élément `DirectorySource`.

`DirectorySourceBytesToRead`

Nombre d'octets connus disponibles en lecture n'ayant pas encore été lus par l'Agent Kinesis pour Windows.

`DirectorySourceFilesToProcess`

Nombre de fichiers connus à examiner qui n'ont pas encore été examinés par Kinesis Agent for Windows.

`DirectorySourceRecordsRead`

Nombre d'enregistrements ayant été lus au cours de l'intervalle pour cet élément `DirectorySource`.

Métriques WindowsEventLogSource

`EventLogSourceEventsError`

Nombre d'événements du journal d'événements Windows n'ayant pas été lus avec succès.

`EventLogSourceEventsRead`

Nombre d'événements du journal d'événements Windows ayant été lus avec succès.

Métriques de récepteur KinesisFirehose

`KinesisFirehoseBytesAccepted`

Nombre d'octets ayant été acceptés au cours de l'intervalle.

`KinesisFirehoseClientLatency`

Temps écoulé entre la génération des enregistrements et leur diffusion vers le service Kinesis Data Firehose.

`KinesisFirehoseLatency`

Temps écoulé entre le début et la fin de la diffusion des enregistrements pour le service Kinesis Data Firehose.

`KinesisFirehoseNonrecoverableServiceErrors`

Nombre de fois où des enregistrements n'ont pas pu être envoyés sans erreur au service Kinesis Data Firehose malgré les nouvelles tentatives.

`KinesisFirehoseRecordsAttempted`

Nombre d'enregistrements ayant tenté d'être diffusés vers le service Kinesis Data Firehose.

`KinesisFirehoseRecordsFailedNonrecoverable`

Nombre d'enregistrements n'ayant pas pu être diffusés vers le service Kinesis Data Firehose malgré les nouvelles tentatives.

`KinesisFirehoseRecordsFailedRecoverable`

Nombre d'enregistrements ayant pu être diffusés vers le service Kinesis Data Firehose, mais uniquement à la suite de nouvelles tentatives.

`KinesisFirehoseRecordsSuccess`

Nombre d'enregistrements ayant pu être diffusés vers le service Kinesis Data Firehose sans nouvelles tentatives.

`KinesisFirehoseRecoverableServiceErrors`

Nombre de fois où des enregistrements ont pu être envoyés au service Kinesis Data Firehose, mais uniquement à la suite de nouvelles tentatives.

Métriques KinesisStream

`KinesisStreamBytesAccepted`

Nombre d'octets ayant été acceptés au cours de l'intervalle.

`KinesisStreamClientLatency`

Temps écoulé entre la génération des enregistrements et leur diffusion vers le service Kinesis Data Streams.

`KinesisStreamLatency`

Temps écoulé entre le début et la fin de la diffusion des enregistrements pour le service Kinesis Data Streams.

`KinesisStreamNonrecoverableServiceErrors`

Nombre de fois où des enregistrements n'ont pas pu être envoyés sans erreur au service Kinesis Data Streams malgré les nouvelles tentatives.

`KinesisStreamRecordsAttempted`

Nombre d'enregistrements ayant tenté d'être diffusés vers le service de Kinesis Data Streams.

`KinesisStreamRecordsFailedNonrecoverable`

Nombre d'enregistrements n'ayant pas pu être diffusés vers le service Kinesis Data Streams malgré les nouvelles tentatives.

`KinesisStreamRecordsFailedRecoverable`

Nombre d'enregistrements ayant pu être diffusés vers le service Kinesis Data Streams, mais uniquement à la suite de nouvelles tentatives.

`KinesisStreamRecordsSuccess`

Nombre d'enregistrements ayant pu être diffusés vers le service de Kinesis Data Streams sans nouvelles tentatives.

`KinesisStreamRecoverableServiceErrors`

Nombre de fois où des enregistrements ont pu être envoyés au service Kinesis Data Streams, mais uniquement à la suite de nouvelles tentatives.

Métriques CloudWatchLog

`CloudWatchLogBytesAccepted`

Nombre d'octets ayant été acceptés au cours de l'intervalle.

`CloudWatchLogClientLatency`

Temps écoulé entre la génération des enregistrements et leur diffusion vers le service CloudWatch Logs.

`CloudWatchLogLatency`

Temps écoulé entre le début et la fin de la diffusion des enregistrements pour le service CloudWatch Logs.

`CloudWatchLogNonrecoverableServiceErrors`

Nombre de fois où des enregistrements n'ont pas pu être envoyés sans erreur au service CloudWatch Logs malgré les nouvelles tentatives.

`CloudWatchLogRecordsAttempted`

Nombre d'enregistrements ayant tenté d'être diffusés vers le service CloudWatch Logs.

`CloudWatchLogRecordsFailedNonrecoverable`

Nombre d'enregistrements n'ayant pas pu être diffusés vers le service CloudWatch Logs malgré les nouvelles tentatives.

`CloudWatchLogRecordsFailedRecoverable`

Nombre d'enregistrements ayant pu être diffusés vers le service CloudWatch Logs, mais uniquement à la suite de nouvelles tentatives.

`CloudWatchLogRecordsSuccess`

Nombre d'enregistrements ayant pu être diffusés vers le service CloudWatch Logs sans nouvelles tentatives.

`CloudWatchLogRecoverableServiceErrors`

Nombre de fois où des enregistrements ont pu être envoyés au service CloudWatch Logs, mais uniquement à la suite de nouvelles tentatives.

Métriques CloudWatch

`CloudWatchLatency`

Temps écoulé en moyenne entre le début et la fin de la diffusion des métriques pour le service CloudWatch.

`CloudWatchNonrecoverableServiceErrors`

Nombre de fois où des métriques n'ont pas pu être envoyées sans erreur au service CloudWatch malgré les nouvelles tentatives.

`CloudWatchRecoverableServiceErrors`

Nombre de fois où des métriques ont été envoyées sans erreur au service CloudWatch, mais uniquement à la suite de nouvelles tentatives.

`CloudWatchServiceSuccess`

Nombre de fois où des métriques ont été envoyées sans erreur au service CloudWatch sans aucune nouvelle tentative.

Configuration des signets

Par défaut, l'agent Kinesis pour Windows envoie les enregistrements de journaux aux récepteurs créés après le démarrage de l'agent. Parfois, il est utile d'envoyer des enregistrements de journaux plus tôt, par exemple, les enregistrements de journaux créés pendant l'arrêt de Kinesis Agent pour Windows au cours d'une mise à jour automatique. La fonction de signet suit les enregistrements qui ont été envoyés aux récepteurs. Lorsque Kinesis Agent pour Windows est en mode signet et démarre, il envoie tous les enregistrements de journal créés après l'arrêt de Kinesis Agent pour Windows, ainsi que tous les enregistrements de journal créés ultérieurement. Pour contrôler ce comportement, les déclarations de sources basées sur un fichier peuvent éventuellement inclure les paires clé-valeur suivantes :

`InitialPosition`

Spécifie la position initiale du signet. Les valeurs possibles sont les suivantes :

`EOS`

Spécifie la fin du flux (EOS). Seuls les enregistrements de journaux créés pendant l'exécution de l'agent sont envoyés aux récepteurs.

`0`

Tous les enregistrements de journaux et les événements disponibles sont initialement envoyés. Ensuite, un signet est créé pour faire en sorte que chaque nouvel enregistrement de journal et événement créé après le signet soit finalement envoyé, que Kinesis Agent pour Windows soit en cours d'exécution ou non.

`Bookmark`

Le signet est initialisé juste après le dernier enregistrement de journal ou événement. Ensuite, un signet est créé pour faire en sorte que chaque nouvel enregistrement de journal et événement créé après le signet soit finalement envoyé, que Kinesis Agent pour Windows soit en cours d'exécution ou non.

Les signets sont activés par défaut. Les fichiers sont stockés dans l'`%ProgramData%\Amazon\KinesisTap`Répertoire.

`Timestamp`

Les enregistrements de journaux et événements qui sont créés après la valeur `InitialPositionTimestamp` (définition ci-après) sont envoyés. Ensuite, un signet est créé pour faire en sorte que chaque nouvel enregistrement de journal et événement créé après le

signet soit finalement envoyé, que Kinesis Agent pour Windows soit en cours d'exécution ou non.

`InitialPositionTimestamp`

Spécifie le premier horodatage d'enregistrement de journal ou d'événement que vous voulez. Spécifiez cette paire clé-valeur uniquement si `InitialPosition` a la valeur `Timestamp`.

`BookmarkOnBufferFlush`

Ce paramètre peut être ajouté à n'importe quelle source marquable. Lorsqu'il est défini sur`true`, garantit que les mises à jour de signet se produisent uniquement lorsqu'un puits envoie un événement à AWS. Vous ne pouvez abonner qu'un seul puits à une source. Si vous expédiez des journaux vers plusieurs destinations, dupliquez vos sources pour éviter d'éventuels problèmes de perte de données.

Lorsque Kinesis Agent pour Windows a été arrêté pendant longtemps, il peut être nécessaire de supprimer ces signets, car les enregistrements de journaux et les événements qui sont signés peuvent ne plus exister. Les fichiers de signets pour un ID de source donné sont situés dans `%PROGRAMDATA%\Amazon\AWSKinesisTap\`*`source id`*`.bm`.

Les signets ne fonctionnent pas sur les fichiers qui sont renommés ou tronqués. En raison de la nature des événements et des compteurs de performances ETW, ils ne peuvent pas faire l'objet d'un signet.

Déclarations de récepteurs

Les déclarations de récepteurs spécifient la forme sous laquelle les journaux, les événements et les métriques doivent être envoyés aux divers services AWS ainsi que leur emplacement de destination. Les sections suivantes décrivent les configurations des types de récepteur intégrés qui sont disponibles dans Amazon Kinesis Agent pour Microsoft Windows. Étant donné que Kinesis Agent pour Windows est extensible, vous pouvez ajouter des types de récepteurs personnalisés. Chaque type de récepteur nécessite généralement des paires clé-valeur uniques dans les déclarations de configuration qui sont pertinentes pour ce type de récepteur.

Toutes les déclarations de récepteurs peuvent contenir les paires clé-valeur suivantes :

`Id`

Chaîne unique qui identifie un récepteur spécifique au sein du fichier de configuration (obligatoire).

`SinkType`

Nom du type de ce récepteur (obligatoire). Le type de récepteur spécifie la destination des données de journal, d'événement ou de métrique qui sont diffusées par ce récepteur.

`AccessKey`

Spécifie la clé d'accès AWS à utiliser pour autoriser l'accès au service AWS associé au type de récepteur. La paire clé-valeur est facultative. Pour plus d'informations, consultez Configuration de la sécurité des récepteurs.

`SecretKey`

Spécifie la clé secrète AWS à utiliser pour autoriser l'accès au service AWS associé au type de récepteur. La paire clé-valeur est facultative. Pour plus d'informations, consultez Configuration de la sécurité des récepteurs.

`Region`

Spécifie la région AWS qui contient les ressources de destination pour le streaming. La paire clé-valeur est facultative.

`ProfileName`

Spécifie le profil AWS à utiliser pour l'authentification. Cette paire clé-valeur est facultative, mais si elle est spécifiée, elle remplace toute clé d'accès ou clé secrète spécifiée. Pour plus d'informations, consultez Configuration de la sécurité des récepteurs.

`RoleARN`

Spécifie le rôle IAM à utiliser pour accéder au service AWS associé au type de récepteur. Cette option est utile lorsque Kinesis Agent pour Windows s'exécute sur une instance EC2, mais qu'un rôle différent serait plus approprié que le rôle référencé par le profil d'instance. Par exemple, il est possible d'utiliser un rôle entre comptes pour cibler les ressources qui ne sont pas dans le même compte AWS que l'instance EC2. La paire clé-valeur est facultative.

`Format`

Spécifie le type de sérialisation qui est appliqué aux données de journaux et d'événements avant leur diffusion. Les valeurs valides sont `json` et `xml`. Cette option est utile lorsque les analyses en aval dans le pipeline de données nécessitent des données sous un format particulier ou réagissent mieux à un type de format particulier. Cette paire clé-valeur est facultative et, si elle n'est pas spécifiée, le texte ordinaire provenant de la source est diffusé du récepteur vers le service AWS associé à ce type de récepteur.

`TextDecoration`

Lorsqu'aucun élément `Format` n'est spécifié, `TextDecoration` spécifie le texte supplémentaire à inclure lors de la diffusion des enregistrements de journaux ou d'événements. Pour plus d'informations, consultez Configuration des décorations de récepteurs. La paire clé-valeur est facultative.

`ObjectDecoration`

Lorsque l'élément `Format` est spécifié, `ObjectDecoration` spécifie les données supplémentaires à inclure dans l'enregistrement de journal ou d'événement avant la sérialisation ou la diffusion. Pour plus d'informations, consultez Configuration des décorations de récepteurs. La paire clé-valeur est facultative.

`BufferInterval`

Afin de réduire les appels d'API au service AWS associé au type de récepteur, Kinesis Agent pour Windows place en mémoire tampon plusieurs enregistrements de journaux, d'événements ou de métriques avant la diffusion. Cela permet d'économiser de l'argent pour les services qui sont facturés en fonction du nombre d'appels d'API. `BufferInterval` spécifie la durée maximale (en secondes) du placement des enregistrements en mémoire tampon avant leur diffusion vers le service AWS. Cette paire clé-valeur est facultative et, si elle est spécifiée, vous devez utiliser une chaîne pour représenter la valeur.

`BufferSize`

Afin de réduire les appels d'API au service AWS associé au type de récepteur, Kinesis Agent pour Windows place en mémoire tampon plusieurs enregistrements de journaux, d'événements ou de métriques avant la diffusion. Cela permet d'économiser de l'argent pour les services qui sont facturés en fonction du nombre d'appels d'API. `BufferSize` spécifie le nombre maximal d'enregistrements à placer en mémoire tampon avant leur diffusion vers le service AWS. Cette paire clé-valeur est facultative et, si elle est spécifiée, vous devez utiliser une chaîne pour représenter la valeur.

`MaxAttempts`

Spécifie le nombre maximum de tentatives de Kinesis Agent pour Windows pour tente de diffuser un ensemble de journaux, d'événements et de métriques vers un service AWS par si le streaming échoue systématiquement. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez une chaîne pour représenter la valeur. La valeur par défaut est « `3` ».

Pour obtenir des exemples de fichiers de configuration complets utilisant différents types de récepteurs, consultez Diffusion à partir du journal des événements d'application Windows vers les récepteurs.

Rubriques

Configuration du récepteur **`KinesisStream`**

La .`KinesisStream`Le type de récepteur diffuse des enregistrements de journaux et des événements vers le service Kinesis Data Streams. En général, les données diffusées vers Kinesis Data Streams sont traitées par une ou plusieurs applications personnalisées qui s'exécutent via différents services AWS. Les données sont diffusées vers un flux nommé qui est configuré à l'aide Kinesis Data Streams. Pour plus d'informations, consultez le .Guide du développeur Amazon Kinesis Data Streams.

Voici un exemple de déclaration de récepteurs Kinesis Data Streams :

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

Toutes les déclarations de récepteurs `KinesisStream` peuvent fournir les paires clé-valeur supplémentaires suivantes :

`SinkType`

Doit être spécifié. La valeur doit être la chaîne littérale `KinesisStream`.

`StreamName`

Spécifie le nom du flux de données Kinesis qui reçoit les données diffusées à partir de l'`KinesisStream`type d'évier (requis). Avant de diffuser les données, configurez le flux dans AWS Management Console, l'AWS CLI ou via une application à l'aide de l'API Kinesis Data Streams.

`RecordsPerSecond`

Spécifie le nombre maximum d'enregistrements diffusés vers les Kinesis Data Streams par seconde. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez un entier pour représenter la valeur. La valeur par défaut est de 1 000 enregistrements.

`BytesPerSecond`

Spécifie le nombre maximum d'octets diffusés vers les Kinesis Data Streams par seconde. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez un entier pour représenter la valeur. La valeur par défaut est de 1 Mo.

La valeur par défaut de `BufferInterval` pour ce type de récepteur est d'une seconde et la valeur par défaut de `BufferSize` est de 500 enregistrements.

Configuration du récepteur **`KinesisFirehose`**

La .`KinesisFirehose`Le type de récepteur diffuse des enregistrements de journaux et des événements vers le service Kinesis Data Firehose. Kinesis Data Firehose transmet les données diffusées à d'autres services en vue de leur stockage. En général, les données stockées sont ensuite

analysées au cours des étapes ultérieures du pipeline de données. Les données sont diffusées vers un flux de diffusion nommé qui est configuré à l'aide de Kinesis Data Firehose. Pour plus d'informations, consultez le .Guide du développeur Amazon Kinesis Data Firehose.

Voici un exemple de déclaration de récepteurs Kinesis Data Firehose :

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

Toutes les déclarations de récepteurs `KinesisFirehose` peuvent fournir les paires clé-valeur supplémentaires suivantes :

`SinkType`

Doit être spécifié. La valeur doit être la chaîne littérale `KinesisFirehose`.

`StreamName`

Spécifie le nom du flux de diffusion Kinesis Data Firehose qui reçoit les données diffusées à partir de l'`KinesisStream`type d'évier (requis). Avant de diffuser les données, configurez le flux de diffusion via AWS Management Console, l'interface de ligne de commande AWS ou via une application à l'aide de l'API Kinesis Data Firehose.

`CombineRecords`

Lorsqu'il est défini sur`true`, spécifie de combiner plusieurs petits enregistrements dans un enregistrement volumineux avec une taille maximale de 5 Ko. La paire clé-valeur est facultative. Les enregistrements combinés à l'aide de cette fonction sont séparés par\n. Si vous utilisez AWS Lambda pour transformer un enregistrement Kinesis Data Firehose, votre fonction Lambda doit tenir compte du caractère séparateur.

`RecordsPerSecond`

Spécifie le nombre maximum d'enregistrements diffusés vers les Kinesis Data Streams par seconde. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez un entier pour représenter la valeur. La valeur par défaut est de 5 000 enregistrements.

`BytesPerSecond`

Spécifie le nombre maximum d'octets diffusés vers les Kinesis Data Streams par seconde. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez un entier pour représenter la valeur. La valeur par défaut est de 5 Mo.

La valeur par défaut de `BufferInterval` pour ce type de récepteur est d'une seconde et la valeur par défaut de `BufferSize` est de 500 enregistrements.

Configuration du récepteur CloudWatch Sink

La .`CloudWatch`Type de récepteur diffuse les métriques vers le service CloudWatch. Vous pouvez afficher les métriques dans AWS Management Console. Pour plus d'informations, consultez le Guide de l'utilisateur Amazon CloudWatch.

Voici un exemple de déclaration de récepteurs `CloudWatch` :

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

Toutes les déclarations de récepteurs `CloudWatch` peuvent fournir les paires clé-valeur supplémentaires suivantes :

`SinkType`

Doit être spécifié. La valeur doit être la chaîne littérale `CloudWatch`.

`Interval`

Spécifie la fréquence (en secondes) à laquelle Kinesis Agent pour Windows communique les métriques au service CloudWatch. La paire clé-valeur est facultative. Si cet élément est spécifié, utilisez un entier pour représenter la valeur. La valeur par défaut est de 60 secondes. Spécifiez 1 seconde si vous voulez des métriques CloudWatch haute résolution.

`Namespace`

Spécifie l'espace de noms CloudWatch où les données de métriques sont présentées. Les espaces de noms CloudWatch regroupent un ensemble de métriques. La paire clé-valeur est facultative. La valeur par défaut est `KinesisTap`.

`Dimensions`

Spécifie les dimensions CloudWatch utilisées pour isoler les ensembles de métriques au sein d'un espace de noms. Cela peut être utile pour fournir des ensembles de données de métriques distincts pour chaque ordinateur de bureau ou serveur, par exemple. Cette paire clé-valeur est facultative et, si elle est spécifiée, la valeur doit respecter le format suivant : "clé1=valeur1clé2=valeur...". La valeur par défaut est `"ComputerName={computername};InstanceId={instance_id}"`. Cette valeur prend en charge la substitution des variables de récepteur. Pour plus d'informations, consultez Configuration des substitutions de variables de récepteur.

`MetricsFilter`

Spécifie les métriques qui sont diffusées vers CloudWatch à partir de la source de métriques Kinesis Agent for Windows intégrée. Pour plus d'informations sur la source de métriques intégrée de Kinesis Agent pour Windows, notamment sur les détails de la syntaxe de la valeur de cette paire clé-valeur, consultezSource des métriques prédéfinies de Kinesis Agent pour Windows.

Configuration du récepteur **`CloudWatchLogs`**

La .`CloudWatchLogs`Le type de récepteur diffuse des enregistrements de journaux et des événements vers des Amazon CloudWatch Logs. Vous pouvez afficher les journaux dans AWS Management Console ou les traiter au cours des étapes supplémentaires d'un pipeline de données. Les données sont diffusées dans un flux de journaux nommé qui est configuré dans CloudWatch Logs. Les flux de journaux sont organisés en groupes de journaux nommés. Pour plus d'informations, consultez le .Guide de l'utilisateur Amazon CloudWatch Logs.

Voici un exemple de déclaration de récepteurs CloudWatch Logs :

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

Toutes les déclarations de récepteurs `CloudWatchLogs` doivent fournir les paires clé-valeur supplémentaires suivantes :

`SinkType`

Cet élément doit avoir pour valeur la chaîne littérale `CloudWatchLogs`.

`LogGroup`

Spécifie le nom du groupe de journaux CloudWatch Logs qui contient le flux de journaux qui reçoit les enregistrements de journaux et d'événements diffusés par l'`CloudWatchLogs`type d'évier. Si le groupe de journaux spécifié n'existe pas, Kinesis Agent for Windows tente de le créer.

`LogStream`

Spécifie le nom du flux de CloudWatch Logs qui reçoit le flux d'enregistrements de journaux et d'événements diffusés par l'`CloudWatchLogs`type d'évier. Cette valeur prend en charge la substitution des variables de récepteur. Pour plus d'informations, consultez Configuration des substitutions de variables de récepteur. Si le flux de journaux spécifié n'existe pas, Kinesis Agent for Windows tente de le créer.

La valeur par défaut de `BufferInterval` pour ce type de récepteur est d'une seconde et la valeur par défaut de `BufferSize` est de 500 enregistrements. La taille de tampon maximale est de 10 000 enregistrements.

Locale**`FileSystem`**Configuration du récepteur

Le type d'évier`FileSystem`enregistre les enregistrements de journaux et d'événements dans un fichier sur le système de fichiers local au lieu de les diffuser vers les services AWS.`FileSystem`sont utiles pour les tests et les diagnostics. Par exemple, vous pouvez utiliser ce type de collecteur pour examiner les enregistrements avant de les envoyer à AWS.

avec`FileSystem`, vous pouvez également utiliser des paramètres de configuration pour simuler le traitement par lots, la limitation et le retour sur erreur pour imiter le comportement des puits AWS réels.

Tous les enregistrements de toutes les sources connectées à un`FileSystem`sont enregistrés dans le fichier unique spécifié en tant que`FilePath`. Si`FilePath`n'est pas spécifié, les enregistrements sont enregistrés dans un fichier nommé*`SinkId`*`.txt`dans le`%TEMP%`, qui est généralement`C:\Users\`*`UserName`*`\AppData\Local\Temp`où :*`SinkId`*est l'identifiant unique du récepteur et*`UserName`*est le nom d'utilisateur Windows de l'utilisateur actif.

Ce type de puits prend en charge les attributs de décoration de texte. Pour plus d'informations, consultez Configuration des décorations de récepteurs.

Un exemple d'`FileSystem`La configuration de type de récepteur apparaît dans l'exemple suivant.

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

La .`FileSystem`La configuration se compose des paires clé-valeur suivantes.

`SinkType`

Cet élément doit avoir pour valeur la chaîne littérale `FileSystem`.

`FilePath`

Spécifie le chemin d'accès et le fichier où les enregistrements sont enregistrés. La paire clé-valeur est facultative. S'il n'est pas spécifié, la valeur par défaut est*`TempPath`*`\ \`*`SinkId`*`.txt`où :*`TempPath`*est le dossier stocké dans la stratégie%TEMP%Variable et*`SinkId`*est l'identifiant unique du récepteur.

`Format`

Spécifie le format de l'événement à`json`ou`xml`. Cette paire de valeur de clé est facultative et ne respecte pas la casse. S'il est omis, les événements sont écrits dans le fichier en texte brut.

`TextDecoration`

S'applique uniquement aux événements écrits en texte brut. La paire clé-valeur est facultative.

`ObjectDecoration`

S'applique uniquement aux événements où`Format`a la valeur`json`. La paire clé-valeur est facultative.

Utilisation avancée — Limitation des enregistrements et simulation des défaillances

`FileSystem`peut imiter le comportement des puits AWS en simulant la limitation des enregistrements. Vous pouvez utiliser les paires clé-valeur suivantes pour spécifier des attributs de limitation d'enregistrements et de simulation d'échec.

En acquérant un verrou sur le fichier de destination et en empêchant les écritures sur celui-ci, vous pouvez utiliser`FileSystem`pour simuler et examiner le comportement des puits AWS en cas de défaillance du réseau.

L'exemple suivant illustre un`FileSystem`avec des attributs de simulation.

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

`RequestsPerSecond`

Facultatif et spécifié en tant que type de chaîne. S'il n'est pas spécifié, la valeur par défaut est`"5"`. Contrôle le taux de demandes que le sovier traite, c'est-à-dire les écritures dans un fichier, et non le nombre d'enregistrements. Kinesis Agent pour Windows effectue des requêtes par lots aux points de terminaison AWS, de sorte qu'une demande peut contenir plusieurs enregistrements.

`BufferSize`

Facultatif et spécifié en tant que type de chaîne. Spécifie le nombre maximal d'enregistrements d'événements que le sovier effectue par lots avant d'enregistrer dans le fichier.

`MaxBatchSize`

Facultatif et spécifié en tant que type de chaîne. Spécifie la quantité maximale de données d'enregistrement d'événement, en octets, que le sovier effectue par lots avant d'enregistrer dans le fichier.

La limite de taux d'enregistrement maximal est fonction de`BufferSize`, qui détermine le nombre maximum d'enregistrements par demande, et`RequestsPerSecond`. Vous pouvez calculer la limite de taux d'enregistrement par seconde à l'aide de la formule suivante.

Taux d'enregistrement=`BufferSize`*`RequestsPerSecond`

Compte tenu des valeurs de configuration dans l'exemple ci-dessus, il y a un taux d'enregistrement maximal de 1000 enregistrements par seconde.

Configuration de la sécurité des récepteurs

Configuration de l'authentification

Pour que Kinesis Agent pour Windows diffuse des journaux, des événements et des métriques vers les services AWS, l'accès doit être authentifié. Il existe plusieurs manières de fournir une authentification pour Kinesis Agent for Windows. La façon dont vous procédez dépend de la situation dans laquelle l'Agent Kinesis pour Windows s'exécute et des exigences de sécurité spécifiques pour une organisation donnée.

- Si Kinesis Agent pour Windows s'exécute sur un hôte Amazon EC2, la façon la plus simple et la plus sécurisée de fournir une authentification consiste à créer un rôle IAM ayant suffisamment de droits d'accès aux opérations requises pour les services AWS requis, ainsi qu'un profil d'instance EC2 faisant référence à ce rôle. Pour plus d'informations sur la création de profils d'instance, consultez Utilisation de profils d'instance. Pour plus d'informations sur les stratégies à attacher au rôle IAM, consultezConfiguration de l'autorisation.

  Après avoir créé le profil d'instance, vous pouvez l'associer à des instances EC2 qui utilisent Kinesis Agent pour Windows. Si les instances disposent déjà d'un profil d'instance associé, vous pouvez attacher les stratégies appropriées au rôle associé à ce profil d'instance.
- Si Kinesis Agent pour Windows s'exécute sur un hôte EC2 dans un compte, mais que les ressources qui sont la cible du récepteur résident dans un autre compte, vous pouvez créer un rôle IAM pour l'accès entre comptes. Pour de plus amples informations, veuillez consulterDidacticiel : Delegate Access Across AWS Accounts Using IAM Roles. Après avoir créé le rôle entre comptes, spécifiez son Amazon Resource Name (ARN) sous la forme de la valeur de l'`RoleARN`Paire clé-valeur dans la déclaration. Kinesis Agent pour Windows tente ensuite d'assumer le rôle entre comptes spécifié lors de l'accès aux ressources AWS qui sont associées au type de récepteur.
- Si Kinesis Agent pour Windows s'exécute en dehors d'Amazon EC2 (par exemple, sur site), il existe plusieurs options :

- S'il est acceptable d'enregistrer le serveur ou l'ordinateur de bureau sur site en tant qu'instance gérée par Amazon EC2 Systems Manager, utilisez la procédure suivante pour configurer l'authentification :

  1. Utilisez le processus décrit dans Configuration d'AWS Systems Manager dans des environnements hybrides pour créer un rôle de service, créer une activation pour une instance gérée et installer l'agent SSM.
  2. Attachez les stratégies appropriées au rôle de service pour permettre à Kinesis Agent pour Windows d'accéder aux ressources nécessaires pour la diffusion des données à partir des récepteurs configurés. Pour plus d'informations sur les stratégies à attacher au rôle IAM, consultezConfiguration de l'autorisation.
  3. Utilisez le processus décrit dansConfigurationProfileRefreshingAWSCredentialProviderPour actualiser les informations d'identification AWSpour actualiser les informations d'identification AWS.

  Il s'agit de l'approche recommandée pour les instances autres qu'EC2, car les informations d'identification sont gérées en toute sécurité par SSM et AWS.

- S'il est acceptable d'exécuter le service AWSKinesisTap pour Kinesis Agent pour Windows sous un utilisateur spécifique au lieu du compte système par défaut, utilisez la procédure suivante :

  1. Créez un utilisateur IAM dans le compte AWS où les services AWS seront utilisés. Capturez la clé d'accès et la clé secrète de cet utilisateur pendant le processus de création. Vous aurez besoin de ces informations pour les étapes ultérieures de cette procédure.
  2. Attachez à l'utilisateur IAM des stratégies qui autorisent l'accès aux opérations requises pour les services requis. Pour plus d'informations sur les stratégies à attacher à l'utilisateur IAM, consultezConfiguration de l'autorisation.
  3. Modifiez le service AWSKinesisTap sur chaque ordinateur de bureau ou serveur afin qu'il s'exécute sous un utilisateur spécifique et non sous le compte système par défaut.
  4. Créez un profil dans le magasin SDK à l'aide de la clé d'accès et de la clé secrète enregistrées précédemment. Pour plus d'informations, consultez Configuration des informations d'identification AWS.
  5. Mettez à jour le fichier `AWSKinesisTap.exe.config` du répertoire `%PROGRAMFILES%\Amazon\AWSKinesisTap` en spécifiant le nom du profil créé à l'étape précédente. Pour plus d'informations, consultez Configuration des informations d'identification AWS.

Il s'agit de la méthode recommandée pour les hôtes autres qu'EC2 qui ne peuvent pas être des instances gérées, car les informations d'identification sont chiffrées pour l'hôte et l'utilisateur spécifiques.

- S'il est nécessaire d'exécuter le service AWSKinesisTap pour Kinesis Agent pour Windows sous le compte système par défaut, vous devez utiliser un fichier d'informations d'identification partagé. En effet, le compte système n'a pas de profil utilisateur Windows pour activer le magasin SDK. Les fichiers d'informations d'identification ne sont pas chiffrés ; nous déconseillons donc cette approche. Pour plus d'informations sur l'utilisation des fichiers de configuration partagés, consultezConfiguration des informations d'identification AWSdans leKit AWS SDK pour .NET. Si vous utilisez cette approche, nous vous recommandons d'utiliser le chiffrement NTFS et l'accès restreint au fichier de configuration partagé. Les clés doivent faire l'objet d'une rotation par une plateforme de gestion et le fichier de configuration partagé doit être mis à jour lors de la rotation des clés.

Bien qu'il soit possible de fournir directement les clés d'accès et les clés secrètes dans les déclarations de récepteur, cette approche est déconseillée, car les déclarations ne sont pas chiffrées.

Configuration de l'autorisation

Attachez les stratégies appropriées qui suivent à l'utilisateur ou au rôle IAM que utilisera Kinesis Agent pour Windows pour diffuser les données vers les services AWS :

Kinesis Data Streams

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

Pour limiter l'autorisation à un nom de région, de compte ou de flux spécifique, remplacez les astérisques appropriés dans l'ARN par des valeurs spécifiques. Pour plus d'informations, consultez « Amazon Resource Names (ARN) pour Kinesis Data Streams » dans Contrôle de l'accès aux ressources Amazon Kinesis Data Streams avec IAM.

Kinesis Data Firehose

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

Pour limiter l'autorisation à un nom de région, de compte ou de flux de diffusion spécifique, remplacez les astérisques appropriés dans l'ARN par des valeurs spécifiques. Pour de plus amples informations, veuillez consulterContrôle de l'accès avec Amazon Kinesis Data Firehosedans leGuide du développeur Amazon Kinesis Data Firehose.

CloudWatch

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

Pour de plus amples informations, veuillez consulterPrésentation de la gestion des autorisations d'accès à vos ressources CloudWatchdans leGuide de l'utilisateur Amazon CloudWatch Logs.

CloudWatch Logs avec un groupe de journaux et un flux de journaux existants

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

Pour limiter l'accès à une région, un compte, un groupe de journaux ou un flux de journaux spécifique, remplacez les astérisques appropriés dans l'ARN par les valeurs appropriées. Pour de plus amples informations, veuillez consulterPrésentation de la gestion des autorisations d'accès à vos ressources de CloudWatch Logsdans leGuide de l'utilisateur Amazon CloudWatch Logs.

CloudWatch Logs avec des autorisations supplémentaires permettant à Kinesis Agent for Windows de créer des groupes de journaux et des flux de journaux

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

Pour limiter l'accès à une région, un compte, un groupe de journaux ou un flux de journaux spécifique, remplacez les astérisques appropriés dans l'ARN par les valeurs appropriées. Pour de plus amples informations, veuillez consulterPrésentation de la gestion des autorisations d'accès à vos ressources de CloudWatch Logsdans leGuide de l'utilisateur Amazon CloudWatch Logs.

Autorisations requises pour l'extension des variables EC2 Tag

L'utilisation de l'extension des variables avec le préfixe de variable `ec2tag` nécessite l'autorisation `ec2:Describe*`.

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

```
}
```

> **Note**
>
> Vous pouvez combiner plusieurs instructions dans une seule stratégie, du moment que l'élément `Sid` de chaque instruction est unique au sein de cette stratégie. Pour plus d'informations sur la création de stratégies, consultezCréation de stratégies IAMdans leGuide de l'utilisateur IAM.

Configuration**`ProfileRefreshingAWSCredentialProvider`**Pour actualiser les informations d'identification AWS

Si vous utilisez AWS Systems Manager pour les environnements hybrides pour gérer les informations d'identification AWS, Systems Manager fait pivoter les informations d'identification de session dans`c:\Windows\System32\config\systemprofile\.aws\credentials`. Pour plus d'informations sur Systems Manager pour les environnements hybrides, consultezConfiguration d'AWS Systems Manager pour les environnements hybridesdans leGuide de l'utilisateur AWS Systems Manager.

Étant donné que le kit SDK .net AWS ne récupère pas automatiquement de nouvelles informations d'identification, nous fournissons le`ProfileRefreshingAWSCredentialProvider`pour actualiser les informations d'identification.

Vous pouvez utiliser la stratégie`CredentialRef`de n'importe quelle configuration de synchronisation AWS pour référencer un`Credentials`où la stratégie`CredentialType`est défini sur`ProfileRefreshingAWSCredentialProvider`Comme illustré dans l'exemple suivant.

```
{
    "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
    }]
}
```

Une définition d'informations d'identification se compose des attributs suivants sous forme de paires clé-valeur.

`Id`

Définit la chaîne que les définitions de puits peuvent spécifier en utilisant`CredentialRef`pour référencer cette configuration d'informations d'identification.

`CredentialType`

Définissez sur la chaîne littérale`ProfileRefreshingAWSCredentialProvider`.

`Profile`

Facultatif. La valeur par défaut est `default`.

`FilePath`

Facultatif. Indique le chemin d'accès au fichier d'informations d'identification AWS. Si ce paramètre n'est pas spécifié, `%USERPROFILE%/.aws/credentials` est la valeur par défaut.

`RefreshingInterval`

Facultatif. Fréquence à laquelle les informations d'identification sont actualisées, en secondes. Si ce paramètre n'est pas spécifié, `300` est la valeur par défaut.

Configuration des décorations de récepteurs

Les déclarations de récepteurs peuvent éventuellement inclure des paires clé-valeur qui spécifient des données supplémentaires à diffuser vers différents services AWS pour améliorer les enregistrements recueillis à partir de la source.

`TextDecoration`

Utilisez cette paire clé-valeur si aucun élément `Format` n'est spécifié dans la déclaration. La valeur est une chaîne de format spécial dans laquelle se produit une substitution de variables.

Par exemple, supposons qu'un élément `TextDecoration` de `"{ComputerName}:::{timestamp:yyyy-MM-dd HH:mm:ss}:::{_record}"` soit fourni pour un récepteur. Lorsqu'une source émet un enregistrement de journal contenant le texte `The system has resumed from sleep.` et que cette source est connectée au récepteur via un pipeline, le texte `MyComputer1:::2017-10-26 06:14:22:::The system has resumed from sleep.` est diffusé vers le service AWS associé au type de récepteur. La variable `{_record}` fait référence à l'enregistrement de texte d'origine fourni par la source.

`ObjectDecoration`

Utilisez cette paire clé-valeur lorsque `Format` est spécifié dans la déclaration de récepteurs pour ajouter des données supplémentaires avant la sérialisation des enregistrements. Par exemple, supposons qu'un élément `ObjectDecoration` de `"ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd HH:mm:ss}"` soit fourni pour un récepteur qui spécifie l'élément JSON `Format`. Le JSON résultant diffusé vers le service AWS associé au type de récepteur inclut les paires clé-valeur suivantes, en plus des données d'origine provenant de la source :

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

Pour obtenir un exemple d'utilisation de `ObjectDecoration`, consultez la section Didacticiel : Diffuser les fichiers journaux JSON vers Amazon S3 à l'aide de Kinesis Agent pour Windows.

`ObjectDecorationEx`

Spécifie une expression qui permet une extraction et une mise en forme plus flexibles des données par rapport à`ObjectDecoration`. Ce champ peut être utilisé lorsque le format de l'évier est`json`. La syntaxe d'expression est illustrée dans ce qui suit.

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

Par exemple,`ObjectDecorationEx`Attribut :

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

Transforme l'enregistrement littéral :

```
System log message
```

Dans un objet JSON comme suit, avec les valeurs renvoyées par les expressions :

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

Pour plus d'informations sur la formulation des expressions, consultezConseils pour écrire des expressions. La plupart des`ObjectDecoration`doit fonctionner en utilisant la nouvelle syntaxe à l'exception des variables d'horodatage. A`{timestamp:yyyyMMdd}`Champ dans`ObjectDecoration`est exprimé sous forme de`{format(_timestamp,'yyyyMMdd')}`in`ObjectDecorationEx`.

`TextDecorationEx`

Spécifie une expression qui permet une extraction et une mise en forme plus flexibles des données par rapport à`TextDecoration`, comme illustré dans l'exemple suivant.

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

Vous pouvez utiliser`TextDecorationEx`pour composer des objets JSON. Utilisez '@ {'pour échapper à l'accolade ouverte, comme illustré dans l'exemple suivant.

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

Si le type de la source connectée au récepteur est `DirectorySource`, le récepteur peut utiliser trois variables supplémentaires :

`_FilePath`

Chemin complet du fichier journal.

`_FileName`

Nom et extension de nom du fichier.

`_Position`

Entier qui représente l'emplacement de l'enregistrement dans le fichier journal.

Ces variables sont utiles lorsque vous utilisez une source qui recueille les enregistrements de journaux de plusieurs fichiers connectés à un récepteur qui diffuse tous les enregistrements vers un seul flux. L'injection des valeurs de ces variables dans les enregistrements diffusés permet aux analyses en aval du pipeline de données de classer les enregistrements par fichier et par emplacement au sein de chaque fichier.

Conseils pour écrire des expressions

Une expression peut avoir l'une des expressions suivantes :

- Une expression variable.
- Une expression constante, par exemple,`'hello'`,`1`,`1.21`,`null`,`true`,`false`.
- Expression d'appel qui appelle une fonction, comme illustré dans l'exemple suivant.

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

Caractères spéciaux

Deux barres obliques inverses sont nécessaires pour échapper aux caractères spéciaux.

Nesting

Les invocations de fonctions peuvent être imbriquées, comme illustré dans l'exemple suivant.

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

Variables

Il existe trois types de variables : locale, méta et globale.

- Variables localesCommencez par un$tels que`$message`. Ils sont utilisés pour résoudre la propriété de l'objet d'événement, une entrée si l'événement est un dictionnaire, ou un attribut si l'événement est un objet JSON. Si la variable locale contient de l'espace ou des caractères spéciaux, utilisez une variable locale entre guillemets telle que`$'date created'`.

- Variables MétadonnéesCommencez par un trait de soulignement (_) et sont utilisés pour résoudre les métadonnées de l'événement. Tous les types d'événements prennent en charge les méta variables suivantes.

  `_timestamp`

  Horodatage de l'événement.

  `_record`

  Représentation de texte brut de l'événement.

  Les événements de journal prennent en charge les méta variables supplémentaires suivantes.

  `_filepath`

  `_filename`

  `_position`

  `_linenumber`

- Variables globalesrésoudre en variables d'environnement, métadonnées d'instance EC2 ou EC2Tag. Pour des performances améliorées, nous vous recommandons d'utiliser le préfixe pour limiter la portée de la recherche, comme`{env:ComputerName}`,`{ec2:InstanceId}`, et`{ec2tag:Name}`.

Fonctions intégrées

Kinesis Agent pour Windows prend en charge les fonctions prédéfinies ci-dessous. Si l'un des arguments est`NULL`et la fonction n'est pas conçue pour gérer`NULL`, un`NULL`est retourné.

```
//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)
```

Configuration des substitutions de variables de récepteur

Les déclarations de récepteurs `KinesisStream`, `KinesisFirehose` et `CloudWatchLogs` nécessitent une paire clé-valeur `LogStream` ou `StreamName`. La valeur de ces paires clé-valeur peut contenir des références de variables qui sont automatiquement résolues par l'Agent Kinesis pour Windows. Pour`CloudWatchLogs`, le`LogGroup`paire clé-valeur est également requise et peut contenir des références de variables qui sont automatiquement résolues par l'Agent Kinesis pour Windows. Les variables sont spécifiées en utilisant le modèle {*`prefix`*`:`*`variablename`*}, où *`prefix`*`:` est facultatif. Les préfixes pris en charge sont les suivants :

- `env`— La référence de variable est résolue par la valeur de la variable d'environnement portant le même nom.
- `ec2`— La référence de variable est résolue par les métadonnées d'instance EC2 portant le même nom.

- `ec2tag`— La référence de variable est résolue par la valeur de la balise d'instance EC2 portant le même nom. L'autorisation `ec2:Describe*` est nécessaire pour accéder aux balises d'instance. Pour plus d'informations, consultez Autorisations requises pour l'extension des variables EC2 Tag.

Si le préfixe n'est pas spécifié et s'il y a une variable d'environnement portant le même nom que `variablename`, la référence de variable est résolue par la valeur de la variable d'environnement. Sinon, si `variablename` a pour valeur `instance_id` ou `hostname`, la référence de variable est résolue par la valeur des métadonnées EC2 portant le même nom. Dans le cas contraire, la référence de variable n'est pas résolue.

Voici des exemples de paires clé-valeur valides utilisant des références de variable :

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

Les déclarations de récepteurs `CloudWatchLogs` prennent en charge une variable d'horodatage de format spécial qui autorise l'horodatage de l'enregistrement de journal ou d'événement d'origine à partir de la source pour modifier le nom du flux de journaux. Le format est `{timestamp:timeformat}`. Consultez l'exemple suivant:

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

Si l'enregistrement de journal ou d'événement a été généré le 5 juin 2017, la valeur de la paire clé-valeur `LogStream` de l'exemple précédent sera résolue par `"LogStream_20170605"`.

Si cela est autorisé, le type de récepteur `CloudWatchLogs` peut créer automatiquement de nouveaux flux de journaux lorsque cela est nécessaire en fonction des noms générés. Vous ne pouvez pas le faire pour d'autres types de récepteurs, car ils nécessitent une configuration supplémentaire au-delà du nom du flux.

Il existe des substitutions de variables spéciales qui se produisent dans les éléments TextDecoration et ObjectDecoration. Pour plus d'informations, consultez Configuration des décorations de récepteurs.

Configuration de la mise en file d'attente des récepteurs

Les déclarations de récepteurs `CloudWatchLogs`, `KinesisStream` et `KinesisFirehose` peuvent éventuellement autoriser la mise en file d'attente des enregistrements qui n'ont pas pu être diffusés vers le service AWS associé à ces types de récepteurs en raison de problèmes de connectivité transitoires. Pour activer la mise en file d'attente et l'automatisation des nouvelles tentatives de diffusion en streaming lorsque la connectivité est restaurée, utilisez les paires clé-valeur suivantes dans les déclarations de récepteurs :

`QueueType`

Spécifie le type de mécanisme de mise en file d'attente à utiliser. La seule valeur prise en charge est `file`, ce qui indique que les enregistrements doivent être mis en file d'attente dans un fichier. Cette paire clé-valeur est obligatoire pour activer la fonctionnalité de mise en file d'attente de Kinesis Agent for Windows. Si cet élément n'est pas spécifié, le comportement par défaut est le placement en file d'attente en mémoire uniquement et l'échec de la diffusion lorsque les limites de la mise en file d'attente en mémoire sont atteintes.

`QueuePath`

Spécifie le chemin d'accès au dossier contenant les fichiers des enregistrements placés en file d'attente. La paire clé-valeur est facultative. La valeur par défaut est `%PROGRAMDATA%\KinesisTap\Queue\SinkId`, où SinkId représente l'identifiant que vous avez affecté comme valeur de l'élément `Id` pour la déclaration de récepteurs.

`QueueMaxBatches`

Limite la quantité totale d'espace pouvant être consommée par Kinesis Agent pour Windows lors de la mise en file d'attente des enregistrements pour la diffusion en streaming. La quantité d'espace est limitée à la valeur de cette paire clé-valeur multipliée par le nombre maximal d'octets par lot. Le nombre maximal d'octets par lot des types de récepteurs `CloudWatchLogs`, `KinesisStream` et `KinesisFirehose` est respectivement 5 Mo, 4 Mo et 1 Mo. Lorsque cette limite est atteinte, les défaillances de diffusion en streaming ne sont pas mises en file d'attente et sont signalées comme des défaillances irrécupérables. La paire clé-valeur est facultative. La valeur par défaut est de 10 000 lots.

Configuration d'un proxy pour les récepteurs

Pour configurer un proxy pour tous les types de récepteur Kinesis Agent pour Windows qui accèdent aux services AWS, modifiez le fichier de configuration Kinesis Agent for Windows situé à

l'adresse`%Program Files%\Amazon\KinesisTap\AWSKinesisTap.exe.config`. Pour obtenir des instructions, consultez`proxy`Section dansRéférence aux fichiers de configuration pour le AWS SDK for .NETdans leManuel du développeur du kit SDK AWS pour .NET.

Configuration de la résolution de variables dans d'autres attributs de collecteur

L'exemple suivant illustre une configuration de récepteur qui utilise l'`Region`Variable d'environnement pour la valeur de la stratégie`Region`paire clé/valeur d'attribut. Pour`RoleARN`, il spécifie la clé de balise EC2`MyRoleARN`, qui évalue la valeur associée à cette clé.

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

Configuration des points de terminaison régionaux AWS STS lors de l'utilisation de la propriété RoleARN dans les puits AWS

Cette fonctionnalité ne s'applique que si vous utilisez KinesiStap sur Amazon EC2 et que vous utilisez l'`RoleARN`des puits AWS pour assumer un rôle IAM externe pour s'authentifier auprès des services AWS de destination.

En définissant`UseSTSRegionalEndpoints`sur`true`, vous pouvez spécifier qu'un agent utilise le point de terminaison régional (par exemple,`https://sts.us-east-1.amazonaws.com`) au lieu du point de terminaison global (par exemple,`https://sts.amazonaws.com`). L'utilisation d'un point de terminaison STS régional réduit la latence aller-retour pour l'opération et limite l'impact des défaillances dans le service de point de terminaison global.

Configuration du point de terminaison VPC pour les puits AWS

Vous pouvez spécifier un point de terminaison VPC dans la configuration du récepteur pour`CloudWatchLogs`,`CloudWatch`,`KinesisStreams`, et`KinesisFirehose`types d'évier. Un point de terminaison de VPC permet une connexion privée entre votre VPC et les services AWS pris en charge ou les services de point de terminaison VPC gérés par AWS PrivateLink sans nécessiter une passerelle Internet, un périphérique NAT et une connexion VPN ou une connexion AWS Direct

Connect. Les instances de votre VPC ne requièrent pas d'adresses IP publiques pour communiquer avec les ressources du service. Le trafic entre votre VPC et les autres services ne quitte pas le réseau Amazon. Pour de plus amples informations, veuillez consulterPoints de terminaison d'un VPCdans leGuide de l'utilisateur Amazon VPC.

Vous spécifiez le point de terminaison VPC à l'aide de la`ServiceURL`comme illustré dans l'exemple suivant d'un`CloudWatchLogs`configuration de l'évier. Définissez la valeur de`ServiceURL`à la valeur affichée sur leDétails du point de terminaison du VPCÀ l'aide de la console Amazon VPC.

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

Configuration d'un autre moyen de proxy

Cette fonctionnalité vous permet de configurer un serveur proxy dans une configuration de collecteur à l'aide de la prise en charge du proxy intégrée au kit SDK AWS au lieu de .NET. Auparavant, la seule façon de configurer l'agent pour qu'il utilise un proxy était d'utiliser une fonctionnalité native de .NET, qui acheminait automatiquement toutes les requêtes HTTP/S via le proxy défini dans le fichier proxy.

Si vous utilisez actuellement l'agent avec un serveur proxy, vous n'avez pas besoin de changer pour utiliser cette méthode.

Vous pouvez utiliser la stratégie`ProxyHost`and`ProxyPort`Pour configurer un proxy alternatif, comme illustré dans l'exemple suivant.

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

Déclarations de canal

UtiliserDéclarations de canalpour connecter une source (voirDéclarations de sources) à un évier (voirDéclarations de récepteurs) dans Amazon Kinesis Agent pour Microsoft Windows. Une déclaration de canal est exprimée sous la forme d'un objet JSON. Lorsque Kinesis Agent pour Windows démarre, les journaux, les événements ou les métriques sont collectés à partir de la source d'un canal donné. Ils sont ensuite diffusés vers différents services AWS à l'aide d'un récepteur associé à ce canal.

Voici un exemple de déclaration de canal :

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

Rubriques

- Configuration des canaux
- Configuration de Kinesis Agent pour les canaux métriques Windows

Configuration des canaux

Toutes les déclarations de canal peuvent contenir les paires clé-valeur suivantes :

`Id`

Spécifie le nom du canal (obligatoire). Il doit être unique dans le fichier de configuration.

`Type`

Spécifie le type de transformation (le cas échéant) qui est appliqué par le canal lorsque les données de journaux sont transférées de la source au récepteur. La seule valeur prise en charge est `RegexFilterPipe`. Cette valeur permet de filtrer les expressions régulières de la représentation textuelle sous-jacente de l'enregistrement de journal. L'utilisation du filtrage permet de réduire les coûts de transmission et de stockage par l'envoi des seuls enregistrements de journal pertinents en aval vers le pipeline de données. La paire clé-valeur est facultative. La valeur par défaut consiste à ne fournir aucune transformation.

`FilterPattern`

Spécifie l'expression régulière pour les pipelines `RegexFilterPipe` qui sont utilisés pour filtrer les enregistrements de journal collectés par la source avant le transfert vers le récepteur. Les enregistrements de journal sont transférés par les canaux de type `RegexFilterPipe` lorsque l'expression régulière correspond à la représentation textuelle sous-jacente de l'enregistrement. Les enregistrements de journal structurés qui sont générés, par exemple lors de l'utilisation de la paire clé-valeur `ExtractionPattern` dans une déclaration `DirectorySource`, peuvent toujours être filtrés à l'aide du mécanisme `RegexFilterPipe`. En effet, ce mécanisme agit sur la représentation textuelle d'origine avant de procéder à l'analyse. Cette paire clé-valeur est facultative, mais elle doit être indiquée si le canal spécifie le type `RegexFilterPipe`.

Voici un exemple de déclaration de canal `RegexFilterPipe` :

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

`SourceRef`

Spécifie le nom (la valeur de la paire clé-valeur `Id`) de la déclaration de source qui définit la source qui collecte les données de journaux, d'événements et de métriques pour le canal (obligatoire).

`SinkRef`

Spécifie le nom (la valeur de la paire clé-valeur `Id`) de la déclaration de récepteur qui définit le récepteur qui reçoit les données de journaux, d'événements et de métriques pour le canal (obligatoire).

`IgnoreCase`

Facultatif. Accepte les valeurs de`true`ou`false`. Lorsqu'il est défini sur`true`, l'expression Regex fait correspondre les enregistrements d'une manière insensible à la casse.

`Negate`

Facultatif. Accepte les valeurs `de`true`ou`false. Lorsqu'il est défini sur`true`, le tuyau transmettra les enregistrementsNeCorrespond à l'expression régulière.

Pour obtenir un exemple d'un fichier de configuration complet qui utilise le type de canal `RegexFilterPipe`, consultez Utilisation des canaux.

Configuration de Kinesis Agent pour les canaux métriques Windows

Il existe une source de métrique intégrée nommée`_KinesisTapMetricsSource`qui produit des mesures sur Kinesis Agent pour Windows. S'il y a un`CloudWatch`déclaration de lavabo avec une`IddeMyCloudWatchSink`, l'exemple de déclaration de pipeline suivant transfère les métriques générées par Kinesis Agent pour Windows vers ce récepteur :

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

Pour plus d'informations sur les sources de métriques intégrées à Kinesis Agent pour Windows, consultezSource des métriques prédéfinies de Kinesis Agent pour Windows.

Si le fichier de configuration diffuse également les métriques de compteurs de performances Windows, nous vous conseillons d'utiliser un autre canal et un autre récepteur plutôt que d'utiliser le même récepteur pour les métriques Kinesis Agent pour Windows et les métriques de compteurs de performances Windows.

Configuration des mises à jour automatiques

Utilisation de l'`appsettings.json`Pour permettre la mise à jour automatique d'Amazon Kinesis Agent for Microsoft Windows et du fichier de configuration de Kinesis Agent for Windows. Pour contrôler le comportement de mise à jour, spécifiez la paire clé-valeur `Plugins` au même niveau du fichier de configuration que `Sources`, `Sinks` et `Pipes`.

La paire clé-valeur `Plugins` spécifie les fonctionnalités générales supplémentaires à utiliser et qui ne relèvent pas spécifiquement des catégories sources, récepteurs (sinks) et canaux (pipes). Par exemple, il existe un plug-in pour la mise à jour de Kinesis Agent pour Windows et un plug-in pour

la mise à jour de`appsettings.json`Fichier de configuration. Les plug-ins sont représentés en tant qu'objets JSON et ont toujours une paire clé-valeur `Type`. La paire `Type` détermine les autres paires clé-valeur qui peuvent être spécifiées pour le plug-in. Les types de plug-ins suivants sont pris en charge :

`PackageUpdate`

Spécifie que l'agent Kinesis pour Windows doit vérifier périodiquement le fichier de configuration de la version du package. Si le fichier de la version du package indique qu'une autre version de Kinesis Agent pour Windows doit être installée, Kinesis Agent pour Windows la télécharge et l'installe. Les paires clé-valeur du plug-in `PackageUpdate` incluent :

`Type`

La valeur est obligatoire et doit être la chaîne `PackageUpdate`.

`Interval`

Spécifie en minutes (sous forme de chaîne) la fréquence à laquelle le fichier de la version du package doit être contrôlé en cas de modifications éventuelles. La paire clé-valeur est facultative. Si elle n'est pas spécifiée, la valeur par défaut est 60 minutes. Si la valeur est inférieure à 1, il n'est procédé à aucun contrôle de mise à jour.

`PackageVersion`

Spécifie l'emplacement du fichier JSON de la version du package. Le fichier peut résider sur un partage de fichiers (`file://`), un site web (`http://`) ou Amazon S3 (`s3://`). Par exemple, une valeur de`s3://mycompany/config/agent-package-version.json`indique que Kinesis Agent pour Windows doit vérifier le contenu de la`config/agent-package-version.json`Dans le fichier`mycompany`Compartiment Amazon S3. Il doit effectuer les mises à jour en fonction du contenu de ce fichier.

> **Note**
>
> La valeur de la propriété`PackageVersion`La paire clé-valeur est sensible à la casse pour Amazon S3.

L'exemple suivant est celui du contenu du fichier d'une version de package :

```
{
```

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

La .`Version`Spécifie la version de Kinesis Agent pour Windows à installer si elle n'est pas déjà installée. La référence de la variable `{Version}` dans `PackageUrl` résout la valeur que vous spécifiez pour la paire clé-valeur `Version`. Dans cet exemple, la variable est résolue dans la chaîne `1.0.0.106`. Cette résolution de variable est fournie de telle sorte qu'il y ait un seul emplacement dans le fichier de la version du package où la version spécifique souhaitée est stockée. Vous pouvez utiliser plusieurs fichiers de la version du package pour contrôler le rythme du déploiement de nouvelles versions de Kinesis Agent pour Windows afin de valider une nouvelle version avant un déploiement plus important. Pour restaurer un déploiement de Kinesis Agent pour Windows, modifiez un ou plusieurs fichiers de la version du package afin de spécifier une version antérieure de Kinesis Agent pour Windows connue pour fonctionner dans votre environnement.

La valeur de la paire clé-valeur `PackageVersion` est affectée par la substitution de variable afin de faciliter la sélection automatique des différents fichiers de version du package. Pour plus d'informations sur la substitution de variable, consultez Configuration des substitutions de variables de récepteur.

`AccessKey`

Spécifie la clé d'accès à utiliser lors de l'authentification de l'accès au fichier de la version du package dans Amazon S3. La paire clé-valeur est facultative. Il est déconseillé d'utiliser cette paire clé-valeur. Pour obtenir les autres méthodes d'authentification recommandées, consultez Configuration de l'authentification .

`SecretKey`

Spécifie la clé secrète à utiliser lors de l'authentification de l'accès au fichier de la version du package dans Amazon S3. La paire clé-valeur est facultative. Il est déconseillé d'utiliser cette paire clé-valeur. Pour obtenir les autres méthodes d'authentification recommandées, consultez Configuration de l'authentification .

`Region`

Spécifie le point de terminaison de la région à utiliser lors de l'accès au fichier de la version du package depuis Amazon S3. La paire clé-valeur est facultative.

`ProfileName`

Spécifie le profil de sécurité à utiliser lors de l'authentification de l'accès au fichier de la version du package dans Amazon S3. Pour plus d'informations, consultez Configuration de l'authentification . La paire clé-valeur est facultative.

`RoleARN`

Spécifie le rôle à endosser lors de l'authentification de l'accès au fichier de la version du package dans Amazon S3 dans au sein d'un scénario entre comptes. Pour plus d'informations, consultez Configuration de l'authentification . La paire clé-valeur est facultative.

S'il n'est pas spécifié de plug-in `PackageUpdate`, aucun fichier de la version du package n'est vérifié pour déterminer si une mise à jour est requise.

`ConfigUpdate`

Spécifie que Kinesis Agent pour Windows doit vérifier périodiquement la présence d'un`appsettings.json`Fichier de configuration stocké dans un partage de fichiers, sur un site web ou sur Amazon S3. S'il existe un fichier de configuration mis à jour, il est téléchargé et installé par Kinesis Agent pour Windows.`ConfigUpdate`Les paires clé-valeur incluent les informations suivantes :

`Type`

La valeur est obligatoire et doit être la chaîne `ConfigUpdate`.

`Interval`

Spécifie en minutes (sous la forme d'une chaîne) la fréquence à laquelle un nouveau fichier de configuration doit être recherché. Cette paire clé-valeur est facultative, et si elle n'est pas spécifiée, la valeur par défaut est de 5 minutes. Si la valeur est inférieure à 1, la mise à jour du fichier de configuration n'est pas contrôlée.

`Source`

Spécifie à quel emplacement rechercher un fichier de configuration mis à jour. Le fichier peut résider sur un partage de fichiers (`file://`), un site web (`http://`) ou Amazon S3 (`s3://`). Par exemple, une valeur des`s3://mycompany/config/appsettings.json`indique que Kinesis Agent pour Windows doit rechercher les mises à jour de la`config/appsettings.json`Dans le fichier`mycompany`Compartiment Amazon S3.

> **Note**
>
> La valeur de la propriété`Source`La paire clé-valeur est sensible à la casse pour Amazon S3.

La valeur de la paire clé-valeur `Source` est affectée par la substitution de variable afin de faciliter la sélection automatique des différents fichiers de configuration. Pour plus d'informations sur la substitution de variable, consultez Configuration des substitutions de variables de récepteur.

`Destination`

Spécifie à quel emplacement le fichier de configuration doit être stocké sur l'ordinateur local. Il peut s'agir d'un chemin relatif, d'un chemin absolu ou d'un chemin contenant les références de variables d'environnement telles que `%PROGRAMDATA%`. Si le chemin est relatif, il l'est à l'emplacement où Kinesis Agent pour Windows est installé. Généralement, la valeur est `.\appsettings.json`. Cette paire clé-valeur est requise.

`AccessKey`

Spécifie la clé d'accès à utiliser lors de l'authentification de l'accès au fichier de configuration dans Amazon S3. La paire clé-valeur est facultative. Il est déconseillé d'utiliser cette paire clé-valeur. Pour obtenir les autres méthodes d'authentification recommandées, consultez Configuration de l'authentification .

`SecretKey`

Spécifie la clé secrète à utiliser lors de l'authentification de l'accès au fichier de configuration dans Amazon S3. La paire clé-valeur est facultative. Il est déconseillé d'utiliser cette paire clé-valeur. Pour obtenir les autres méthodes d'authentification recommandées, consultez Configuration de l'authentification .

`Region`

Spécifie le point de terminaison de la région à utiliser lors de l'accès au fichier de configuration à partir d'Amazon S3. La paire clé-valeur est facultative.

`ProfileName`

Spécifie le profil de sécurité à utiliser lors de l'authentification de l'accès au fichier de configuration dans Amazon S3. Pour plus d'informations, consultez Configuration de l'authentification . La paire clé-valeur est facultative.

`RoleARN`

Spécifie le rôle à endosser lors de l'authentification de l'accès au fichier de configuration dans Amazon S3 dans au sein d'un scénario entre comptes. Pour plus d'informations, consultez Configuration de l'authentification . La paire clé-valeur est facultative.

S'il n'est pas spécifié de plug-in `ConfigUpdate`, aucun fichier de configuration n'est vérifié pour déterminer si une mise à jour du fichier de configuration est requise.

L'exemple suivant présente un fichier de configuration `appsettings.json` qui illustre l'utilisation des plug-ins `PackageUpdate` et `ConfigUpdate`. Dans cet exemple, il existe un fichier de la version du package situé dans le`mycompany`Compartiment Amazon S3 nommé`config/agent-package-version.json`. Le fichier est vérifié en quête de modification éventuelle toutes les 2 heures environ. Si une autre version de Kinesis Agent pour Windows est mentionnée dans le fichier, la version de l'agent spécifiée est installée à partir de l'emplacement indiqué dans le fichier de la version du package.

De plus, il existe un`appsettings.json`stocké dans le fichier de configuration`mycompany`Compartiment Amazon S3 nommé`config/appsettings.json`. Environ toutes les 30 minutes, le fichier est comparé au fichier de configuration actuel. S'ils sont différents, le fichier de configuration mis à jour est téléchargé à partir d'Amazon S3 et installé à l'emplacement local standard pour le`appsettings.json`Fichier de configuration.

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

Exemples de configuration de l'agent Kinesis pour Windows

La `.appsettings.json`est un document JSON qui vérifie la manière dont Amazon Kinesis Agent pour Microsoft Windows collecte les journaux, les événements et les métriques. Il vérifie également la manière dont Kinesis Agent for Windows transforme ces données et les diffuse vers différents services AWS. Pour plus d'informations sur les déclarations des sources, des récepteur et des canaux dans le fichier de configuration, consultez Déclarations de sources, Déclarations de récepteurs et Déclarations de canal.

Les sections suivantes proposent des exemples de fichiers de configuration pour différents types de scénarios.

Rubriques

- Diffusion à partir de diverses sources vers les Kinesis Data Streams
- Diffusion à partir du journal des événements d'application Windows vers les récepteurs
- Utilisation des canaux
- Utilisation de plusieurs sources et canaux

Diffusion à partir de diverses sources vers les Kinesis Data Streams

L'exemple suivant`appsettings.json`Les fichiers de configuration illustrent les événements et les journaux de diffusion à partir de diverses sources vers Kinesis Data Streams et à partir des compteurs de performance Windows vers les métriques Amazon CloudWatch.

`DirectorySource`, Analyseur d'enregistrement **`SysLog`**

Le fichier suivant diffuse les enregistrements des journaux au format syslog à partir de tous les fichiers avec`.log`extension de fichier dans le fichier`C:\LogSource\`Accédez au répertoire`SyslogKinesisDataStream`Diffusion Kinesis Data Streams dans la région us-east-1. Un signet est créé pour veiller à ce que toutes les données des fichiers journaux soient envoyées, même si l'agent est arrêté et redémarré ultérieurement. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `SyslogKinesisDataStream`.

```
{
  "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, Analyseur d'enregistrement **SingleLineJson**

Le fichier suivant diffuse les enregistrements des journaux au format JSON à partir de tous les fichiers avec`.log`extension de fichier dans le fichier`C:\LogSource\`Accédez au répertoire`JsonKinesisDataStream`Diffusion Kinesis Data Streams dans la région us-east-1. Avant la diffusion, les paires clé-valeur des clés `ComputerName` et `DT` sont ajoutées à chaque objet JSON, avec les valeurs du nom de l'ordinateur et de la date et de l'heure auxquelles l'enregistrement est traité. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `JsonKinesisDataStream`.

```
{
  "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

Le fichier suivant diffuse les enregistrements des journaux générés par Microsoft Exchange et stockés dans des fichiers avec `.log`extension dans le fichier`C:\temp\ExchangeLog\`Accédez au répertoire`ExchangeKinesisDataStream`Flux de données Kinesis dans la région us-east-1 au format JSON. Bien que les journaux Exchange ne soient pas au format JSON, Kinesis Agent pour Windows peut les analyser et les convertir au format JSON. Avant la diffusion, les paires clé-valeur des clés `ComputerName` et `DT` sont ajoutées à chaque objet JSON contenant les valeurs du nom de l'ordinateur et de la date et de l'heure auxquelles l'enregistrement est traité. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `ExchangeKinesisDataStream`.

```
{
  "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`

Le fichier suivant diffuse les enregistrements des journaux Internet Information Services (IIS) pour Windows stockés à l'emplacement standard de ces fichiers vers le répertoire`IISKinesisDataStream`Diffusion Kinesis Data Streams dans la région us-east-1. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `IISKinesisDataStream`. IIS (Internet Information Services) est un serveur web pour 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` avec requête (Query)

Le fichier suivant transmet les événements du journal des événements système Windows qui ont un niveau de`CriticalouError`(inférieurs ou égaux à 2) au`SystemKinesisDataStream`Flux de

données Kinesis dans la région us-east-1 au format JSON. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `SystemKinesisDataStream`.

```
{
  "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`

Le fichier suivant diffuse les événements de sécurité et d'exception Microsoft Common Language Runtime (CLR) vers le répertoire`ClrKinesisDataStream`Flux de données Kinesis dans la région us-east-1 au format JSON. Une application personnalisée peut lire et traiter les enregistrements à partir du flux `ClrKinesisDataStream`.

```
{
  "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

Le fichier suivant diffuse les compteurs de performance relatifs au nombre total de fichiers ouverts, au nombre total de tentatives de connexion depuis le redémarrage, au nombre de lectures disque par seconde et au pourcentage d'espace disque disponible, vers les métriques CloudWatch de la région us-east-1. Vous pouvez représenter graphiquement ces métriques dans CloudWatch, créer des tableaux de bord à partir des graphiques et définir des alarmes qui envoient des notifications lorsque les seuils sont dépassés.

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

Diffusion à partir du journal des événements d'application Windows vers les récepteurs

L'exemple suivant`appsettings.json`Les fichiers de configuration illustrent la diffusion de journaux d'événements d'application Windows vers différents récepteurs dans Amazon Kinesis Agent for Microsoft Windows. Pour obtenir des exemples d'utilisation des types de récepteur `KinesisStream` et `CloudWatch`, consultez Diffusion à partir de diverses sources vers les Kinesis Data Streams.

KinesisFirehose

Les flux de fichiers suivantsCriticalouErrorLes événements du journal de l'application Windows dans leWindowsLogFirehoseDeliveryStreamDiffusion Kinesis Data Firehose dans la région us-east-1. Si la connexion à Kinesis Data Firehose est interrompue, les événements sont d'abord mis en file d'attente en mémoire. Ensuite, si nécessaire, ils sont mis en file d'attente dans un fichier disque jusqu'à ce que la connexion soit rétablie. Enfin, les événements sont extraits de la file d'attente et envoyés, suivis par les nouveaux événements éventuels.

Vous pouvez configurer Kinesis Data Firehose pour stocker les données diffusées selon différents types de services de stockage et d'analyse en fonction des exigences des canaux de données.

```
{
  "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`

Les flux de fichiers suivants`Critical`ou`Error`Journalisation des événements d'application Windows vers CloudWatch Logs diffuse les flux de diffusion de la page`MyServiceApplicationLog-Group`log group. Le nom de chaque flux commence par `Stream-`. Il se termine par l'année (quatre chiffres), le mois (deux chiffres) et le jour (deux chiffres) de création du flux ; toutes les valeurs sont concaténées (par exemple, `Stream-20180501` représente le flux créé le 1er mai 2018).

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

Utilisation des canaux

L'exemple suivant de fichier de configuration `appsettings.json` illustre l'utilisation des fonctions liées aux canaux.

Cet exemple diffuse les entrées des journaux à partir du fichier`c:\LogSource\`à la`ApplicationLogFirehoseDeliveryStream`Flux de diffusion Kinesis Data Firehose. Il inclut uniquement les lignes qui correspondent à l'expression régulière spécifiée par la paire clé-valeur `FilterPattern`. Plus précisément, seules les lignes du fichier journal qui commencent par`10`ou`11`sont diffusés vers Kinesis Data Firehose.

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

Utilisation de plusieurs sources et canaux

L'exemple suivant de fichier de configuration `appsettings.json` illustre l'utilisation de plusieurs sources et canaux.

Cet exemple diffuse les journaux d'événements d'application, les journaux d'événements de sécurité et les journaux d'événements système Windows vers`EventLogStream`Diffusion Kinesis Data Firehose à l'aide de trois sources, de trois canaux et d'un seul récepteur.

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

Configuration de la télémétrie

Pour assurer une meilleure prise en charge, Amazon Kinesis Agent pour Microsoft Windows collecte par défaut des statistiques sur le fonctionnement de l'agent et les envoie à AWS. Ces données ne comportent pas d'informations personnelles identifiables. Elles n'incluent aucune donnée collectée depuis les services AWS ou diffusée vers ces services. Nous recueillons environ 1 à 2 Ko de ces données métriques toutes les 60 minutes.

Vous pouvez choisir de désactiver la collecte et la transmission de ces statistiques. Pour cela, ajoutez la paire clé-valeur suivante au fichier de configuration `appsettings.json` au même niveau que les sources, récepteurs et canaux :

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

Par exemple, le fichier de configuration suivant configure une source, un récepteur et un canal, et désactive également la télémétrie :

```
{
  "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"
    }
}
```

Nous collectons les métriques suivantes lorsque la télémétrie est activée :

`ClientId`

ID unique affecté automatiquement lorsque le logiciel est installé.

`ClientTimestamp`

Date et heure de collecte des données télémétriques.

`OSDescription`

Description du système d'exploitation.

`DotnetFramework`

Version de l'infrastructure dotnet actuelle.

`MemoryUsage`

Quantité de mémoire utilisée par Kinesis Agent pour Windows (en Mo).

`CPUUsage`

Quantité d'Kinesis C (pourcentage au format décimal). Par exemple, 0,01 équivaut à 1 %.

`InstanceId`

ID d'instance Amazon EC2 si Kinesis Agent pour Windows est exécuté sur une instance Amazon EC2.

`InstanceType (string)`

Type d'instance Amazon EC2 si Kinesis Agent pour Windows s'exécute sur une instance Amazon EC2.

En outre, nous collectons les métriques indiquées dans Liste des mesures Kinesis Agent pour Windows.

Didacticiel : Diffuser les fichiers journaux JSON vers Amazon S3 à l'aide de Kinesis Agent pour Windows

Ce didacticiel présente les étapes détaillées pour configurer un pipeline de données à l'aide d'Amazon Kinesis Agent pour Microsoft Windows (Kinesis Agent for Windows).

Le didacticiel comprend les étapes suivantes :

- Utilisation de Kinesis Agent for Windows pour diffuser des fichiers journaux au format JSON versAmazon Simple Storage Service (Amazon S3)viaAmazon Kinesis Data Firehose. Pour plus d'informations sur Kinesis Agent pour Windows, consultezPrésentation de Amazon Kinesis Agent pour Microsoft Windows ?.
- Amélioration des données de journal avant la diffusion à l'aide de la décoration d'objet. Pour plus d'informations, consultez Configuration des décorations de récepteurs.
- UtiliserAmazon AthenaPour rechercher certains types d'enregistrements de journaux.

Prerequisites

Si vous ne possédez pas de compte AWS, suivez les instructions deConfiguration d'un compte AWSpour en obtenir un.

Rubriques

Étape 1 : Configuration d'AWS

Suivez les étapes ci-dessous pour préparer votre environnement à la diffusion de données de journaux vers Amazon Simple Storage Service (Amazon S3) à l'aide d'Amazon Kinesis Agent pour Microsoft Windows. Pour plus d'informations et pour connaître les prérequis, consultez Didacticiel : Diffuser des fichiers journaux JSON vers Amazon S3.

Utilisez AWS Management Console pour configurer AWS Identity and Access Management (IAM), Amazon S3, Kinesis Data Firehose et Amazon Elastic Compute Cloud (Amazon EC2) afin de préparer la diffusion en continu des données du journal à partir d'une instance EC2 vers Amazon S3.

Rubriques

Configuration de stratégies et de rôles IAM

Créez la stratégie suivante, qui autorise Kinesis Agent pour Windows à diffuser des enregistrements vers un flux de diffusion Kinesis Data Firehose :

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

Remplacez*region*Avec le nom de la région AWS dans laquelle le flux de diffusion Kinesis Data Firehose sera créé (`us-east-1`, par exemple). Remplacez *account-id* par l'ID de compte à 12 chiffres du compte AWS dans lequel le flux de diffusion sera créé.

Dans la barre de navigation, choisissezSupport, puisCentre de support. Votre numéro de compte (ID) à 12 chiffres actuellement connecté apparaît dans leCentre de supportVolet de navigation.

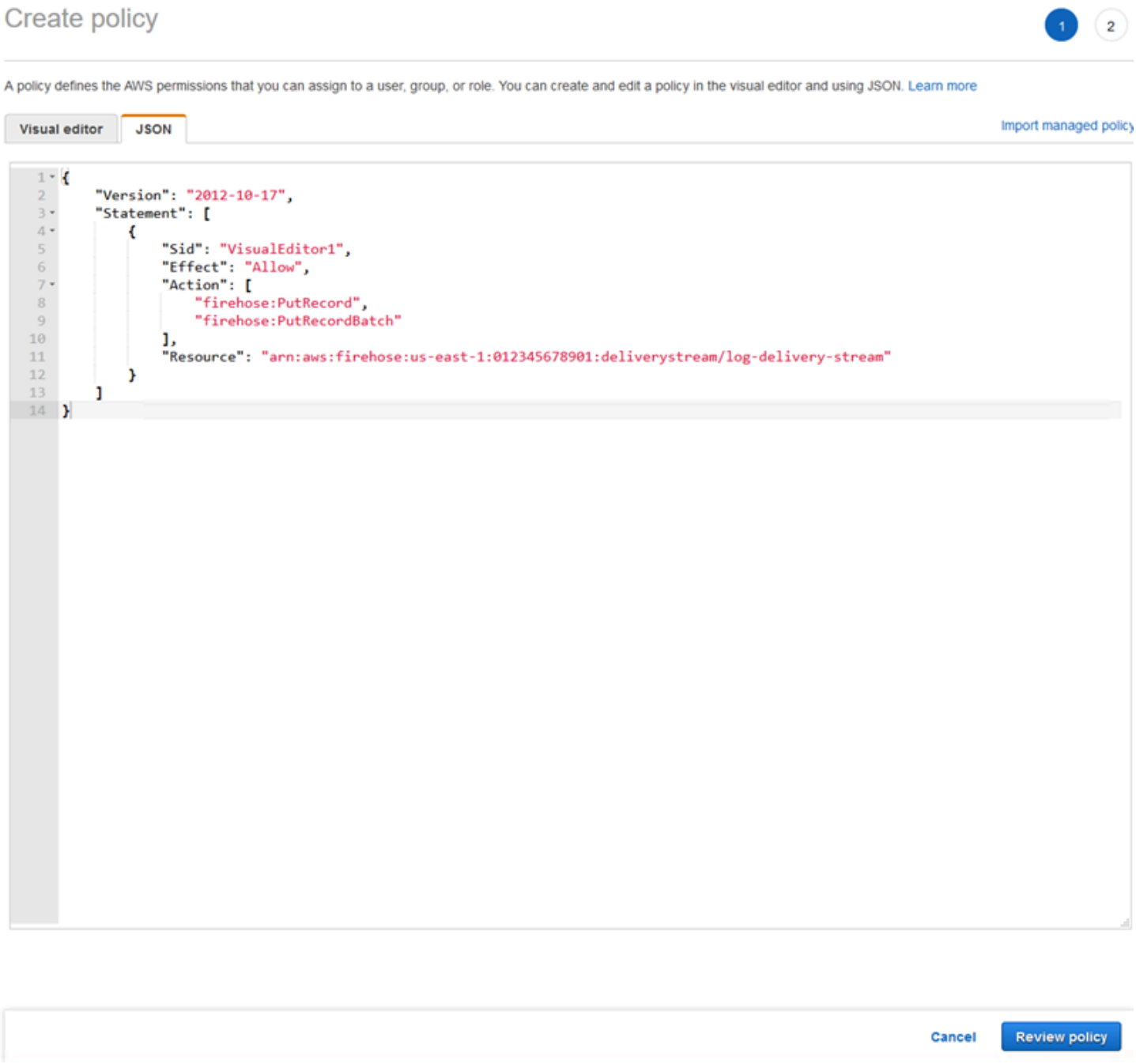
Créez la stratégie à l'aide de la procédure suivante. Nommez la stratégie `log-delivery-stream-access-policy`.

Pour créer une stratégie à l'aide de l'éditeur de stratégie JSON

1. Connectez-vous à AWS Management Console et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Stratégies.

   Si vous choisissez Stratégies pour la première fois, la page Bienvenue dans les stratégies gérées s'affiche. Sélectionnez Get Started.
3. En haut de la page, sélectionnez Créer une stratégie.
4. Choisissez l'onglet JSON.
5. Entrez un document de stratégie JSON. Pour plus d'informations sur le langage de stratégie IAM, consultezRéférence de stratégie JSON IAMdans leGuide de l'utilisateur IAM.
6. Lorsque vous avez terminé, choisissez Examiner une stratégie. Le programme de validation de stratégie signale les éventuelles erreurs de syntaxe.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les onglets Éditeur visuel et JSON. Toutefois, si vous apportez des modifications ou choisissezExaminer une stratégiedans leVisual editor (Éditeur visuel), IAM peut restructurer votre stratégie afin de l'optimiser pour l'éditeur visuel. Pour de plus amples informations, veuillez consulterRestructuration de stratégiedans leGuide de l'utilisateur IAM.
7. Dans la page Examiner une stratégie, entrez un nom et une description (facultatif) pour la stratégie que vous êtes en train de créer. Vérifiez le récapitulatif de stratégie pour voir les autorisations accordées par votre stratégie. Choisissez ensuite Créer une stratégie pour enregistrer votre travail.



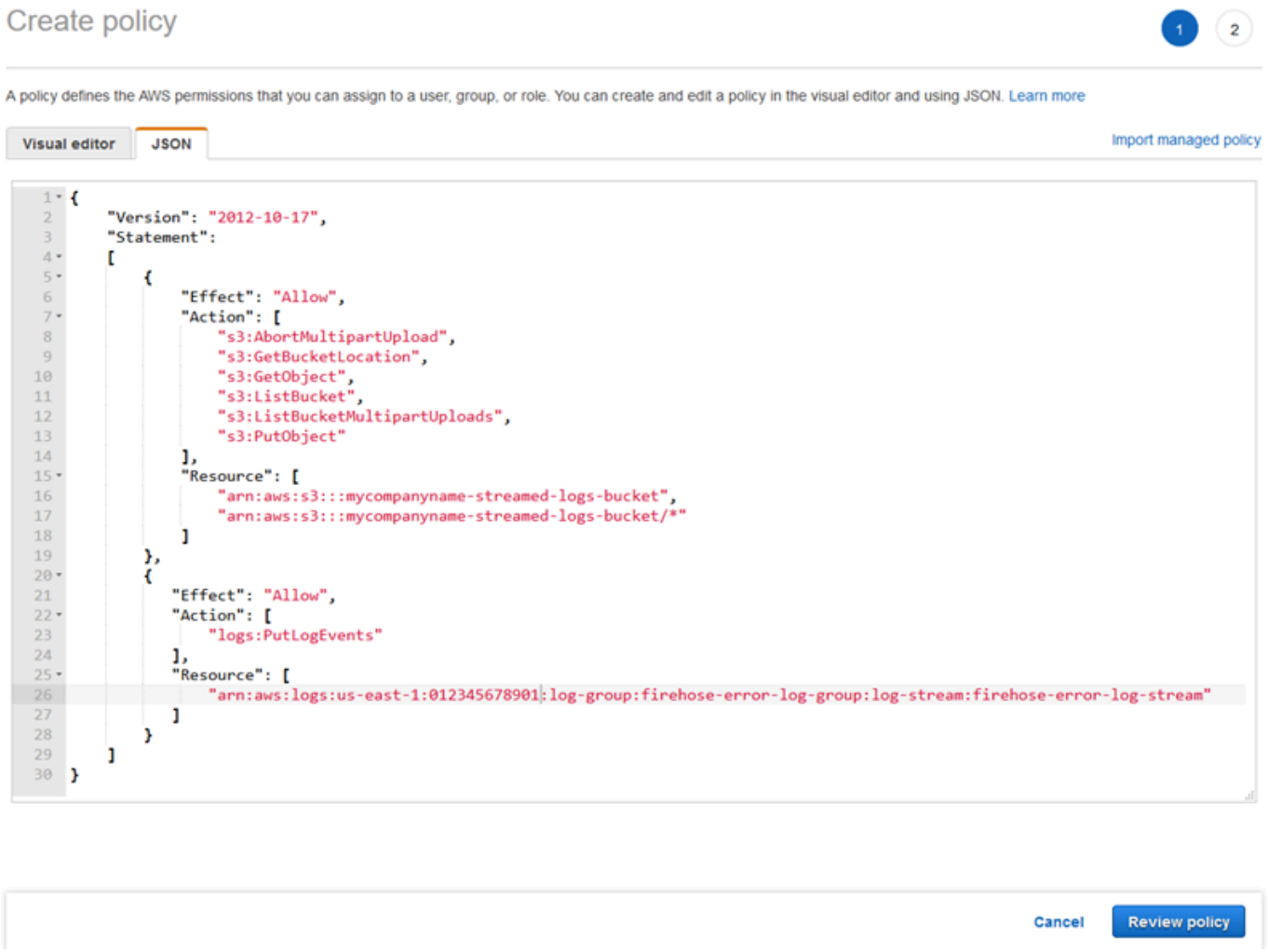
Pour créer le rôle qui accorde à Kinesis Data Firehose à l'accès à un compartiment S3

1. À l'aide de la procédure précédente, créez une stratégie nommée `firehose-s3-access-policy` qui est définie à l'aide du code JSON suivant :

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

Remplacez *bucket-name* par un nom de compartiment unique dans lequel les journaux seront stockés. Remplacez*region*Avec la région AWS dans laquelle le groupe de journaux et le flux de journaux CloudWatch seront créés. Ces derniers contiendront les erreurs qui se produisent au cours de la diffusion de données vers Amazon S3 via Kinesis Data Firehose. Remplacez *account-id* par l'ID de compte à 12 chiffres pour le compte dans lequel le groupe de journaux et le flux de journaux seront créés.



2. Dans le volet de navigation de la console IAM, choisissez Rôles, puis Créer un rôle.
3. Cliquez sur l'ongletService AWS, puis sélectionnez leKinesisservice.
4. ChoisissezKinesis Data FirehosePour le cas d'utilisation, puis sélectionnezSuivant: Autorisations.
5. Dans la zone de recherche, entrez**`firehose-s3-access-policy`**, choisissez cette stratégie, puis sélectionnezSuivant: Vérification.
6. Dans la zone Nom du rôle, entrez **`firehose-s3-access-role`**.
7. Sélectionnez Créer un rôle.

Pour créer le rôle à associer au profil d'instance pour l'instance EC2 qui exécutera Kinesis Agent pour Windows

1. Dans le volet de navigation de la console IAM, choisissez Rôles, puis Créer un rôle.
2. Cliquez sur l'ongletService AWSType de rôle, puis sélectionnezEC2.

3. ChoisissezSuivant: Autorisations.
4. Dans la zone de recherche, entrez **`log-delivery-stream-access-policy`**.
5. Choisissez la stratégie, puis sélectionnezSuivant: Vérification.
6. Dans la zone Nom du rôle, entrez **`kinesis-agent-instance-role`**.
7. Sélectionnez Créer un rôle.

Créer le compartiment Amazon S3

Créez le compartiment S3 dans lequel Kinesis Data Firehose diffuse les journaux.

Pour créer le compartiment S3 pour le stockage des journaux

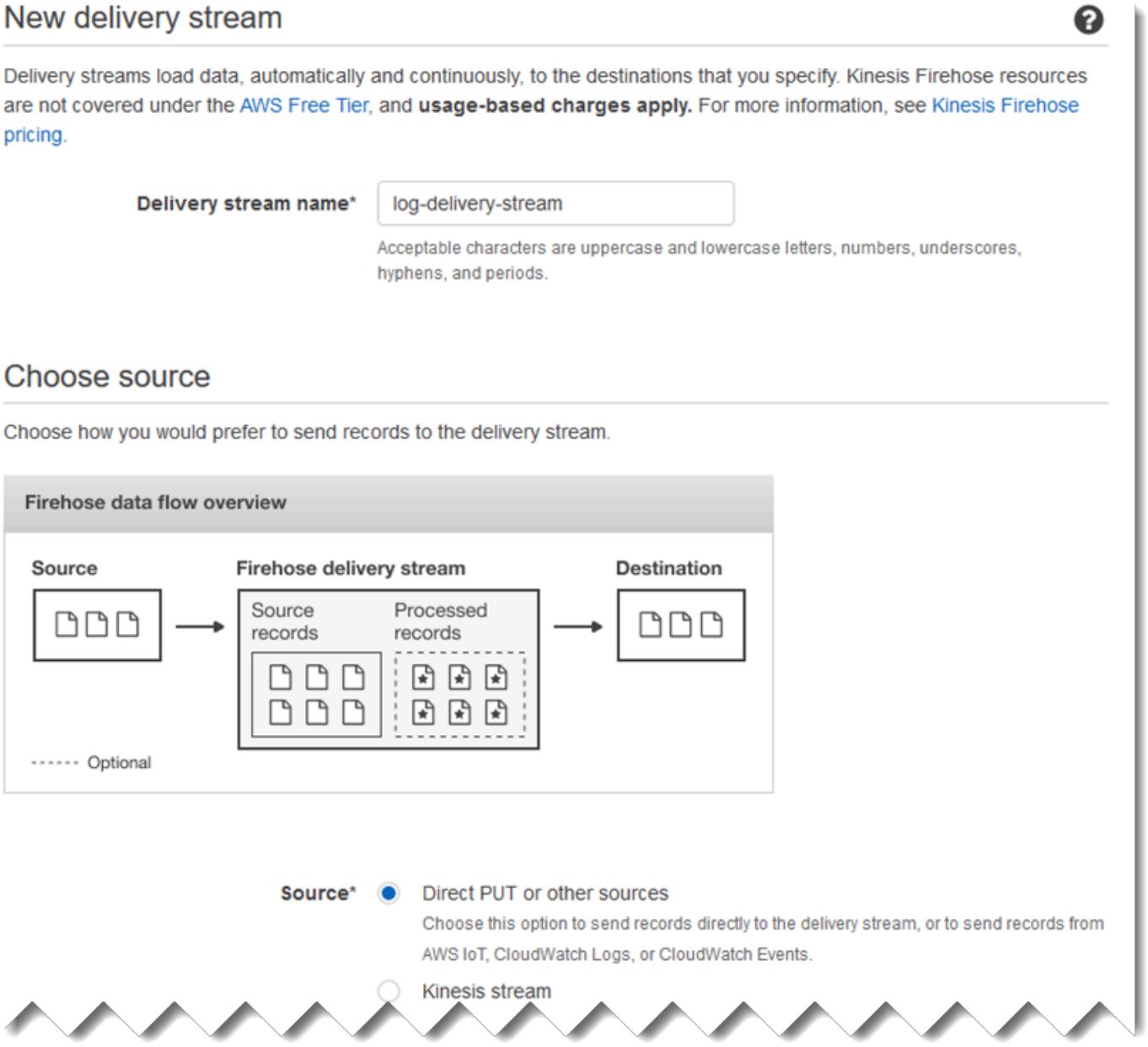
1. Ouvrez la console Amazon S3 à l'adresse https://console.aws.amazon.com/s3/.
2. Choisissez Créer un compartiment.
3. Dans la zone Nom du compartiment, entrez le nom unique du compartiment S3 que vous avez choisi lors de l'étape Configuration de stratégies et de rôles IAM.
4. Choisissez la région dans laquelle le compartiment doit être créé. Il s'agit généralement de la région dans laquelle vous prévoyez de créer le flux de diffusion Kinesis Data Firehose et l'instance Amazon EC2.
5. Sélectionnez Créer.

Création du flux de diffusion Kinesis Data Firehose

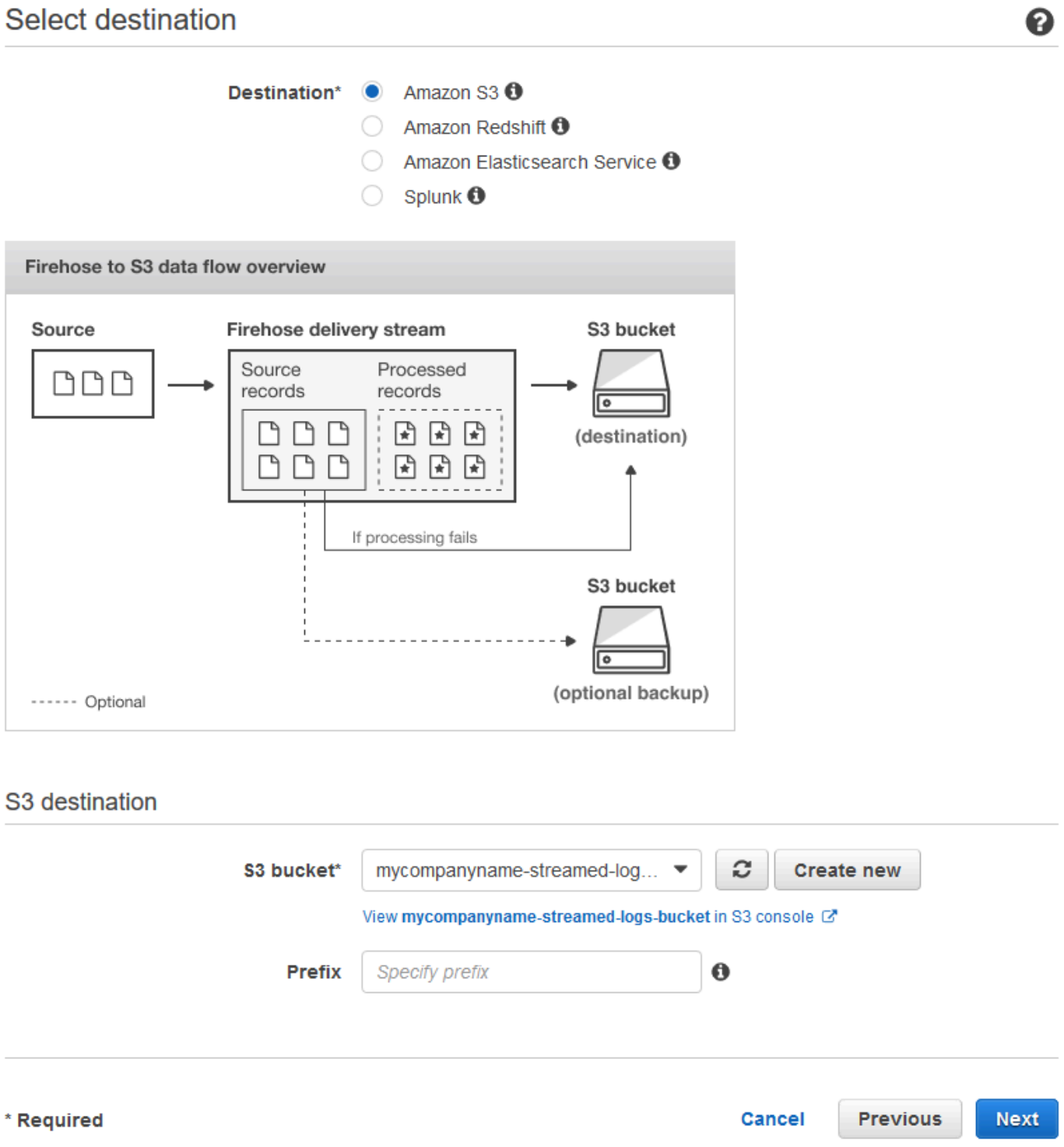
Créez le flux de diffusion Kinesis Data Firehose qui stockera les enregistrements diffusés dans Amazon S3.

Pour créer le flux de diffusion Kinesis Data Firehose

1. Ouvrez la console Kinesis Data Firehose surhttps://console.aws.amazon.com/firehose/.
2. Sélectionnez Create Delivery Stream (Créer un flux de diffusion).
3. Dans la zone de texte Nom du flux de diffusion, entrez **`log-delivery-stream`**.
4. Pour le champ Source, choisissez Instruction PUT directe ou autres sources.

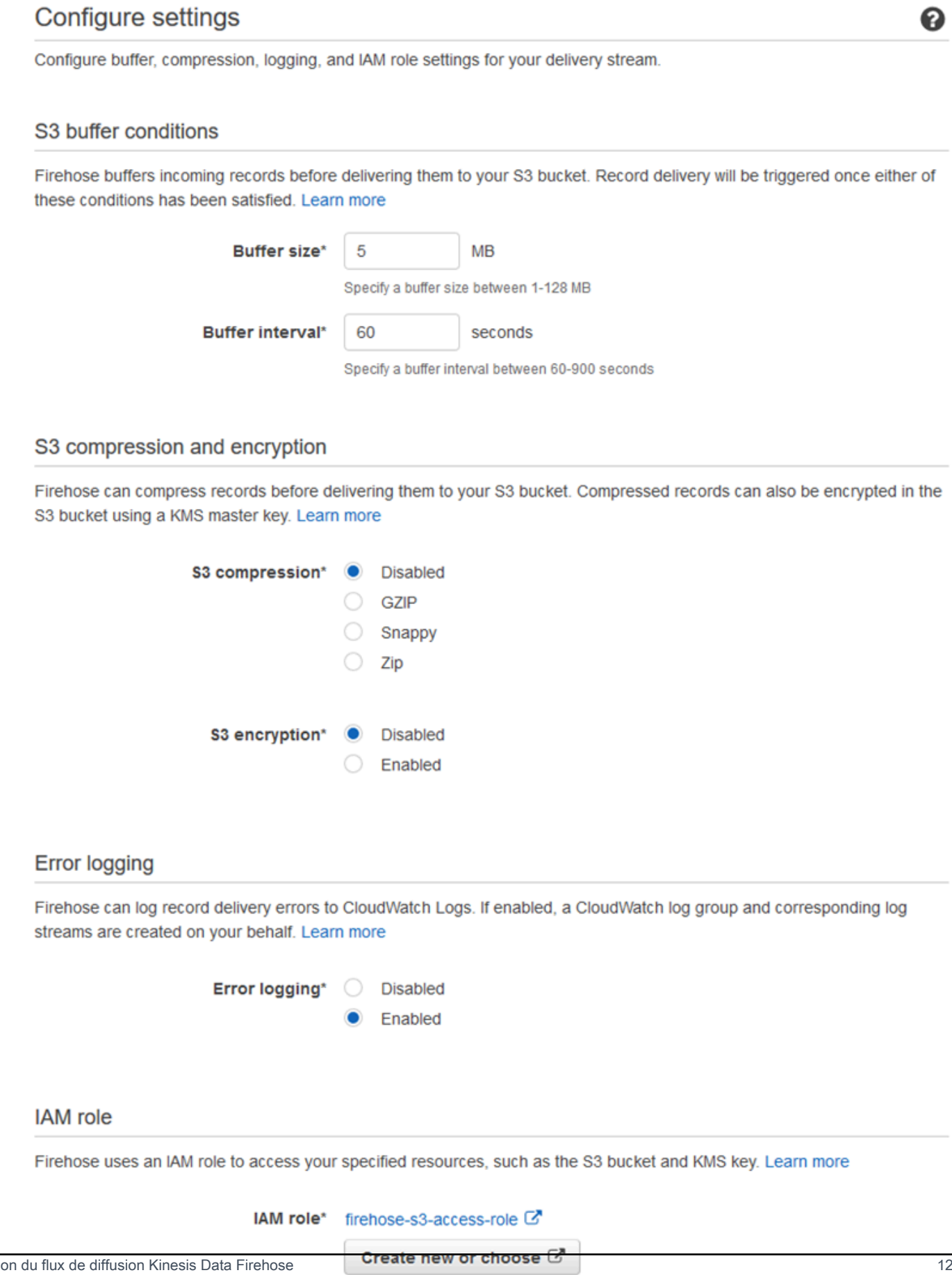


5. Choisissez Suivant.
6. Choisissez Suivant à nouveau.
7. Pour la destination, choisissezAmazon S3.
8. Pour leCompartiment S3, choisissez le nom du compartiment que vous avez créé dansCréer le compartiment Amazon S3.



9. Choisissez Suivant.
10. Dans la zone Intervalle de mise en mémoire tampon entrez **`60`**.
11. Sour Rôle IAM, choisissez Créer ou choisir.
12. Pour Rôle IAM, choisissez `firehose-s3-access-role`.

13. Sélectionnez Allow (Autoriser).


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. Choisissez Suivant.
15. Sélectionnez Create delivery Stream (Créer un flux de diffusion).

Créer l'instance Amazon EC2 pour exécuter Kinesis Agent pour Windows

Créez l'instance EC2 qui utilise Kinesis Agent pour Windows pour diffuser des enregistrements de journaux via Kinesis Data Firehose.

Pour créer l'instance EC2

1. Ouvrez la console Amazon EC2 sur https://console.aws.amazon.com/ec2/.
2. Suivez les instructions de Premiers pas avec les instances Windows Amazon EC2, ainsi que les étapes supplémentaires suivantes :
   - Pour leRôle IAMpour l'instance, choisissez`kinesis-agent-instance-role`.
   - Si vous ne disposez pas déjà d'un VPC connecté à l'Internet public, suivez les instructions deConfiguration d'Amazon EC2dans leGuide de l'utilisateur Amazon EC2 pour les instances Windows.
   - Créez ou utilisez un groupe de sécurité qui restreint l'accès à l'instance à partir de votre ordinateur uniquement ou des ordinateurs de votre organisation uniquement. Pour de plus amples informations, veuillez consulterConfiguration d'Amazon EC2dans leGuide de l'utilisateur Amazon EC2 pour les instances Windows.
   - Si vous spécifiez une paire de clés existante, assurez-vous d'avoir accès à la clé privée de la paire de clés. Ou créez une nouvelle paire de clés et enregistrez la clé privée en lieu sûr.
   - Avant de poursuivre, patientez jusqu'à ce que l'instance soit en cours d'exécution et qu'elle ait exécuté deux vérifications de l'état sur deux.
   - Votre instance nécessite une adresse IP publique. Si aucune n'a été allouée, suivez les instructions deAdresses IP Elasticdans leGuide de l'utilisateur Amazon EC2 pour les instances Windows.

Étapes suivantes

Étape 2 : Installer, configurer et exécuter Kinesis Agent pour Windows

Étape 2 : Installer, configurer et exécuter Kinesis Agent pour Windows

Au cours de cette étape, vous utilisez AWS Management Console pour vous connecter à distance à l'instance que vous avez lancée dansCréer l'instance Amazon EC2 pour exécuter Kinesis Agent pour Windows. Vous installez ensuite Amazon Kinesis Agent pour Microsoft Windows sur l'instance, créez et déployez le fichier de configuration pour Kinesis Agent pour Windows, puis démarrez leAWSKineSISTAPservice.

1. Connectez-vous à distance à l'instance via le protocole RDP en suivant les instructions deÉtape 2 : Connectez-vous à votre instancedans leGuide de l'utilisateur Amazon EC2 pour les instances Windows.
2. Sur l'instance, utilisez le gestionnaire Windows Server pour désactiver la configuration de sécurité renforcée d'Internet Explorer pour les utilisateurs et les administrateurs. Pour plus d'informations, consultez How To Turn Off Internet Explorer Enhanced Security Configuration sur le site web Microsoft TechNet.
3. Sur l'instance, installez et configurez Kinesis Agent pour Windows. Pour plus d'informations, consultez Installation de Kinesis Agent pour Windows.
4. Sur l'instance, utilisez le Bloc-Notes pour créer un fichier de configuration Kinesis Agent for Windows. Enregistrez le fichier dans `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`. Ajoutez le contenu suivant au fichier de configuration :

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

Ce fichier configure Kinesis Agent pour Windows afin qu'il envoie des enregistrements de journal au format JSON à partir de fichiers duc:\logsource\(le répertoiresource) vers un flux de diffusion Kinesis Data Firehose nommélog-delivery-stream(lasink). Avant qu'un enregistrement de journal soit diffusé dans Kinesis Data Firehose, il est renforcé avec deux paires clé-valeur supplémentaires qui contiennent le nom de l'ordinateur et un horodatage.

5. Créez le répertoire `c:\LogSource\`, puis utilisez le Bloc-Notes pour créer un fichier `test.log` dans ce répertoire avec le contenu suivant :

```
{ "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. Dans une session PowerShell de niveau élevé, utilisez la commande suivante pour démarrer le service AWSKinesisTap :

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

7. À l'aide de l'Explorateur de fichiers, accédez au répertoire `%PROGRAMDATA%\Amazon \AWSKinesisTap\logs`. Ouvrez le fichier journal le plus récent. Le fichier journal doit se présenter comme suit :

```
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.
```

Ce fichier journal indique que le service a démarré et que les enregistrements de journaux sont en cours de collecte dans le répertoire `c:\LogSource\`. Chaque ligne est analysée en tant qu'objet JSON unique. Des paires clé-valeur correspondant au nom de l'ordinateur et à l'horodatage sont ajoutées à chaque objet. Ensuite, il est diffusé sur Kinesis Data Firehose.

8. Dans une ou deux minutes, accédez au compartiment Amazon S3 que vous avez créé dansCréer le compartiment Amazon S3Utilisation d'AWS Management Console. Assurez-vous d'avoir sélectionné la bonne région sur la console.

   Ce compartiment comporte un dossier pour l'année en cours. Ouvrez ce dossier pour afficher le dossier pour le mois en cours. Ouvrez ce dossier pour afficher le dossier pour le jour en cours. Ouvrez ce dossier pour afficher le dossier pour l'heure en cours (en heure UTC). Ouvrez ce dossier pour afficher un ou plusieurs éléments commençant par le nom `log-delivery-stream`.

9. Ouvrez le contenu du dernier élément pour confirmer que les enregistrements de journaux ont bien été stockés dans Amazon S3 avec les améliorations souhaitées. Si tout est configuré correctement, le contenu est similaire à ce qui suit :

```
{"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. Pour plus d'informations sur la résolution des problèmes suivants, consultez Dépannage d'Amazon Kinesis Agent pour Microsoft Windows :

    - Le fichier journal Kinesis Agent pour Windows contient des erreurs.
    - Les dossiers ou éléments attendus dans Amazon S3 n'existent pas.
    - Le contenu d'un article Amazon S3 n'est pas correct.

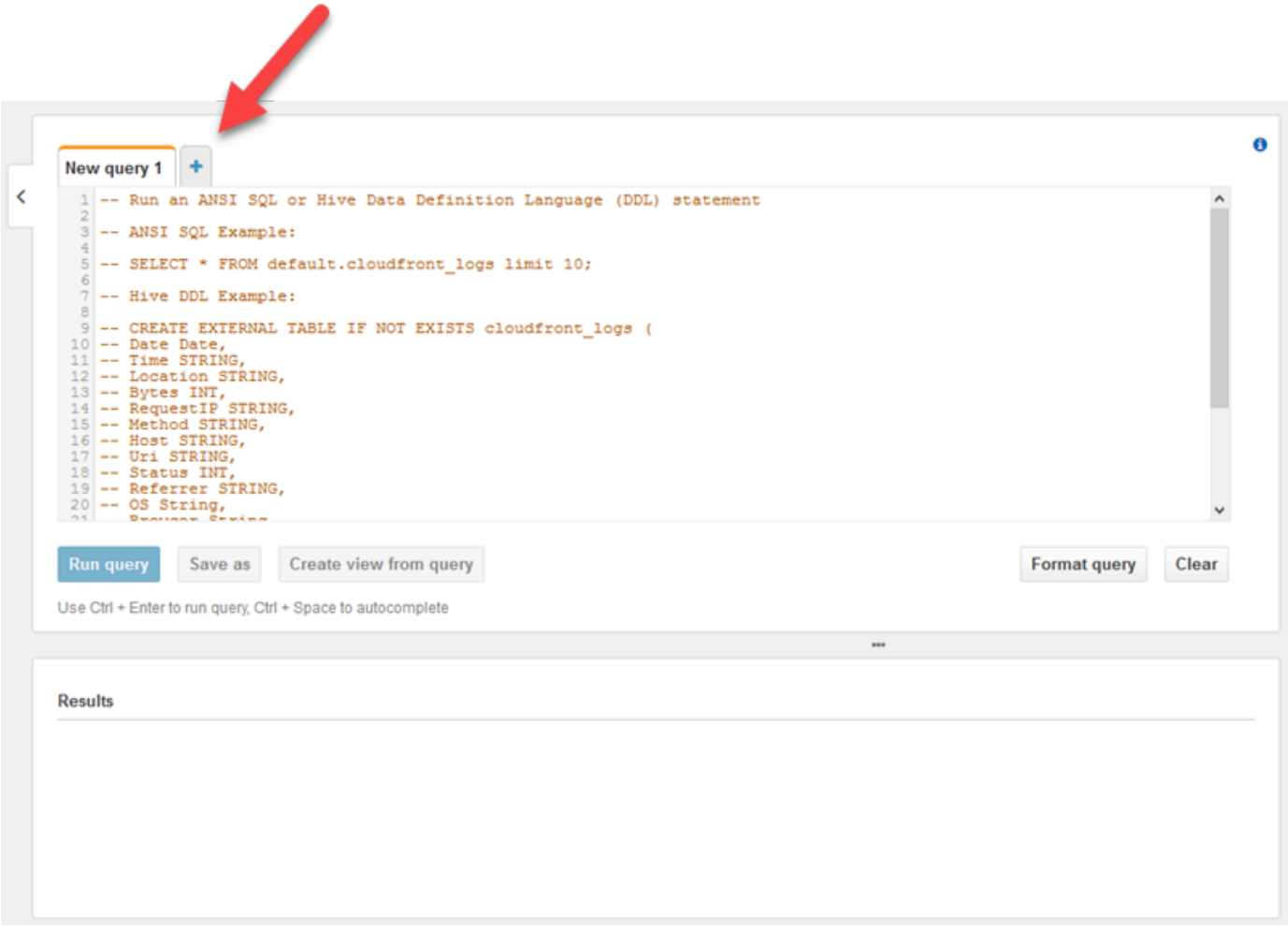
Étapes suivantes

Étape 3 : Interroger les données de journal dans Amazon S3

Étape 3 : Interroger les données de journal dans Amazon S3

Dans la dernière étape de cet agent Amazon Kinesis pour Microsoft WindowsDidacticiel, vous utilisez Amazon Athena pour interroger les données de journaux stockées dans Amazon Simple Storage Service (Amazon S3).

1. Ouvrez la console Athena à l'adresse https://console.aws.amazon.com/athena/.
2. Choisissez le signe plus (+) dans la fenêtre de requête Athena pour créer une nouvelle fenêtre de requête.



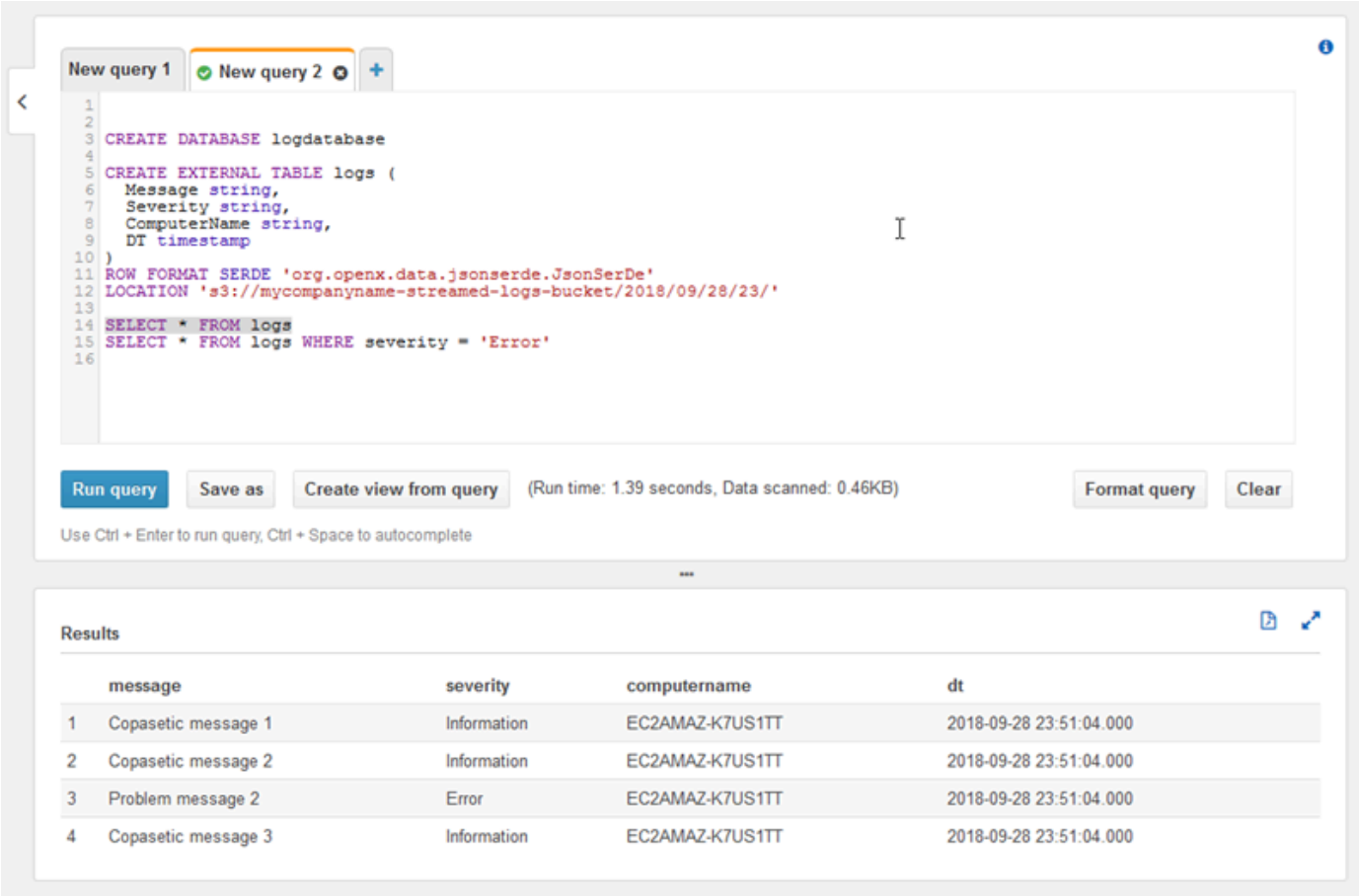
3. Saisissez le texte suivant dans la fenêtre de requête :

```
CREATE DATABASE logdatabase

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

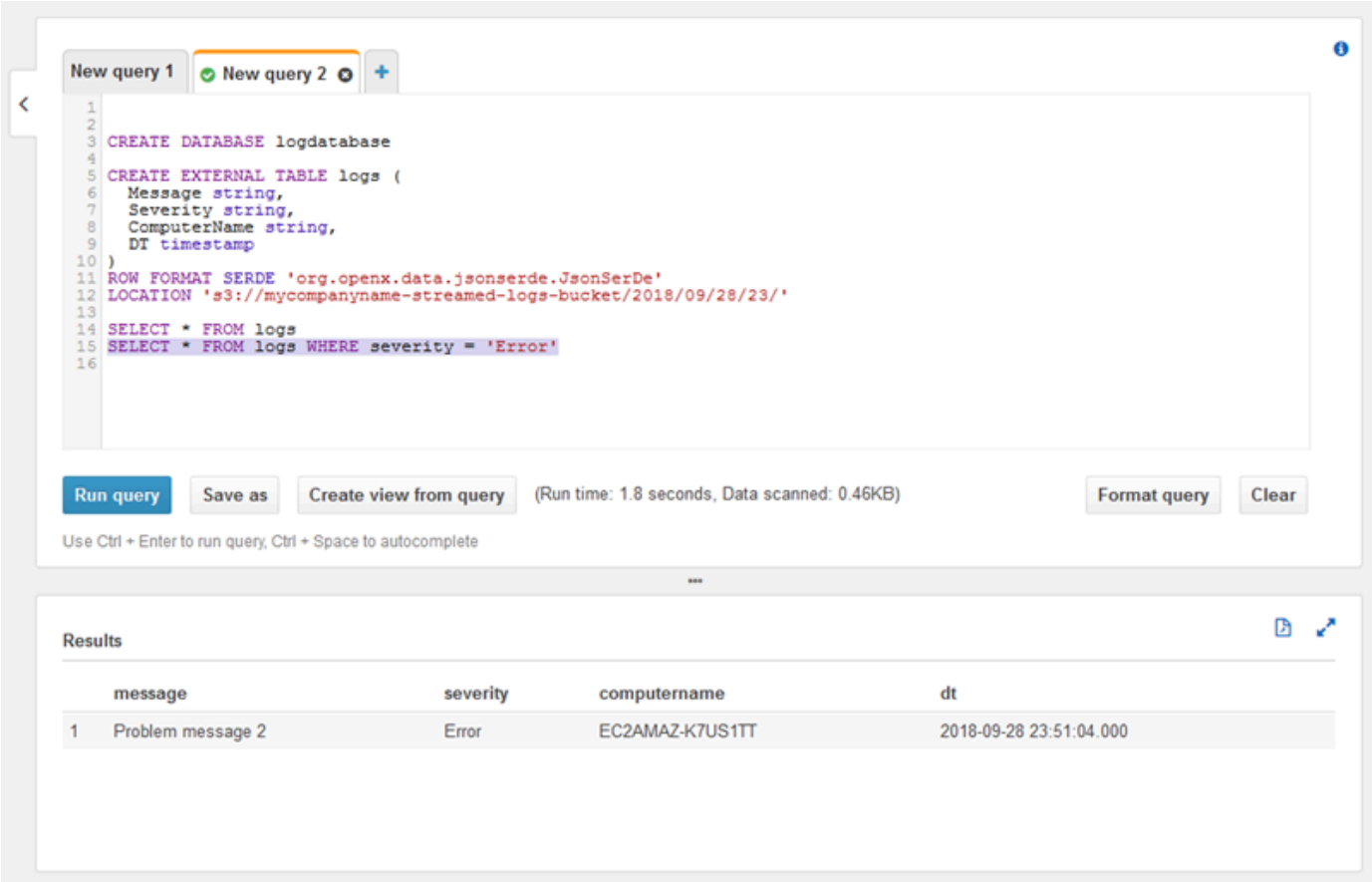
```
  ComputerName string,
  DT timestamp
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://bucket/year/month/day/hour/'

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

Remplacez *bucket* par le nom du compartiment que vous avez créé dans Créer le compartiment Amazon S3. Remplacez*year*,*month*,*day*and*hour*Par l'année, le mois, le jour et l'heure de création du fichier journal Amazon S3, en heure UTC.

4. Sélectionnez le texte pour l'instruction `CREATE DATABASE`, puis choisissez Exécuter la requête. La base de données des journaux est ainsi créée dans Athena.
5. Sélectionnez le texte pour l'instruction `CREATE EXTERNAL TABLE`, puis choisissez Exécuter la requête. Une table Athena est créée. Elle référence le compartiment S3 avec les données de journaux, ce qui mappe le schéma du fichier JSON au schéma de la table Athena.
6. Sélectionnez le texte pour la première instruction `SELECT`, puis choisissez Exécuter la requête. Cette action permet d'afficher toutes les lignes de la table.



7. Sélectionnez le texte pour la seconde instruction `SELECT`, puis choisissez Exécuter la requête. Cette action permet d'afficher uniquement les lignes de la table qui représentent les enregistrements de journaux dont le niveau de gravité est `Error`. Ce type de requête identifie les enregistrements de journaux intéressants à partir d'un ensemble d'enregistrements de journaux potentiellement vaste.



Étapes suivantes

Utilisez AWS Management Console pour nettoyer les ressources créées au cours de ce didacticiel :

1. Résiliez l'instance EC2 (voir l'étape 3 dans Premiers pas avec les instances Windows Amazon EC2).

   > **Important**
   >
   > Si vous avez lancé une instance qui ne se trouvait pas dans leNiveau gratuit d'AWS, vous recevrez une facture pour l'instance jusqu'à ce que vous la mettiez hors service.

2. Supprimez le flux de diffusion Kinesis Data Firehose.

   a. Ouvrez la console Kinesis Data Firehose surhttps://console.aws.amazon.com/firehose/.

   b. Choisissez le flux de diffusion que vous avez créé.

   c. Sélectionnez Delete.

d. Choisissez Supprimer un flux de diffusion.

3. Supprimez le compartiment S3. Pour obtenir des instructions, consultezComment supprimer un compartiment S3 ?dans leManuel de l'utilisateur Amazon Simple Storage Service Console.

Pour plus d'informations, consultez les rubriques suivantes :

- Configuration d'Amazon Kinesis Agent pour Microsoft Windows
- Présentation de Amazon Kinesis Data Firehose
- En quoi consiste Amazon S3 ?
- Qu'est-ce que Amazon Athena ?

Dépannage d'Amazon Kinesis Agent pour Microsoft Windows

Utilisez les instructions suivantes pour diagnostiquer et corriger les problèmes liés à l'utilisation d'Amazon Kinesis Agent pour Microsoft Windows.

Rubriques

Aucune donnée n'est diffusée à partir d'ordinateurs de bureaux ou de serveurs vers les services AWS attendus

Symptoms

Lorsque vous examinez les journaux, les événements et les métriques hébergés par différents services AWS qui sont configurés pour recevoir des flux de données de Kinesis Agent for Windows, aucune donnée n'est diffusée par Kinesis Agent for Windows.

Causes

Plusieurs causes possibles peuvent entraîner ce problème :

- Une source, un récepteur ou un canal n'est pas configuré correctement.
- L'authentification pour Kinesis Agent pour Windows n'est pas configurée correctement.
- L'autorisation pour Kinesis Agent pour Windows n'est pas configurée correctement.
- Une expression régulière fournie dans une déclaration `DirectorySource` comporte une erreur.
- Un répertoire qui n'existe pas est spécifié pour une déclaration `DirectorySource`.

- Des valeurs non valides sont fournies à des services AWS, qui rejette ainsi les demandes provenant de Kinesis Agent pour Windows.
- Un récepteur référence une ressource qui n'existe pas dans la région AWS spécifiée ou implicite.
- Une requête non valide est spécifiée pour une déclaration `WindowsEventLogSource`.
- Une valeur non valide est spécifiée pour la paire clé-valeur `InitialPosition` pour une source.
- Le fichier de configuration `appsettings.json` n'est pas conforme au schéma JSON pour ce fichier.
- Les données sont diffusées vers une région différente de celle sélectionnée dans AWS Management Console.
- Kinesis Agent pour Windows n'est pas installé correctement ou n'est pas en cours d'exécution.

Resolutions

Pour résoudre les problèmes de données non diffusées, procédez de la manière suivante :

1. Examinez les journaux Kinesis Agent for Windows dans la`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`Répertoire. Recherchez la chaîne `ERROR`.
   a. Si le chargement d'une source ou d'un récepteur n'a pas eu lieu, procédez comme suit :
      i. Examinez le message d'erreur et identifiez la valeur `Id` de la source ou du récepteur.
      ii. Vérifiez que la déclaration de source ou de récepteur correspondant à cet `Id` dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` afin d'identifier les erreurs liées au message d'erreur trouvé. Pour en savoir plus, consultez Configuration d'Amazon Kinesis Agent pour Microsoft Windows.
      iii. Corrigez les problèmes du fichier de configuration liés à l'erreur.
      iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.
   b. Si le message d'erreur indique qu'une référence `SourceRef` ou `SinkRef` n'a pas été trouvée pour un canal, procédez comme suit :
      i. Notez l'`Id` du canal.
      ii. Examinez la déclaration de canal dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` qui correspond à la valeur `Id` indiquée. Assurez-vous que les valeurs des paires clé-valeur `SourceRef` et `SinkRef` sont des `Id` correctement orthographiées pour les déclarations de source et de récepteur que vous avez prévu de référencer. Corrigez les fautes d'orthographe ou de frappe. Si une déclaration de

source ou de récepteur ne figure pas dans le fichier de configuration, ajoutez-la. Pour plus d'informations, consultez Configuration d'Amazon Kinesis Agent pour Microsoft Windows.

iii. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

c. Si le message d'erreur indique qu'un utilisateur ou un rôle IAM n'est pas autorisé à effectuer certaines opérations, procédez comme suit :

i. Vérifiez que l'utilisateur ou le rôle IAM utilisé par Kinesis Agent pour Windows est correct. Si ce n'est pas le cas, passez en revueConfiguration de la sécurité des récepteurset modifiez l'authentification de Kinesis Agent pour Windows pour vous assurer que l'utilisateur ou le rôle IAM utilisé est correct.

ii. Si l'utilisateur ou le rôle IAM utilisé est correct, utilisez AWS examiner les stratégies qui sont associées à l'utilisateur ou au rôle. Assurez-vous que l'utilisateur ou le rôle dispose de toutes les autorisations mentionnées dans le message d'erreur pour toutes les ressources AWS auxquelles accède Kinesis Agent for Windows. Pour plus d'informations, consultez Configuration de l'autorisation.

iii. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vérifier que les problèmes de sécurité ont été résolus.

d. Si le message d'erreur indique une erreur d'argument lors de l'analyse d'une expression régulière qui se trouve dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`, procédez comme suit :

i. Examinez l'expression régulière dans le fichier de configuration.

ii. Vérifiez la syntaxe de l'expression régulière. Plusieurs sites web peuvent vous permettre de vérifier les expressions régulières. Vous pouvez également utiliser les lignes de commande suivantes pour vérifier les expressions régulières pour une déclaration de source `DirectorySource` :

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

Remplacez *`sourceId`* par la valeur de la paire clé-valeur `Id` de la déclaration de source `DirectorySource` comportant une expression régulière incorrecte.

iii. Apportez toutes les corrections nécessaires à l'expression régulière dans le fichier de configuration afin qu'il devienne valide.

iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

e. Si le message d'erreur indique une erreur d'argument lors de l'analyse d'une expression régulière qui ne se trouve pas dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` et qui est liée à un récepteur particulier, procédez comme suit :

   i. Recherchez la déclaration de récepteur dans le fichier de configuration.

   ii. Vérifiez que les paires clé-valeur qui sont spécifiquement liées à un service AWS utilisent des noms conformes aux règles de validation pour ce service. Par exemple, les noms des groupes CloudWatch Logs doivent uniquement contenir un ensemble de caractères spécifié à l'aide de l'expression régulière`[\.\-_/#A-Za-z0-9]+`.

   iii. Corrigez les noms non valides dans les paires clé-valeur pour la déclaration de récepteur, et assurez-vous que ces ressources sont correctement configurées dans AWS.

   iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

f. Si le message d'erreur indique que le chargement d'une source ou d'un récepteur échoue en raison d'un paramètre manquant ou null, procédez comme suit :

   i. Notez la valeur `Id` de la source ou du récepteur.

   ii. Recherchez la déclaration de source ou de récepteur correspondant à la valeur `Id` relevée dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

   iii. Vérifiez les paires clé-valeur qui sont fournies dans la déclaration de source ou de récepteur par rapport aux exigences de type de récepteur dans la documentation Configuration d'Amazon Kinesis Agent pour Microsoft Windows du type de récepteur concerné. Ajoutez les paires clé-valeur manquantes à la déclaration de source ou de récepteur.

   iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

g. Si le message d'erreur indique qu'un nom de répertoire n'est pas valide, procédez comme suit :

   i. Localisez le nom de répertoire non valide dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

   ii. Vérifiez que ce répertoire existe et contient les fichiers journaux qui doivent être diffusés.

   iii. Corrigez les fautes de frappe ou les erreurs dans le nom du répertoire spécifié dans le fichier de configuration.

   iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

h. Si le message d'erreur indique qu'une ressource n'existe pas :

   i. Localisez la référence de ressource pour la ressource qui n'existe pas dans une déclaration de récepteur dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

   ii. Utilisez AWS Management Console pour localiser la ressource dans la région AWS devant être utilisée dans la déclaration de récepteur. Comparez-la aux éléments spécifiés dans le fichier de configuration.

   iii. Modifiez la déclaration de récepteur dans le fichier de configuration afin que le nom de la ressource et la région soient corrects.

   iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

i. Si le message d'erreur indique qu'une requête n'est pas valide pour un `WindowsEventLogSource` spécifique, procédez comme suit :

   i. Dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`, localisez la déclaration `WindowsEventLogSource` ayant le même `Id` que dans le message d'erreur.

   ii. Vérifiez que la valeur de la paire clé-valeur `Query` dans la déclaration de source correspond aux informations indiquées dans Event queries and Event XML.

   iii. Apportez toutes les modifications nécessaires pour que la requête soit conforme.

   iv. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

j. Si le message d'erreur indique une position initiale non valide, procédez comme suit :

   i. Dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`, localisez la déclaration de source ayant le même `Id` que dans le message d'erreur.

   ii. Modifiez la valeur de la paire clé-valeur `InitialPosition` dans la déclaration de source de manière à ce qu'elle respecte les valeurs autorisées, comme décrit dans Configuration des signets.

   iii. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

2. Vérifiez que le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` est conforme au schéma JSON.

   a. Dans une fenêtre d'invite de commande, appelez les lignes suivantes :

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

b. Corrigez les problèmes détectés avec le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.

c. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite le fichier journal le plus récent afin de vous assurer que les problèmes de configuration ont été résolus.

3. Modifiez le niveau de journalisation pour essayer d'obtenir des informations de journalisation plus détaillées.

a. Remplacez le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\nlog.xml` par le contenu suivant :

```
<?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. Arrêtez et démarrez le service `AWSKinesisTap`. Vérifiez ensuite les fichiers journaux les plus récents pour vérifier la présence de messages supplémentaires dans le journal, susceptibles d'aider à diagnostiquer et résoudre le problème.

4. Vérifiez que vous consultez les ressources dans la bonne région dans AWS Management Console.
5. Vérifiez que l'agent Kinesis Agent pour Windows est installé et en cours d'exécution.
   a. Dans Windows, choisissez Démarrer, puis accédez à Panneau de configuration, Outils d'administration, Services.
   b. Recherchez le service AWSKinesisTap.
   c. Si le service AWSKinesisTap n'est pas visible, installez Kinesis Agent pour Windows en suivant les instructions deDémarrez avec l'agent Amazon Kinesis.
   d. Si le service est visible, déterminez s'il est en cours d'exécution. Si ce n'est pas le cas, ouvrez le menu contextuel (clic droit) pour le service et choisissez Démarrer.
   e. Vérifiez que le service a démarré en examinant le dernier fichier journal dans le répertoire `%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`.

S'applique à

Cette information s'applique à Kinesis Agent pour Windows version 1.0.0.0.0.0.115 et aux versions ultérieures.

Des données prévues peuvent être manquantes

Symptoms

Kinesis Agent pour Windows diffuse correctement les données la plupart du temps, mais, occasionnellement, certaines données sont manquantes.

Causes

Plusieurs causes possibles peuvent entraîner ce problème :

- La fonctionnalité de signets n'est pas utilisée.
- Les limites de débit de données pour les services AWS sont dépassées par rapport à la configuration actuelle de ces services.
- Les limites de taux d'appels d'API pour les services AWS sont dépassées par rapport à la`appsettings.json`et les limites de compte AWS.

Resolutions

Pour résoudre les problèmes de données manquantes, procédez comme suit :

1. Envisagez d'utiliser la fonctionnalité de signets documentée dans Configuration des signets. Elle permet de s'assurer que toutes les données sont finalement envoyées, même lorsque Kinesis Agent pour Windows s'arrête et démarre.
2. Utilisez les mesures intégrées de Kinesis Agent pour Windows pour détecter les problèmes :
   a. Activez la diffusion des métriques Kinesis Agent pour Windows, comme décrit dansConfiguration de Kinesis Agent pour les canaux métriques Windows.
   b. Si le nombre d'erreurs non récupérables est significatif pour un ou plusieurs récepteurs, déterminez le nombre d'octets ou d'enregistrements envoyés par seconde. Déterminez ensuite si ce nombre respecte la limite configurée pour ces services AWS dans la région et le compte dans lesquels les données sont diffusées.
   c. En cas de dépassement des limites, réduisez le débit ou la quantité de données envoyées, demandez une augmentation des limites ou augmentez le partitionnement, le cas échéant.
   d. Après les ajustements, continuez à surveiller les métriques intégrées de l'Agent Kinesis pour Windows afin de vous assurer que la situation est résolue.

Pour de plus amples informations sur les limites Kinesis Data Streams, consultezLimites Kinesis Data Streamsdans leKinesis Data Streams - Guide du développeur. Pour de plus amples informations sur les limites Kinesis Data Firehose, veuillez consulterLimites d'Amazon Kinesis Data Firehose.

S'applique à

Cette information s'applique à Kinesis Agent pour Windows version 1.0.0.0.0.0.115 et aux versions ultérieures.

Le format des données reçues est incorrect

Symptoms

Les données arrivent dans le service AWS dans un format incorrect.

Causes

Plusieurs causes possibles peuvent entraîner ce problème :

- La valeur de la paire clé-valeur `Format` d'une déclaration de récepteur dans le fichier de configuration `appsettings.json` est incorrecte.
- La valeur de la paire clé-valeur `RecordParser` dans une déclaration `DirectorySource` est incorrecte.
- Les expressions régulières dans une déclaration `DirectorySource` qui utilise l'analyseur d'enregistrement `Regex` sont incorrectes.

Resolutions

Pour résoudre les problèmes de mise en forme incorrecte, procédez comme suit :

1. Vérifiez les déclarations de récepteur dans le fichier de configuration `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json`.
2. Assurez-vous que la valeur correcte de la paire clé-valeur `Format` est spécifiée pour chaque déclaration de récepteur. Pour plus d'informations, consultez Déclarations de récepteurs.
3. Si les sources avec des déclarations `DirectorySource` sont connectées par des canaux aux récepteurs qui spécifient des valeurs `xml` ou `json` pour la paire clé-valeur `Format`, assurez-vous que ces sources spécifient l'une des valeurs suivantes pour la paire clé-valeur `RecordParser` :

- `SingleLineJson`
- `Regex`
- `SysLog`
- `Delimited`

D'autres analyseurs d'enregistrements sont à base de texte uniquement et ne fonctionnent pas correctement avec les récepteurs nécessitant un format XML ou JSON.

4. Si des enregistrements de journaux ne sont pas correctement analysés par le type de source `DirectorySource`, appelez les lignes suivantes dans une fenêtre d'invite de commande afin de vérifier les paires clé-valeur d'horodatage et d'expression régulière spécifiées dans la déclaration `DirectorySource` :

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

Remplacez *`sourceID`* par la valeur de la paire clé-valeur `Id` de la `DirectorySource` déclaration de source qui ne semble pas fonctionner correctement. Corrigez les éventuels problèmes signalés par `ktdiag.exe`.

S'applique à

Cette information s'applique à Kinesis Agent pour Windows version 1.0.0.0.0.0.115 et aux versions ultérieures.

Problèmes de performance

Symptoms

Les latences des applications et des services ont augmenté depuis l'installation et le démarrage de Kinesis Agent pour Windows.

Causes

Plusieurs causes possibles peuvent entraîner ce problème :

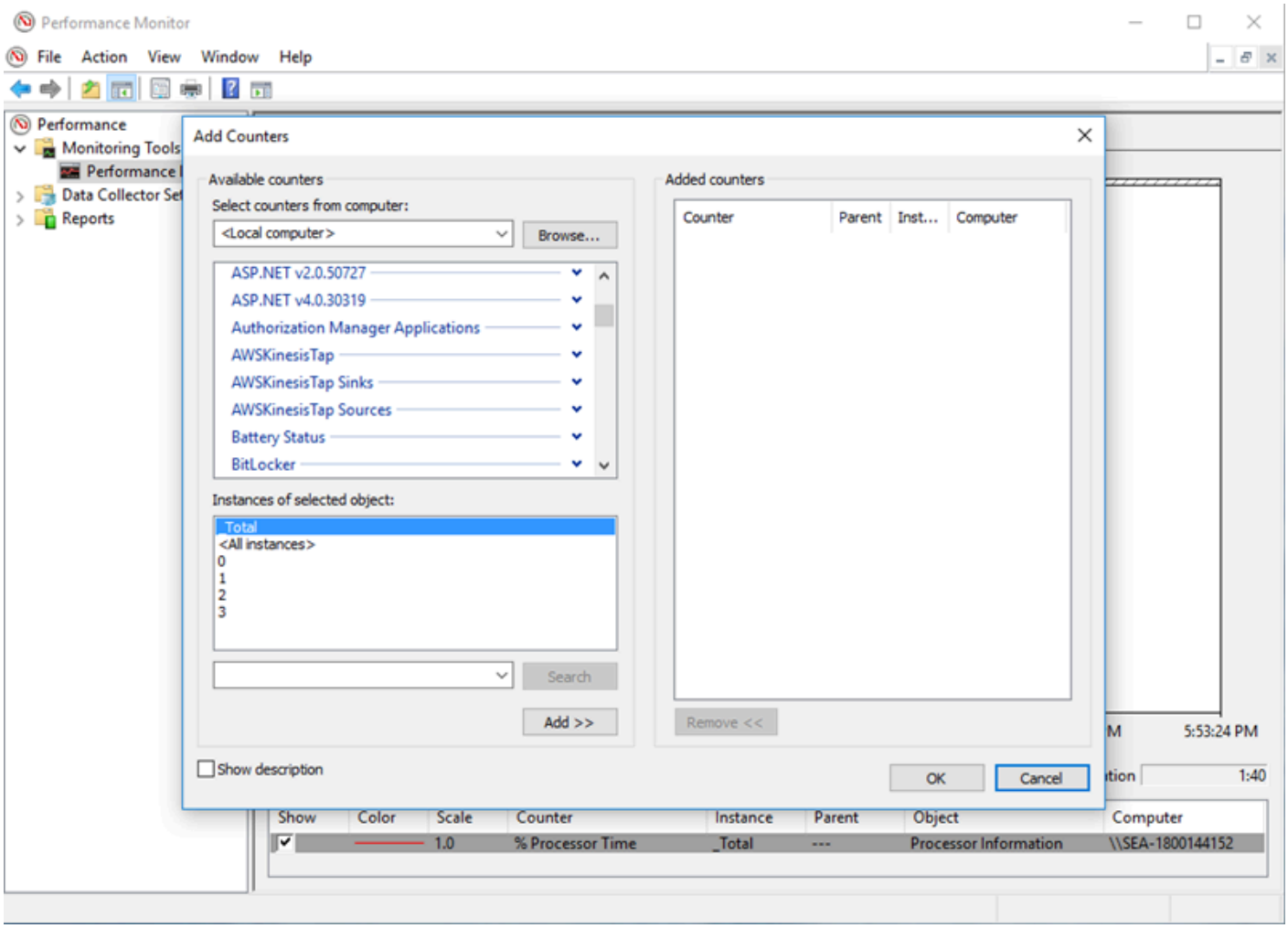
- La capacité de l'ordinateur dans lequel s'exécute Kinesis Agent pour Windows n'a pas une capacité suffisante pour diffuser la quantité de données souhaitée.

- Des données inutiles sont diffusées vers un ou plusieurs services AWS.
- Kinesis Agent pour Windows diffuse des données vers des services AWS qui ne sont pas configurés pour un débit de données aussi élevé.
- Kinesis Agent pour Windows appelle des opérations sur des services AWS dans un compte dans lequel la limite de débit d'appels d'API est trop faible.

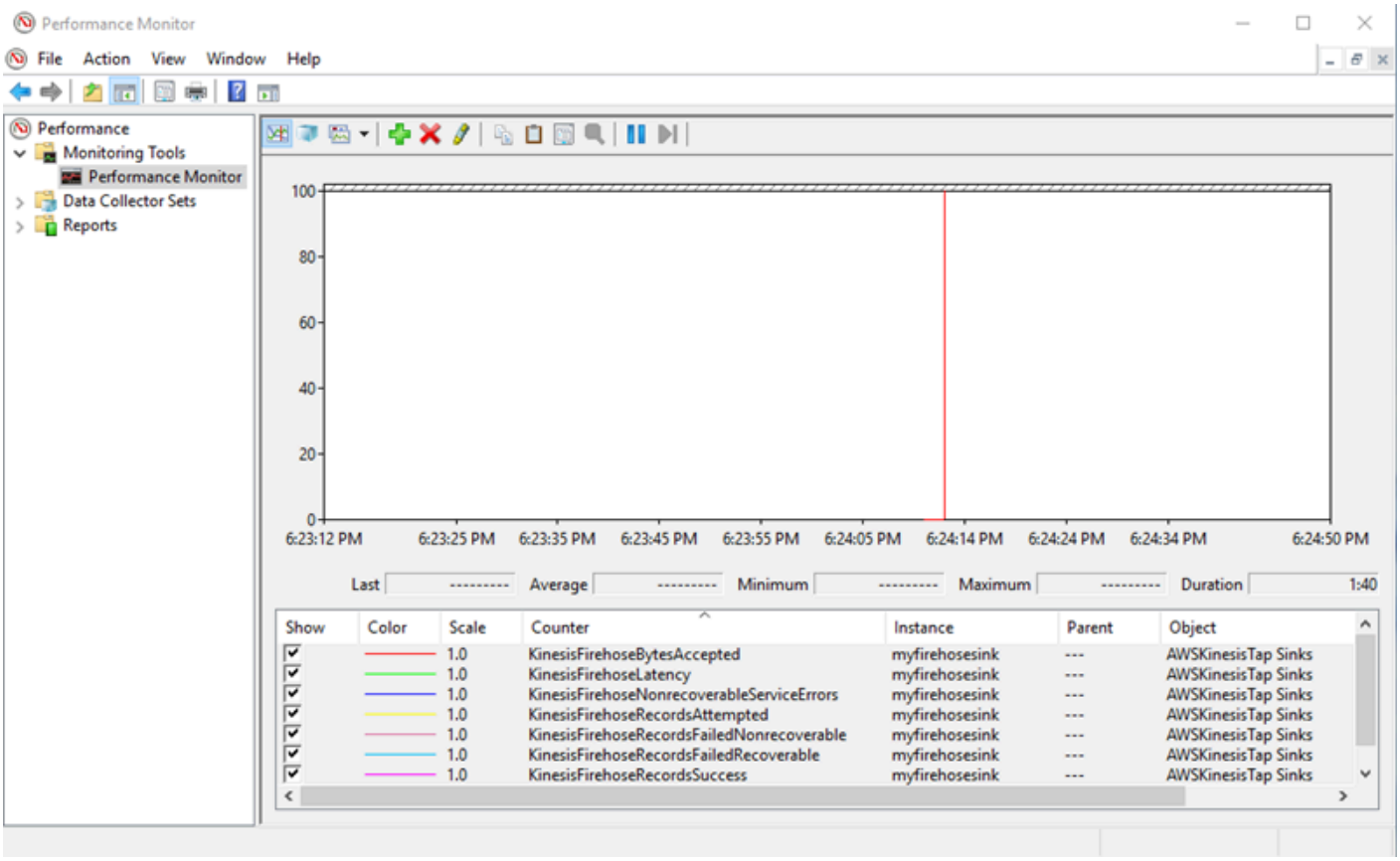
Resolutions

Pour résoudre les problèmes de performance, procédez comme suit :

1. Utilisez l'application de surveillance des ressources Windows pour vérifier l'utilisation de la mémoire, de l'UC, du disque et du réseau. Si vous avez besoin de diffuser de grandes quantités de données avec Kinesis Agent for Windows, vous aurez peut-être besoin de mettre en service une machine dotée de plus grandes capacités dans certains de ces domaines, en fonction de la configuration.
2. Le filtrage peut vous aider à réduire la quantité de données consignées :
   - Consultez les informations relatives à la paire clé-valeur `Query` dans Configuration de WindowsEventLogSource.
   - Consultez les informations relatives au filtrage de pipeline dans Configuration des canaux.
   - Consultez le filtrage des métriques Amazon CloudWatch dansConfiguration du récepteur CloudWatch Sink).
3. Utilisez l'application de surveillance des performances Windows pour afficher les métriques Kinesis Agent for Windows ou diffuser ces métriques vers CloudWatch (consultezSource des métriques prédéfinies de Kinesis Agent pour Windows). Dans l'application de surveillance des performances Windows, vous pouvez ajouter des compteurs pour Kinesis Agent pour les récepteurs et les sources Windows. Ils sont répertoriés sous les catégories de compteurs AWSKinesisTap Sinks (Récepteurs AWSKinesisTap) et AWSKinesisTap Sources (Sources AWSKinesisTap).



Par exemple, pour diagnostiquer les problèmes de performances Kinesis Data Firehose, ajoutez l'Kinesis FirehoseCompteurs de performance.



Si le nombre d'erreurs récupérables est important, consultez les derniers journaux Kinesis Agent pour Windows dans`%PROGRAMDATA%\Amazon\AWSKInesisTap\logs`Répertoire. Si une limitation se produit pour les récepteurs `KinesisStream` ou `KinesisFirehose`, procédez comme suit :

- Si la limitation est due à une diffusion trop rapide des données, envisagez d'augmenter le nombre de partitions pour le flux de données Kinesis. Pour de plus amples informations, veuillez consulterRepartitionnement, mise à l'échelle et traitement parallèledans leKinesis Data Streams - Guide du développeur.
- Envisagez d'augmenter la Kinesis Data Streams API ou d'augmenter la taille du tampon pour le récepteur si les appels d'API sont limités. Pour de plus amples informations, veuillez consulterLimites Kinesis Data Streamsdans leKinesis Data Streams - Guide du développeur.
- Si les données sont diffusées trop rapidement, envisagez de demander une augmentation de limite de débit pour le flux de diffusion Kinesis Data Firehose. Ou, si les appels d'API sont limités, demandez une augmentation de la limite d'appels d'API (consultez Limites d'Amazon Kinesis Data Firehose) ou augmentez la taille du tampon pour le récepteur.
- Après avoir augmenté le nombre de partitions d'un Kinesis Data Streams ou la limite de débit d'un flux de diffusion de Kinesis Data Firehose, modifiez l'Agent Kinesis pour

Windows`appsettings.json`Pour augmenter les enregistrements par seconde ou les octets par seconde pour le récepteur. Sinon, Kinesis Agent pour Windows ne peut pas tirer profit de l'augmentation des limites.

S'applique à

Cette information s'applique à Kinesis Agent pour Windows version 1.0.0.0.0.0.115 et aux versions ultérieures.

Manque d'espace sur le disque

Symptoms

Kinesis Agent pour Windows s'exécute sur une machine dont l'espace disque est très faible sur un ou plusieurs lecteurs de disque.

Causes

Plusieurs causes possibles peuvent entraîner ce problème :

- Le fichier de configuration de journalisation Kinesis Agent pour Windows n'est pas correct.
- La file d'attente persistante Kinesis Agent pour Windows n'est pas configurée correctement.
- Une autre application ou un autre service utilise de l'espace disque.

Resolutions

Pour résoudre les problèmes d'espace disque, procédez comme suit :

- Si l'espace disque est faible sur le disque qui contient les fichiers journaux de l'Agent Kinesis pour Windows, examinez le répertoire du fichier journal (généralement`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`). Assurez-vous de la conservation d'un nombre raisonnable de fichiers journaux et de la taille raisonnable de ces fichiers journaux. Vous pouvez contrôler l'emplacement, la conservation et la description des journaux Kinesis Agent pour Windows en modifiant l'`%PROGRAMFILES%\Amazon\AWSKinesisTap\Nlog.xml`Fichier de configuration.
- Lorsque la fonctionnalité de mise en file d'attente du récepteur est activée, examinez les déclarations du récepteur qui utilisent cette fonctionnalité. Assurez-vous que la paire clé-valeur

`QueuePath` fait référence à un disque disposant de suffisamment d'espace disque pour contenir le nombre maximal de lots spécifié à l'aide de la paire clé-valeur `QueueMaxBatches`. Si ce n'est pas possible, réduisez la valeur de la paire clé-valeur `QueueMaxBatches` afin que les données s'intègrent facilement dans l'espace disque restant pour le disque spécifié.

- Utilisez l'explorateur de fichiers Windows pour rechercher les fichiers qui utilisent l'espace disque, puis transférez ou supprimez les fichiers en trop. Modifiez la configuration de l'application ou du service utilisant de grandes quantités d'espace disque.

S'applique à

Cette information s'applique à Kinesis Agent pour Windows version 1.0.0.0.0.0.115 et aux versions ultérieures.

Outils de dépannage

En plus de vérifier le fichier de configuration, vous pouvez utiliser l'`ktdiag.exe`, qui fournit plusieurs autres fonctionnalités pour diagnostiquer et résoudre les problèmes lors de la configuration et de l'utilisation de Kinesis Agent pour Windows. L'outil `ktdiag.exe` se trouve dans le répertoire `%PROGRAMFILES%\Amazon\AWSKinesisTap`.

- Si vous pensez que les fichiers journaux reposant sur un modèle de fichier spécifique sont écrits dans un répertoire, mais qu'ils ne sont pas traités par l'Agent Kinesis pour Windows, utilisez l'outil`/w`pour vérifier que ces modifications sont détectées. Supposons par exemple que vous vous attendez à ce que les fichiers journaux reposant sur le modèle de nom de fichier `*.log` soient écrits dans le répertoire `c:\foo`. Vous pouvez utiliser le commutateur `/w` lorsque vous exécutez l'outil `ktdiag.exe`, en spécifiant le répertoire et le modèle de fichier :

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

  Si les fichiers journaux sont écrits, vous pouvez voir un résultat similaire à ce qui suit :

  ```
  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
```

Si ce n'est pas le cas, il s'agit d'un problème au niveau de l'application ou du service lors de l'écriture des journaux, ou d'un problème de configuration de la sécurité plutôt que d'un problème avec l'Agent Kinesis pour Windows. Si vous obtenez ce résultat, mais que Kinesis Agent pour Windows ne semble toujours pas traiter les journaux, consultezAucune donnée n'est diffusée à partir d'ordinateurs de bureaux ou de serveurs vers les services AWS attendus.

- Parfois, les journaux ne sont écrits qu'occasionnellement, mais il serait utile de vérifier que l'Agent Kinesis pour Windows fonctionne correctement. Utilisez le commutateur `/log4net` pour simuler une application qui écrit des journaux à l'aide de la bibliothèque `Log4net`, par exemple :

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

Cette opération écrit un fichier journal de style `Log4net` dans le fichier journal `c:\foo\log2.log` et continue d'ajouter de nouvelles entrées de journal jusqu'à ce qu'une touche soit enfoncée. Vous pouvez configurer plusieurs options à l'aide de commutateurs supplémentaires qui peuvent être spécifiés après le nom de fichier :

Verrouillage : `-lm`, `-li` ou `-le`

Vous pouvez spécifier l'un des commutateurs de verrouillage suivants qui contrôlent la façon dont le fichier journal est verrouillé :

`-lm`

Un verrouillage minimum est utilisé sur le fichier journal, ce qui permet un accès maximal au fichier journal.

`-li`

Seuls les threads d'un même processus peuvent accéder au journal en même temps.

`-le`

Un seul thread à la fois peut accéder au journal. Il s'agit de l'option par défaut.

`-tn:`*`millisecondes`*

Spécifie le nombre de *`millisecondes`* entre chaque écriture des entrées de journal. La valeur par défaut est de 1 000 millisecondes (1 seconde).

`-sm:`*`octets`*

Spécifie le nombre d'*`octets`* pour chaque entrée de journal. La valeur par défaut est 1 000 octets.

`-bk:`*`number`*

Spécifie le *`nombre`* d'entrées de journal à écrire à la fois. La valeur par défaut est 1.

- Parfois, il est utile de simuler une application qui écrit dans le journal des événements Windows. Utilisez le commutateur `/e` pour écrire des entrées de journal dans un journal des événements Windows, par exemple :

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

Cette opération écrit des données de journal dans le journal d'événements de l'application Windows jusqu'à ce qu'une touche soit enfoncée. Vous pouvez spécifier les options supplémentaires suivantes après le nom du journal :

`-tn:`*`millisecondes`*

Spécifie le nombre de *`millisecondes`* entre chaque écriture des entrées de journal. La valeur par défaut est de 1 000 millisecondes (1 seconde).

`-sm:`*`octets`*

Spécifie le nombre d'*`octets`* pour chaque entrée de journal. La valeur par défaut est 1 000 octets.

`-bk:`*`number`*

Spécifie le *`nombre`* d'entrées de journal à écrire à la fois. La valeur par défaut est 1.

Création de Kinesis Agent pour les plugins Windows

Dans la plupart des cas, la création d'un plugin Amazon Kinesis Agent pour Microsoft Windows n'est pas nécessaire. Kinesis Agent pour Windows est hautement configurable et contient des sources et des récepteurs puissants, comme`DirectorySource`and`KinesisStream`, qui sont suffisantes pour la plupart des scénarios. Pour plus d'informations sur les sources et récepteurs existants, consultez Configuration d'Amazon Kinesis Agent pour Microsoft Windows.

Dans les situations inhabituelles, il peut être nécessaire d'étendre Kinesis Agent pour Windows à l'aide d'un plug-in personnalisé. C'est le cas dans les exemples de situations suivants :

- Package d'une déclaration `DirectorySource` complexe à l'aide des outils d'analyse `Regex` ou `Delimited` afin qu'elle soit facile à appliquer dans de nombreux types de fichiers de configuration.
- Création d'une source innovante qui n'est pas basée sur un fichier ou qui dépasse les capacités d'analyse fournies par les outils d'analyse existants.
- Création d'un récepteur pour un service AWS qui n'est actuellement pas pris en charge.

Rubriques

Mise en route avec Kinesis Agent pour les plugins Windows

Les plug-ins personnalisés n'ont rien de spécial. Toutes les sources et tous les récepteurs existants utilisent les mêmes mécanismes que ceux utilisés par les plug-ins personnalisés pour se charger lors du démarrage de Kinesis Agent pour Windows. Ils instancient les plug-ins appropriés après avoir lu le`appsettings.json`Fichier de configuration.

Au démarrage de Kinesis Agent pour Windows, la séquence suivante se produit :

1. Kinesis Agent pour Windows analyse les assembly dans le répertoire d'installation (`%PROGRAMFILES%\Amazon\AWSKinesisTap`) pour les classes qui implémentent la méthode`IFactory<T>`définie dans l'interface`Amazon.KinesisTap.Core`Assembly Cette

interface est définie dans`Amazon.KinesisTap.Core\Infrastructure\IFactory.cs`dans le code source de Kinesis Agent pour Windows.

2. Kinesis Agent for Windows charge les assembly contenant ces classes et appelle la méthode`RegisterFactory`sur ces classes.
3. Kinesis Agent pour Windows charge le`appsettings.json`Fichier de configuration. Pour chaque source et chaque récepteur dans le fichier de configuration, les paires clé-valeur `SourceType` et `SinkType` sont examinées. Si des fabriques enregistrées portent le même nom que les valeurs des paires clé-valeur `SourceType` et `SinkType`, la méthode `CreateInstance` est appelée sur ces fabriques. La configuration et d'autres informations sont transmises à la méthode `CreateInstance` en tant qu'objet `IPluginContext`. La méthode `CreateInstance` est chargée de configurer et d'initialiser le plug-in.

Pour qu'un plug-in fonctionne correctement, une classe de fabrique enregistrée doit créer le plug-in, puis la classe de plug-in elle-même doit être définie.

Le code source de Kinesis Agent pour Windows est situé à l'adressehttps://github.com/awslabs/kinesis-agent-windows.

Implémentation de Kinesis Agent pour les usines de plug-in Windows

Suivez les étapes suivantes pour implémenter une fabrique de plug-ins Kinesis Agent pour Windows.

Pour créer une usine de plug-in Kinesis Agent pour Windows

1. Créez un projet de bibliothèque C # ciblant .NET Framework 4.6.
2. Ajoutez une référence à l'assembly `Amazon.KinesisTap.Core`. Cet assembly est situé dans le`%PROGRAMFILES%\Amazon\AWSKinesisTap`après l'installation de Kinesis Agent pour Windows.
3. Utilisez `NuGet` pour installer le package `Microsoft.Extensions.Configuration.Abstractions`.
4. Utilisez `NuGet` pour installer le package `System.Reactive`.
5. Utilisez `NuGet` pour installer le package `Microsoft.Extensions.Logging`.

6. Créez une classe Factory qui implémente `IFactory<IEventSource>` pour les sources ou `IFactory<IEventSink>` pour les récepteurs. Ajoutez les méthodes `RegisterFactory` et `CreateInstance`.

   Par exemple, le code suivant crée une fabrique de plug-ins Kinesis Agent pour Windows qui crée une source générant des données aléatoires :

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

L'instruction `switch` est utilisée dans la méthode `CreateInstance` au cas où vous souhaiteriez améliorer la fabrique de façon à créer différents types d'instances.

Pour créer une fabrique de récepteurs qui crée un récepteur qui ne fait rien, utilisez une classe similaire à la classe suivante :

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

Implémentation de Kinesis Agent pour les sources de plug-in Windows

Suivez les étapes suivantes pour implémenter une source de plug-ins Kinesis Agent pour Windows.

Pour créer une source de plug-in Kinesis Agent pour Windows

1. Ajoutez une classe qui implémente l'interface `IEventSource<out T>` au projet créé précédemment pour la source.

   Par exemple, utilisez le code suivant pour définir une source qui génère des données aléatoires :

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

Dans cet exemple, la classe `RandomSource` hérite de la classe `EventSource<T>` étant donné qu'elle fournit la propriété `Id`. Bien que cet exemple ne prenne pas en charge l'attribution de signets, cette classe de base est également utile pour implémenter cette fonctionnalité. Les enveloppes permettent de stocker des métadonnées et d'envelopper les données arbitraires pour les diffuser auprès des récepteurs. La classe `RandomData` est définie à l'étape suivante et représente le type d'objet de sortie de cette source.

2. Ajoutez une classe au projet défini précédemment, qui contient les données qui sont diffusées à partir de la source.

   Par exemple, un conteneur de données aléatoires peut être défini comme suit :

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

3. Compilez le projet précédemment défini.
4. Copiez l'assembly dans le répertoire d'installation de Kinesis Agent for Windows.
5. Créez ou mettez à jour un`appsettings.json`qui utilise la nouvelle source et placez-le dans le répertoire d'installation de Kinesis Agent pour Windows.
6. Arrêtez et démarrez Kinesis Agent pour Windows.
7. Vérifiez le fichier journal Kinesis Agent pour Windows actuel (généralement situé dans le répertoire`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`) afin de vous assurer qu'il n'y a aucun problème avec le plug-in source personnalisé.
8. Vérifiez que les données sont reçues par le service AWS souhaité.

Pour obtenir un exemple d'extension de la`DirectorySource`pour implémenter l'analyse d'un format de journal particulier, consultez`Amazon.KinesisTap.Uls\UlsSourceFactory.cs`and`Amazon.KinesisTap.Uls\UlsLogParser.cs`dans le code source de Kinesis Agent pour Windows.

Pour obtenir un exemple de création d'une source qui fournit une fonctionnalité d'attribution de signets, consultez`Amazon.KinesisTap.Windows\WindowsSourceFactory.cs`and`Amazon.KinesisTap.Windows\EventLogSource.cs`dans le code source de Kinesis Agent pour Windows.

Implémentation de Kinesis Agent pour les puits de plug-in Windows

Suivez les étapes suivantes pour implémenter un récepteur de plug-ins Kinesis Agent pour Windows.

Pour créer un collecteur de plug-in Kinesis Agent pour Windows

1. Ajoutez une classe à l'objet défini précédemment qui implémente l'interface `IEventSink`.

   Par exemple, le code suivant implémente un récepteur qui ne fait que consigner l'arrivée d'enregistrements, qui sont ensuite supprimés.

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

   Dans cet exemple, la classe de récepteur `NullSink` hérite de la classe `EventSink`, car elle permet de transformer des enregistrements en différents formats de sérialisation tels que JSON et XML.

2. Compilez le projet précédemment défini.
3. Copiez l'assembly dans le répertoire d'installation de Kinesis Agent for Windows.

4. Créez ou mettez à jour un`appsettings.json`qui utilise le nouveau récepteur et placez-le dans le répertoire d'installation de Kinesis Agent pour Windows. Par exemple, pour utiliser les plug-ins personnalisés `RandomSource` et `NullSink`, vous pouvez utiliser le fichier de configuration `appsettings.json` suivant :

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

   Cette configuration crée une source qui envoie une instance de `RandomData` avec la valeur `RandomValue` définie sur un nombre aléatoire compris entre 0 et 50 toutes les 10 secondes. Cela crée un récepteur qui transforme les instances `RandomData` entrantes au format JSON, enregistre ce JSON, puis supprime les instances. Assurez-vous d'inclure les deux exemples de fabriques, la classe de source `RandomSource` et la classe de récepteur `NullSink` du projet défini précédemment pour utiliser l'exemple de fichier de configuration.

5. Arrêtez et démarrez Kinesis Agent pour Windows.

6. Vérifiez le fichier journal Kinesis Agent pour Windows actuel (généralement situé dans le répertoire`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`) afin de vous assurer qu'il n'y a aucun problème avec le plug-in de récepteur personnalisé.
7. Vérifiez que les données sont reçues par le service AWS souhaité. Étant donné que l'exemple de classe `NullSink` ne diffuse vers aucun service AWS, vous pouvez vérifier le bon fonctionnement du récepteur en recherchant les messages de journaux indiquant que les enregistrements ont été reçus.

   Par exemple, vous pouvez accéder à un fichier journal semblable à :

```
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}.
```

Si vous créez un récepteur qui accède aux services AWS, il existe des classes de base qui peuvent vous être utiles. Pour un évier qui utilise le`AWSBufferedEventSink`classe de base, voir`Amazon.KinesisTap.AWS\CloudWatchLogsSink.cs`dans le code source de Kinesis Agent pour Windows.

Historique de document pour Amazon Kinesis Agent for Microsoft Windows User Guide

Version d'API : 2018-10-15

Le tableau suivant décrit les modifications apportées à laGuide de l'utilisateur Amazon Kinesis Agent pour Microsoft Windows(le présent document).

update-history-change	update-history-description	update-history-date
Mise à jour de la documentation	Ajout d'instructions pour l'installation de MSI. Mise à jour de la configuration de DirectorySource et ajout de WindowsEntentLogPollingSource. Pour la configuration du collecteur, ajout de la configuration de synchronisation du système de fichiers local ; Profilerefreshingawsredentialprovider ; informations sur les décorations de texte, la résolution de variables dans les attributs de collecteur, la configuration des points de terminaison régionaux STS pour les puits, la configuration des points de terminaison VPC et la configuration de serveurs proxy alternatifs. Pour les tuyaux, ajout d'attributs de configuration.	23 février 2021
Mise à jour de la documentation	Mise à jour de pour indiquer que les spécifications de	7 novembre 2018

	l'emplacement d'Amazon S3 sont sensibles à la casse.	
Publication initiale, version 1.0.0.115	Première version du Guide de l'utilisateur de Kinesis Agent pour Windows.	5 novembre 2018

Glossaire AWS

Pour connaître la terminologie AWS la plus récente, veuillez consulter le Glossaire AWS dans les Références générales AWS.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.