aws

使用者指南

Microsoft Windows 的 Amazon Kinesis Kinesis 代理程式

Microsoft Windows 的 Amazon Kinesis Kinesis 代理程式: 使用者指南

Table of Contents

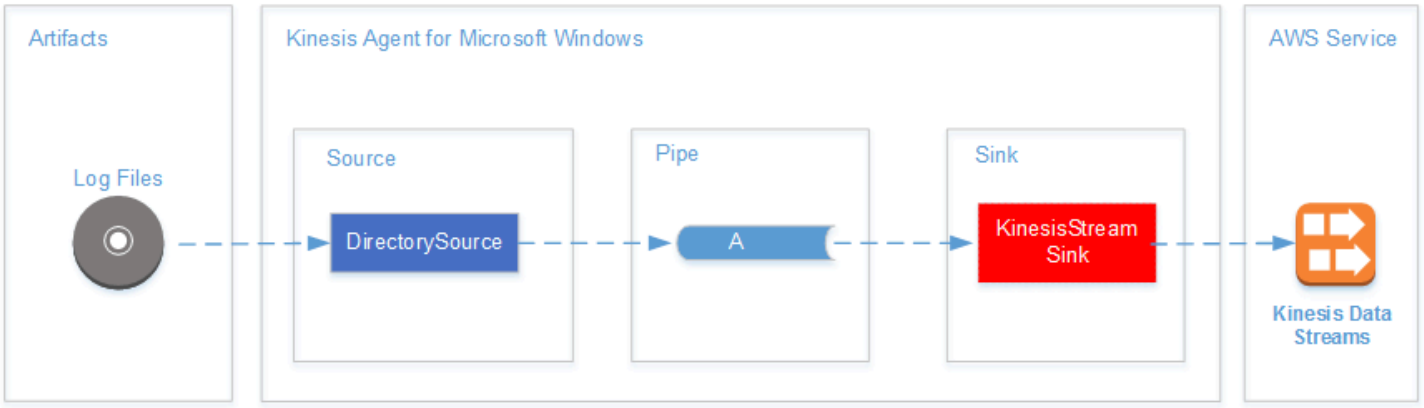
什麼是 Amazon Kinesis Agent，適用於微軟視窗？

Amazon Kinesis Agent (適用於 Windows 的 Kinesis Agent) 是可設定且可延伸的代理程式。它會在 Windows 桌面平台電腦和伺服器的叢集上執行 (現場部署或 AWS 雲端內)。適用於 Windows 的 Kinesis Agent 可以有效率且可靠地收集、剖析、轉換及將日誌、事件和指標串流至各種 AWS 服務，包含Kinesis Data Streams、Kinesis Data Firehose、Amazon CloudWatch，以及CloudWatch Logs。

從這些服務，您接著便可以使用各種其他 AWS 服務存放、分析及視覺化資料，包含以下項目：

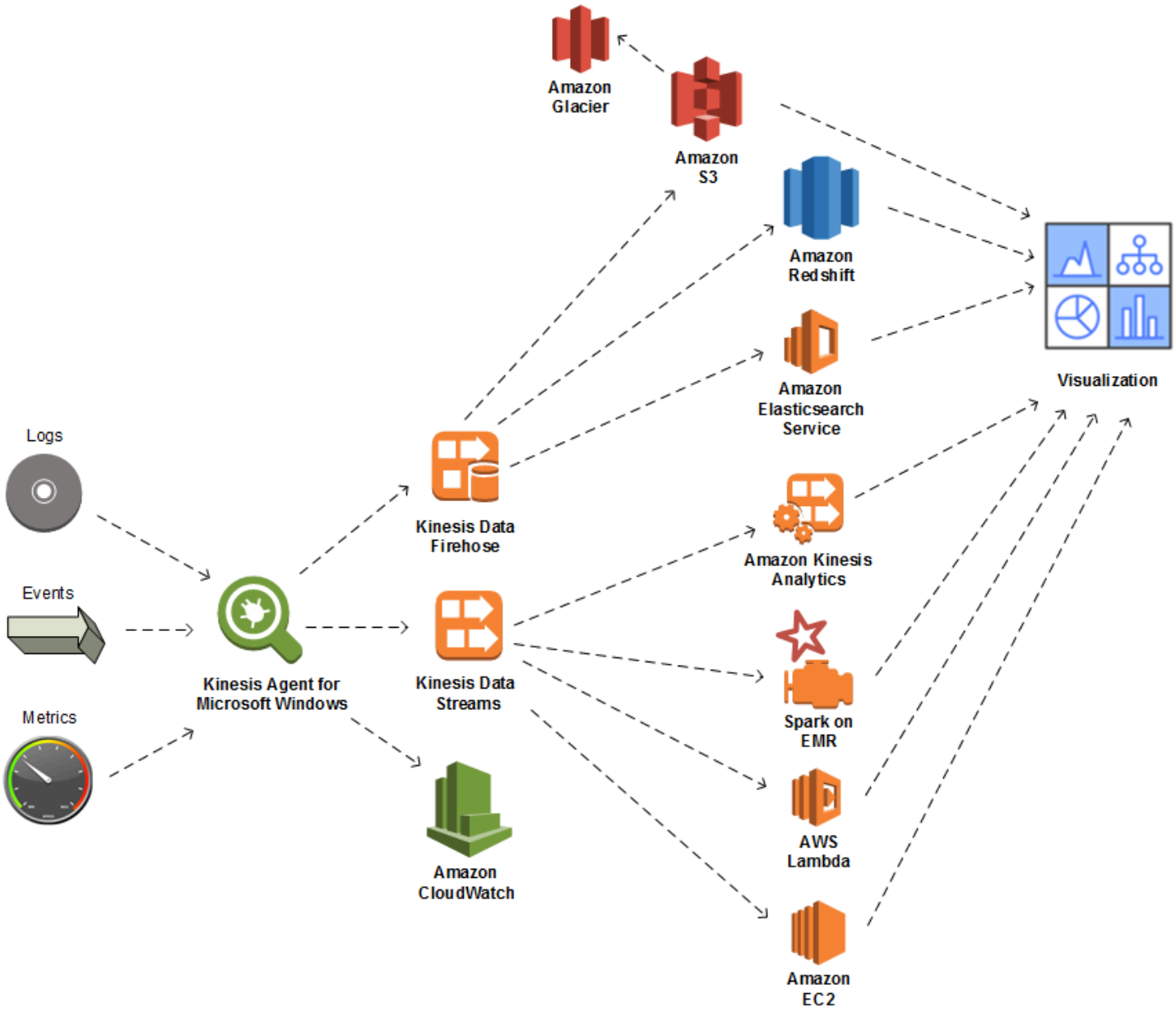
- Amazon Simple Storage Service (Amazon S3)
- Amazon Redshift
- Amazon Elasticsearch Service (Amazon ES)
- Kinesis Data Analytics
- Amazon QuickSight
- Amazon Athena
- Kibana

下圖說明簡易的 Windows 專用 Kinesis Agent 組態，可將日誌檔案串流至 Kinesis Data Streams。



如需來源、管道和接收的詳細資訊，請參閱微軟視窗概念的 Amazon Kinesis Kinesis 代理程式。

下圖說明您可以使用串流處理框架建置自訂、即時資料管道的一些方式。這些架構包括 Kinesis Data Analytics、Amazon EMR 上的 Apache 星火，以及 AWS Lambda。



主題

關於 AWS

Amazon Web Services (AWS) 集合了各項數位基礎設施服務，讓您在開發應用程式時加以利用。服務包含了運算、儲存、資料庫、分析及應用程式同步 (簡訊和佇列)。AWS 採按用量付費的服務模式。您只需為您 (或您應用程式) 使用的服務付費。此外，為了讓服務在原型設計和實驗上更親近易用，AWS 也提供了免費用量方案。在此方案中，特定用量層級以下的服務皆為免費。如需 AWS 成本和免費方案，請參閱資源中心入門。若要建立 AWS 帳戶，請開啟 AWS 首頁並註冊。

使用適用於 Windows 的 Kinesis Agent 可進行什麼作業？

Windows 專用 Kinesis Agent 提供下列功能：

收集日誌、事件及指標資料

適用於 Windows 的 Kinesis Agent 會收集、剖析、轉換及從伺服器叢集和桌面平台，將日誌、事件和指標串流至一或多個 AWS 服務。服務接收到的承載可能會和原始來源的格式有所不同。例如，日誌可能會以特定的文字格式 (例如 syslog 格式) 存放在伺服器上。適用於 Windows 的 Kinesis Agent 可收集和剖析該文字，並選擇性地將它轉換成如 JSON 等格式，再串流至 AWS。這會輔助某些使用 JSON 的 AWS 服務，使其能更輕易地進行處理。串流至 Kinesis Data Streams 可由 Kinesis Data Analytics 持續處理，以產生額外的指標和彙整指標，提供給即時儀表板使用。您可以使用各種 AWS 服務 (例如 Amazon S3) 存放資料，取決於資料在資料管道下游使用的方式。

與 AWS 服務整合

您可以設定 Windows 專用 Kinesis Agent，將日誌檔案、事件和指標傳送至數個不同的 AWS 服務：

- Kinesis Data Firehose— 輕鬆地將串流的資料存放在 Amazon S3、Amazon Redshift、Amazon ES 或Splunk以進一步分析。
- Kinesis Data Streams— 使用裝載於 Kinesis Data Analytics 中的自訂應用程式或Amazon EMR。或者使用運行在Amazon EC2執行個體，或在中執行的自訂無伺服器函數AWS Lambda。

- CloudWatch— 以圖形檢視串流的度量，您可以將其合併到儀表板中。然後設定違反預先設定閾值時，由指標值觸發的 CloudWatch 警示。
- CloudWatch Logs— 儲存串流的日誌和事件，並在 AWS 管理主控台中檢視和搜尋這些日誌和事件，或在資料管道中進一步處理這些日誌和事件。

快速安裝及設定

您只需要幾個步驟，便可安裝及設定 Windows 專用 Kinesis Agent。如需更多詳細資訊，請參閱 安裝適用於視窗的 Kinesis 代理程式 及 設定適用於微軟視窗的 Amazon Kinesis。簡易的宣告式組態檔案會指定以下項目：

- 來源以及要收集的日誌、事件及指標格式。
- 要套用至所收集資料的轉換。其他資料可以包含和現有的資料轉換和篩選。
- 串流最終資料的目標，以及串流承載的緩衝、分片及格式。

Windows 專用 Kinesis Agent 隨附內建的剖析器，可用於常見 Microsoft 企業服務所產生的日誌檔案，例如：

- Microsoft Exchange
- SharePoint
- Active Directory 網域控制站
- DHCP 伺服器

無須全程管理

Windows 專用 Kinesis Agent 會自動因應各種狀況，而不會遺失任何資料。這包含日誌輪換、重新開機後進行復原，以及暫時性的網路或服務中斷。您可以設定 Windows 適用的 Kinesis Agent，自動更新至新的版本。在這些狀況中皆不需要任何操作員的介入。

使用開放架構延伸

若宣告式的功能和內建外掛程式不足以用於監控伺服器或桌面平台系統，您可以透過建立外掛程式，延伸適用於 Windows 的 Kinesis Agent。新的外掛程式可啟用日誌、事件及指標的新來源及目標。適用於 Windows 的 Kinesis Agent 的來源碼可在https://github.com/awslabs/kinesis-agent-windows。

Benefits

Windows 版 Kinesis Agent 會為資料管道執行日誌、事件和指標的初始資料收集、轉換及串流。建置這些資料管道有許多好處：

分析與視覺化

Windows 專用 Kinesis Agent 與 Kinesis Data Firehose 的整合及其轉換功能，讓與數種不同分析和視覺化服務的整合過程更為容易。

- Amazon QuickSight— 可從許多不同來源擷取的雲端式 BI 服務。適用於視窗的 Kinesis 代理程式可以轉換資料，並透過 Kinesis Data Firehose 串流到 Amazon S3 和 Amazon 紅移。此程序可使用 Amazon QuickSight 視覺化，從資料探索更深入的詳情。
- Athena— 啟用以 SQL 為基礎的資料查詢的互動式查詢服務。適用於視窗的 Kinesis 代理程式可以透過 Kinesis 資料防火軟管將資料轉換並串流至 Amazon S3。然後，Athena 可以以互動方式對該資料執行 SQL 查詢，以快速檢查和分析記錄和事件。
- Kibana— 開放原始碼資料視覺化工具。適用於視窗的 Kinesis 代理程式可以透過 Kinesis 資料防火軟管將資料轉換並串流至 Amazon ES。您接著便可以使用 Kibana 來探索該資料。建立並開啟不同的視覺化，包含直方圖、線條圖、圓形圖、熱圖及地理空間圖形。

Security

包含 Windows 版 Kinesis Agent 的日誌和事件資料分析管道可偵測及對組織中的安全違規作出提醒，協助您封鎖或攔截攻擊。

應用程式效能

Windows 專用 Kinesis Agent 可收集應用程式或服務效能的日誌、事件和指標資料。完整資料管道接著便可分析此資料。此分析可透過偵測及報告在其他情況下可能不明顯的缺失，協助您改善應用程式和服務的效能及可靠性。例如，您可以偵測服務 API 呼叫執行時間的大幅度變更。當和部署相互關聯時，此功能可協助您尋找和解決您所擁有服務新的效能問題。

服務操作

資料管道可分析所收集的資料，預測潛在的操作問題，並提供如何避免服務中斷的詳情。例如，您可以分析日誌、事件及指標來判斷目前和預測的容量用量，並在服務使用者受到影響前使額外的容量上線。若發生服務中斷，您可以分析資料來判斷中斷期間對客戶造成的影響。

Auditing

資料管道可處理 Windows 適用 Kinesis Agent 收集和轉換的日誌、事件和指標。您接著便可以使用各種 AWS 服務稽核這項處理後的資料。例如，Kinesis Data Firehose 可從 Windows 專用 Kinesis Agent 接收資料串流，將資料存放在 Amazon S3 中。您接著便可以使用 Athena 執行互動式的 SQL 查詢，稽核這項資料。

Archiving

通常最重要的操作資料便是最近收集的資料。但是，分析在數年間針對應用程式和服務收集的資料也很有用 (例如用於長範圍規劃)。保留大量資料可能會很昂貴。適用於 Windows 的 Kinesis 代理程式可以透過 Kinesis 資料防火軟管收集、轉換和儲存 Amazon S3 中的資料。因此，Amazon S3 Glacier可用於減少存檔較舊資料的成本。

Alerting

適用於視窗串流的測量指標至 CloudWatch 控。您接著便可以建立 CloudWatch 警示，透過Amazon Simple Notification Service (Amazon SNS)當測量結果一致違反特定臨界值時。這可讓工程師更進一步地感知其應用程式和服務的操作問題。

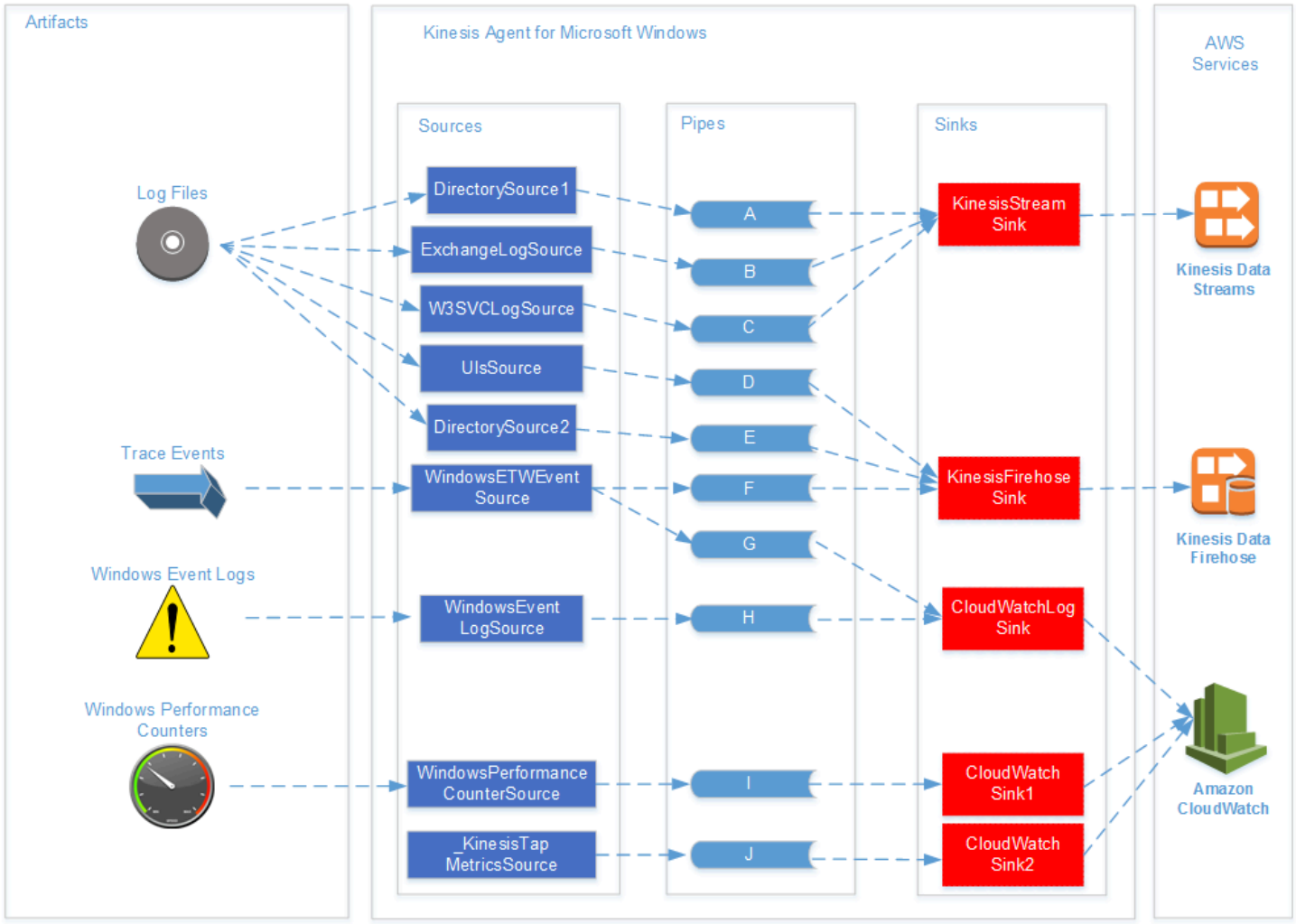
開始使用適用於視窗的 Kinesis 代理程式

若要進一步了解 Windows 適用的 Kinesis Agent，建議您從下列章節開始著手：

- 微軟視窗概念的 Amazon Kinesis Kinesis 代理程式
- Amazon Kinesis 代理程式入門

微軟視窗概念的 Amazon Kinesis Kinesis 代理程式

了解 Amazon Kinesis 代理程式 (Windows 專用 Kinesis 代理程式) 的關鍵概念，可讓您在收集和串流桌面平台和伺服器叢集上資料至資料管道剩餘部分進行處理的過程更為容易。



此資料管道圖說明了下列元件和程序：

伺服器和桌面平台擁有像是日誌檔案、事件及指標等，由一或多個「Windows」的 Kinesis Agent」收集的成品sources。例如，資料可選擇性地從一般檔案的文字格式轉換成物件。

資料 (以物件型式或文字型式) 接著便可以流向一或多個「Windows」的 Kinesis 代理程式」管道。管道會將一個來源連線到一個「Kinesis 代理程式」sink。管道可選擇性地篩選不必要的資料。

接收可選擇地將剖析的資料轉換成物件、JSON 或 XML。接收會將資料傳送到特定 AWS 服務，例如 Kinesis Data Streams、Kinesis Data Firehose 或 Amazon CloudWatch。

使用多個管道，單一來源可將相同資料傳送到多個接收 (例如，請參閱圖中的管道 F 及 G)。使用多個管道，不同來源可將資料串流至單一接收 (例如，請參閱圖中的管道 A、B 及 C)。您可以使用多個管道，將來自多個接收的資料串流至多個來源。來源、接收和管道都擁有類型，並且您可以擁有超過一個相同類型的來源、接收或管道。

如需宣告來源、接收及管道的組態檔案範例，請使用 適用於視窗設定範例的 Kinesis 代理程式。

主題

資料管道

AData Pipeline用於收集、處理、視覺化，以及可能產生應用程式和服務的警示。適用於 Windows 的 Kinesis Agent 會在一開始就適用於資料管線 — 記錄、事件和度量是從桌上型電腦或伺服器叢集收集。Windows 版 Kinesis 代理程式會將收集的資料串流至各種 AWS 服務，構成資料管道剩餘部分。資料管道具有一個目的，例如即時視覺化特定服務的運作狀況，協助工程師更有效地操作該服務。服務運作狀況資料管道可執行以下任何一項作業：

- 在問題影響到服務客戶的體驗前，提醒工程師這些問題。
- 透過顯示資源用量趨勢，協助工程師有效管理服務成本。這些趨勢可讓他們適當地調整資源層級，甚至是實作自動調整規模案例。
- 提供服務客戶所報告問題根本原因的詳情。這可加速解決這些問題，減少支援成本。

如需使用 Windows 專用 Kinesis 代理程式建構資料管道的逐步範例，請參閱教學課程：使用適用於視窗的 Kinesis 代理程式將 JSON 日誌檔串流到 Amazon S3。

Sources

適用於視窗 Kinesis 代理程式source會收集日誌、事件或指標。來源會根據來源的類型，從該資料的特定生產者收集特定資料類型。例如，`DirectorySource` 類型會從檔案系統中的特定目錄收集日誌檔案。若資料尚未結構化 (如某些類型的日誌檔案)，來源在將文字表示剖析至某種結構化型式的過程中會很有用。每個來源對應於一個特定的來源宣告適用於視窗的啟 Kinesis 代理程式`appsettings.json`組態檔案。來源宣告可提供設定來源的基本詳細資訊，根據特定資料收集需求

量身訂做來源。可設定的詳細資訊類型會因來源類型而有所不同。例如，`DirectorySource` 來源類型需要指定日誌檔案所在的目錄。

如需來源類型和來源宣告的詳細資訊，請參閱來源宣告。

Sinks

適用於視窗 Kinesis 代理程式sink會接受 Kinesis 代理程式收集的資料，並將該資料串流至數種可能 AWS 服務中的其中一種，構成資料管道的剩餘部分。每個水槽對應於一個特定的接收宣告適用於視窗的啟 Kinesis 代理程式`appsettings.json`組態檔案。接收宣告可提供設定接收的基本詳細資訊，根據特定資料串流需求量身訂做接收。可設定的詳細資訊類型會因接收類型而有所不同。例如，某些接收類型允許接收宣告為提供給他們的資料指定特定的序列化 `Format`。在接收宣告中指定此選項時，所收集資料的序列化會在將資料串流至與接收相關聯的 AWS 服務前發生。

如需接收類型和接收宣告的詳細資訊，請參閱目的地宣告。

Pipes

適用於視窗 Kinesis 代理程式管道會將適用於 Windows 來源的 Kinesis 代理程式輸出連線到適用於 Windows 接收的 Kinesis 代理程式輸入。它會選擇性地在資料於管道中流動時轉換資料。每個管道都對應到特定 Kinesis「管道宣告」`appsettings.json`組態檔案。管道宣告會提供設定接收的基本詳細資訊，例如管道的來源和接收。

如需管道類型和管道宣告的詳細資訊，請參閱管道宣告。

Amazon Kinesis 代理程式入門

您可以使用適用於微軟視窗的 Amazon Kinesis 代理程式 (適用於視窗) 收集、剖析、轉換並將日誌、事件和指標從您的 Windows 叢集串流至各種 AWS 服務。以下資訊包含安裝及設定適用於 Windows 版 Kinesis 代理程式的事前準備和逐步說明。

主題

Prerequisites

在安裝 Windows 專用 Kinesis 代理程式之前，請務必備妥下列先決條件：

- 熟悉視窗概念的 Kinesis 代理程式。如需詳細資訊，請參閱 微軟視窗概念的 Amazon Kinesis Kinesis 代理程式。
- AWS 帳戶，可使用各種與您資料管道相關的 AWS 服務。如需建立和設定 AWS 帳戶的資訊，請參閱設定 AWS 帳戶。
- 在每個將執行 Kinesis 代理程式的桌面平台或伺服器上安裝 Microsoft .NET 架構 4.6 或更新版本。如需詳細資訊，請參閱 Microsoft .NET 文件中的 Install the .NET Framework for developers。

  若要判斷在桌面平台或伺服器上安裝的 .NET Framework 最新版本，請使用以下 PowerShell 指令碼：

  ```
  [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
  ```

- 您希望從 Windows 專用 Kinesis 代理程式傳送 Amazon Kinesis Data Streams (若使用)。建立串流使用Kinesis Data Streams 主控台，AWS CLI, 或適用於 Windows PowerShell 的 AWS 工具。如需詳細資訊，請參閱「」建立及更新資料串流中的Amazon Kinesis Data Streams 開發人員指南。

- 您希望從 Windows 專用 Kinesis 代理程式傳送資料的 Amazon Kinesis Data Firehose 交串流 (若使用)。建立交付串流使用Kinesis Data Firehose 主控台，AWS CLI, 或適用於 Windows PowerShell 的 AWS 工具。如需詳細資訊，請參閱「」建立 Amazon Kinesis Data Firehose 交付串流中的Amazon Kinesis Data Firehose 開發人員指南。

設定 AWS 帳戶

如果您還沒有 AWS 帳戶，請完成下列步驟建立新帳戶。

註冊 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電，並在電話鍵盤輸入驗證碼。

為您自己建立一個管理員使用者，並將使用者新增至管理員群組 (主控台)

1. 登入。IAM 主控台作為帳戶擁有者，請選擇根使用者，然後輸入您的 AWS 帳戶電子郵件地址。在下一頁中，輸入您的密碼。

   > **Note**
   >
   > 強烈建議您遵循使用**Administrator**遵循並妥善鎖藏根使用者登入資料的 IAM 使用者。只在需要執行少數帳戶和服務管理任務時，才以根使用者身分登入。

2. 在導覽窗格中，選擇 Users (使用者)，然後選擇 Add user (新增使用者)。
3. 在 User name (使用者名稱) 中輸入 **Administrator**。
4. 選取 AWS Management Console access (AWS 管理主控台存取) 旁的核取方塊。然後選取 Custom password (自訂密碼)，接著在文字方塊中輸入您的新密碼。
5. (選用) 根據預設，AWS 會要求新使用者在第一次登入時建立新密碼。您可以清除 User must create a new password at next sign-in (使用者下次登入必須建立新的密碼) 旁的核取方塊，讓新使用者登入時可以重設密碼。
6. 選擇下一頁: 許可。
7. 在 Set permissions (設定許可) 下，選擇 Add user to group (將使用者新增至群組)。
8. 選擇 Create group (建立群組)。

9. 在 Create group (建立群組) 對話方塊中，請於 Group name (群組名稱) 輸入 **`Administrators`**。
10. 選擇篩選政策，然後選取AWS 託管-工作功能來篩選表格內容。
11. 在政策清單中，選取 AdministratorAccess 的核取方塊。接著選擇 Create group (建立群組)。

> **Note**
>
> 您必須先啟用 IAM 使用者和角色對帳單的存取權，才能使用 `AdministratorAccess` 許可存取 AWS 帳單和成本管理主控台。若要這樣做，請遵循委派對帳單主控台的存取權相關教學課程的步驟 1 中的指示。

12. 回到群組清單，選取新群組的核取方塊。必要時，選擇 Refresh (重新整理) 以顯示清單中的群組。
13. 選擇下一頁: Tags (標籤)。
14. (選用) 藉由附加標籤做為索引鍵/值組，將中繼資料新增至使用者。如需在 IAM 中使用標籤的詳細資訊，請參閱標記 IAM 實體中的IAM 使用者指南。
15. 選擇下一頁: 檢閱，查看要新增至新使用者的群組成員資格清單。準備好繼續時，請選擇 Create user (建立使用者)。

您可以使用這個相同的程序建立更多群組和使用者，以及讓使用者能夠存取您的 AWS 帳戶資源。如需了解以政策限制使用者對特定 AWS 資源的許可，請參閱存取管理和政策範例。

註冊 AWS 並建立管理員帳戶

1. 如果您尚未擁有 AWS 帳戶，請開啟https://aws.amazon.com/。選擇 Create an AWS Account (建立 AWS 帳戶)，然後遵循線上說明。

   部分註冊程序需接收來電，並使用電話鍵盤輸入 PIN 碼。
2. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
3. 在導覽窗格中，選擇 Groups (群組)，然後選擇 Create New Group (建立新群組)。
4. 針對 Group Name (群組名稱)，輸入您群組的名稱，例如 **`Administrators`**，然後選擇 Next Step (下一步)。
5. 在政策清單中，選取 AdministratorAccess 政策旁的核取方塊。您可以使用 Filter (篩選) 功能表和 Search(搜尋) 方塊來篩選政策清單。
6. 選擇 Next Step (後續步驟)。選擇 Create Group (建立群組)，您的新群組即會顯示在 Group Name (群組名稱) 下方。

7. 在導覽窗格中，選擇 Users (使用者)，然後選擇 Create New Users (建立新使用者)。
8. 在方塊 1 中，輸入使用者名稱、清除 Generate an access key for each user (為每個使用者產生存取金鑰) 旁邊的核取方塊，然後選擇 Create (建立)。
9. 在使用者清單中，選擇您剛建立的使用者名稱 (不是核取方塊)。您可以使用 Search (搜尋) 方塊來搜尋使用者名稱。
10. 選擇 Groups (群組) 標籤，然後選擇 Add User to Groups (新增使用者到群組)。
11. 選取管理員群組旁的核取方塊，然後選擇 Add to Groups (新增至群組)。
12. 選擇 Security Credentials (安全登入資料) 標籤。在 Sign-In Credentials (登入資料) 下，選擇 Manage Password (管理密碼)。
13. 選取 Assign a custom password (指派自訂密碼)，在 Password (密碼) 及 Confirm Password (確認密碼) 中輸入密碼，然後選擇 Apply (套用)。

安裝適用於視窗的 Kinesis 代理程式

您可以使用三種方式在 Windows 上安裝 Windows 專用 Kinesis 代理程式：

- 使用 MSI (Windows 安裝程式套件) 進行安裝。
- 從安裝AWS Systems Manager，這是一組管理伺服器和桌面平台的服務。
- 執行 PowerShell 指令碼。

> **Note**
>
> 下列說明偶爾會使用 KinesisTap 和 AWSKinesisTap 字詞。這些字詞都和代理程式都和 Windows 版 Kinesis 代理程式相同的項目，但您必須在執行這些指示時指定他們。

使用 MSI 安裝適用於視窗的 Kinesis 代理程式

您可以從 Windows MSI 套件下載最新版的 Kinesis 代理程式。GitHub 上的動態代理程式視窗存儲庫。下載 MSI 之後，請使用 Windows 啟動它，並依照安裝程式提示進行。安裝後，您可以像執行任何 Windows 應用程式一樣解除安裝。

或者，您也可以使用檢查命令以無訊息方式安裝、開啟記錄並解除安裝，如以下範例所示。Replace*AWSKinesisTap.1.1.216.4.msi* with the appropriate version of Kinesis Agent for Windows for your application.

若要以無訊息方式安裝適用於 Windows 的 Kinesis 代理程式：

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

若要將安裝訊息記錄在名為***logfile.log***：

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

如果要使用命令提示字元解除安裝 Windows 專用的 Kinesis 代理程式：

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

使用 AWS Systems Manager 安裝適用於視窗的 Kinesis 代理程式

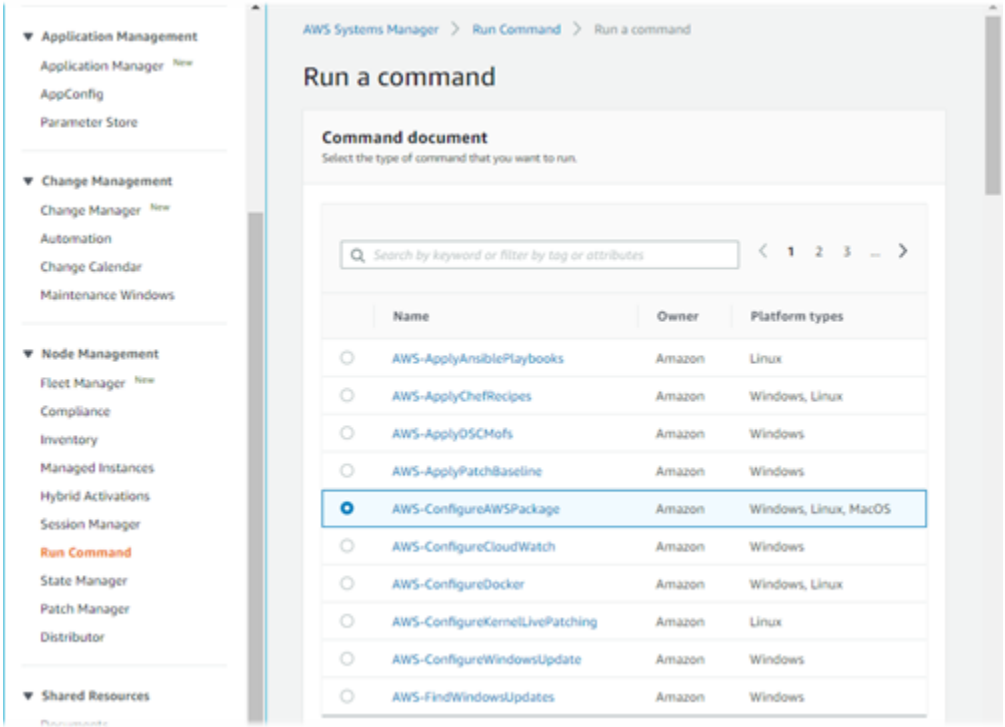
依照下列步驟，使用 Systems Manager Run Command 來安裝 Windows 適用的 Kinesis 代理程式。如需執行命令的詳細資訊，請參閱AWS Systems Manager Run Command中的AWS Systems Manager 使用指南。除了使用 Systems Manager Run Command 之外，您也可以使用 Systems Manager維護時段和狀態管理員來自動化 Windows 專用 Kinesis 代理程式的部署。

> **Note**
>
> 適用於 Windows 版 Kinesis 代理程式的 Systems Manager 安裝可在AWS Systems Manager除下列項目外：
>
> - cn-north-1
> - cn-northwest-1
> - 所有 AWS GovCloud 區域。

使用 Systems Manager 安裝適用於 Windows 的 Kinesis 代理程式

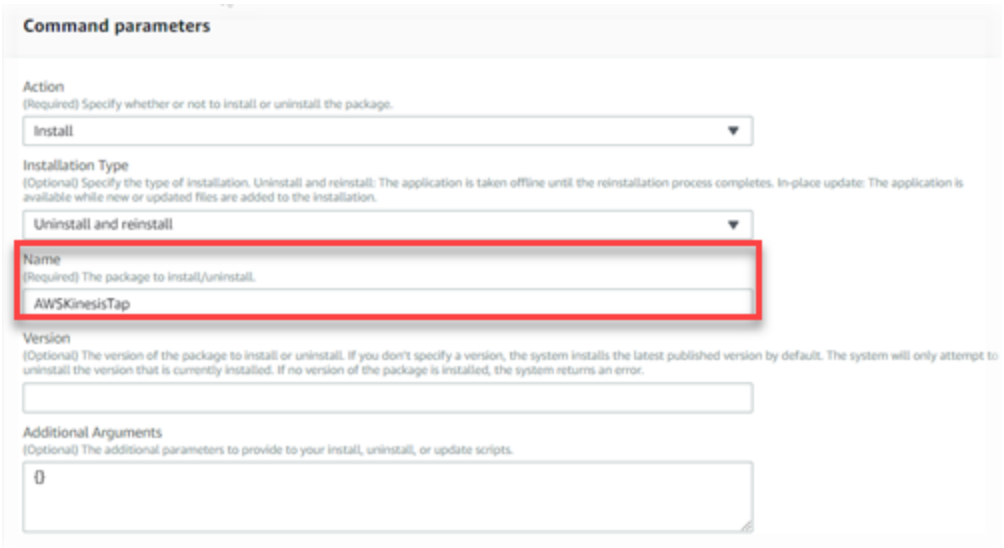
1. 確認已在您希望在安裝 Windows 專用 Kinesis 代理程式的執行個體上安裝 SSM 代理程式 2.2.58.0 或更新版本。如需詳細資訊，請參閱「」在 Windows 執行個體上安裝及設定 SSM 代理程式中的AWS Systems Manager 使用指南。
2. 開啟 AWS Systems Manager 主控台，位於https://console.aws.amazon.com/systems-manager/。
3. 從導覽窗格的節點管理中，選擇Run Command，然後選擇Run Command。

4. 從指令文件清單中，選取AWS ConfigureAWSPackagedocument.



5. DER命令參數，用於名稱，輸入智慧型手機。將其他設定保持為預設值。

> **Note**
>
> 離開版本空白，以指定 AWSinesiStap 套件的最新版本。或者，您也可以選擇輸入要安裝的特定版本。



6. DERTargets (目標)下，指定要執行命令的執行個體。您可以選擇根據與執行處理相關聯的標籤來指定執行處理，也可以手動選擇執行處理，也可以指定包含執行處理的資源群組。
7. 將所有其他設定保持為預設值，並選擇執行。

為視窗安裝 PowerShell 代理程式

使用文字編輯器將下列命令複製到檔案中，並儲存為 PowerShell 指令碼。我們使用`InstallKinesisAgent.ps1`在下列範例中。

```
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
}
```

開啟提高權限執行的命令提示視窗。在下載檔案的目錄中，使用下列命令來執行指令碼：

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

若要安裝特定版本的 Kinesis 代理程式，請將`-version`選項：

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

Replace*version*使用有效的 Kinesis 代理程式。如需版本資訊，請參閱GitHub 上的動態代理程式視窗存儲庫。

有許多部署工具可遠端執行 PowerShell 指令碼。它們可用來在伺服器或桌面平台叢集上自動化安裝 Windows 專用 Kinesis Agent。

設定和啟動視窗的 Kinesis 代理程式

在安裝 Windows 專用 Kinesis 代理程式之後，您必須設定及啟動代理程式。之後，並不需要更進一步的操作介入。

設定及啟動 Windows 版 Kinesis 代理程式

1. 建立及部署 Windows 組態檔案的 Kinesis 代理程式。此檔案會設定來源、接收和管道，以及其他全域組態項目。

   如需 Windows 專用 Kinesis 代理程式組態的詳細資訊，請參閱設定適用於微軟視窗的 Amazon Kinesis。

   如需您可以自訂及安裝的完整組態檔案範例，請參閱適用於視窗設定範例的 Kinesis 代理程式。
2. 開啟提高權限執行的 PowerShell 命令提示視窗，然後使用以下 PowerShell 命令啟動適用於 Windows 的 Kinesis 代理程式：

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

設定適用於微軟視窗的 Amazon Kinesis

在啟動微軟視窗的 Amazon Kinesis 代理程式前，您必須建立一個組態檔案並加以部署。組態檔案可提供收集、轉換 Windows 伺服器和桌上型電腦的資料並串流到各種 AWS 服務所需的資訊。組態檔案可定義幾組來源、目的地和管道 (連接來源和目的地)，以及選用的轉換。

組態檔案命名為`appsettings.json`。將此檔案部署至 `%PROGRAMFILES%\Amazon\AWSKinesisTap`。

主題

基礎組態結構

微軟視窗組態檔案的基本架構 Amazon Kinesis 一種 JSON 文件，其中包含以下範本：

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

- `Sources` 值是一或多個 來源宣告。
- `Sinks` 值是一或多個 目的地宣告。
- `Pipes` 值是一或多個 管道宣告。

如需 Windows 來源、管道和目的地概念的詳細資訊，請參閱微軟視窗概念的 Amazon Kinesis Kinesis 代理程式。

下面的例子是一個完整的appsettings.json組態檔案，其會設定以將 Windows 用的 Kinesis 代理程式串流給作業系統日誌事件。

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

如需每種宣告的資訊，請參閱下列各節：

- 來源宣告
- 目的地宣告
- 管道宣告

組態區分大小寫

JSON 格式的檔案通常會區分大小寫，因此您應該假設 Windows 組態檔案中的所有索引鍵和值也會區分大小寫。appsettings.json 組態檔案中有些鍵和值不區分大小寫，例如：

- 目的地 Format 鍵/值對的值。如需詳細資訊，請參閱 目的地宣告。

- 來源 SourceType 鍵/值對、目的地 SinkType 鍵/值對，以及管道和外掛程式 Type 鍵/值對的值。
- DirectorySource 來源 RecordParser 鍵/值對的值。如需詳細資訊，請參閱 DirectorySource 組態。
- 來源 InitialPosition 鍵/值對的值。如需詳細資訊，請參閱 書籤組態。
- 變數替換的字首。如需詳細資訊，請參閱 設定目的地變數替換。

來源宣告

在適用於微軟視窗的 Amazon Kinesis Kinesis 代理程式中，來源宣告描述應收集的日誌、事件及指標資料位置及種類。他們也可以選擇性地指定剖析該資料所需的資訊，以進行轉換。以下各節說明適用於 Windows 專用 Kinesis 代理程式中可用之內建來源類型的組態。由於 Windows 適用的 Kinesis 代理程式是可延伸的，因此您可以新增自訂來源類型。每個來源類型通常都需要組態物件中具有與該來源類型相關的特定鍵/值對。

所有來源宣告都至少必須包含下列鍵/值對：

`Id`

唯一字串，可在組態檔案中識別特定來源物件。

`SourceType`

此來源物件的來源類型名稱。來源類型會指定此來源物件所收集日誌、事件或指標資料的來源。它也會控制來源可宣告的其他層面為何。

如需使用不同類型來源宣告完整組態檔案的範例，請參閱從各種來源串流到 Kinesis Data Streams。

主題

- DirectorySource 組態
- ExchangeLogSource 組態
- W3SVCLogSource 組態
- UlsSource 組態
- WindowsEventLogSource 組態
- 動態指標視窗選取來源組態
- WindowsETWEventSource 組態
- WindowsPerformanceCounterSource 組態

- Windows 內建指標來源適用的 Kinesis 代理程式
- 適用於視窗度量的 Kinesis 代理程式清單
- 書籤組態

DirectorySource 組態

Overview

DirectorySource 來源類型會從存放在指定目錄中的檔案收集日誌。由於日誌檔案有許多不同的格式，DirectorySource 宣告可讓您指定日誌檔案中資料的格式。然後，您可以將日誌內容轉換成標準格式 (例如 JSON 或 XML)，再串流至各種 AWS 服務。

以下是範例 DirectorySource 宣告：

```
{
    "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"
}
```

所有 DirectorySource 宣告都可提供下列鍵/值對：

SourceType

必須是常值字串 "DirectorySource" (必要)。

Directory

指向包含日誌檔案目錄的路徑 (必要)。

`FileNameFilter`

選擇性地根據萬用字元檔案命名模式，限制收集日誌資料目錄中的檔案組。如果您有多個記錄檔名稱模式，此功能可讓您使用單一DirectorySource，如下列範例所示，所示。

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

系統管理員有時會先壓縮記錄檔，然後再封存記錄檔。如果您指定"*.*"中的FileNameFilter，則現在會排除已知的壓縮檔。此功能可防止.zip、.gz，以及.bz2檔案不小心被串流處理。若沒有指定此鍵/值對，預設會收集目錄中所有檔案的資料。

`IncludeSubdirectories`

指定將子目錄監視為受作業系統限制的任意深度。此功能對於監控具有多個網站的網頁伺服器非常有用。您也可以使用IncludeDirectoryFilter屬性來監視篩選器中指定的某些子目錄。

`RecordParser`

指定 DirectorySource 來源類型應剖析指定目錄中所找到日誌檔案的方式。此鍵/值對是必要項目，其有效值如下所示：

- SingleLine— 記錄檔的每一行都是記錄記錄。
- SingleLineJson— 記錄檔的每一行都是 JSON 格式的記錄記錄。此剖析器在您希望使用物件裝飾，將額外的鍵/值對新增到 JSON 時很有用。如需詳細資訊，請參閱 設定目的地宣告。如需使用 SingleLineJson 記錄剖析器的範例，請參閱教學課程：使用適用於視窗的 Kinesis 代理程式將 JSON 日誌檔串流到 Amazon S3。
- Timestamp— 一或多行可以包含記錄檔記錄。日誌記錄會以時間戳記開頭。此選項會要求指定 TimestampFormat 鍵/值對。
- Regex— 每筆記錄以符合特定規則運算式的文字開頭。此選項會要求指定 Pattern 鍵/值對。
- SysLog— 表示記錄檔是寫入syslog標準格式。日誌檔案會根據該規格剖析至記錄。
- Delimited— 更簡單的 Regex 記錄剖析器版本，其中記錄中的資料項目會以一致的分隔符號分隔。此選項較容易使用且比 Regex 剖析器的執行速度更快，當此選項可用時此為慣用選項。使用此選項時，您必須指定 Delimiter 鍵/值對。

`TimestampField`

指定哪個 JSON 欄位包含記錄的時間戳記。此項目僅搭配 SingleLineJson RecordParser 使用。這個鍵/值對是選用的。若沒有指定，Windows 專用 Kinesis 代理程式會使用針對時間戳記讀取記錄時的時間。指定此鍵-值組的一個優點是針對 Windows 而言，Kinesis 代理程式產生的延遲統計資料會較為準確。

`TimestampFormat`

指定剖析與記錄相關聯日期和時間的方式。此值為字串 epoch 或 .NET 日期/時間格式字串。若值是 epoch，則時間會根據 UNIX Epoch 時間剖析。如需 UNIX Epoch 時間的詳細資訊，請參閱 Unix time。如需 .NET 日期/時間格式字串的詳細資訊，請參閱 Microsoft .NET 文件中的 Custom Date and Time Format Strings)。此鍵/值對只有在指定 `Timestamp` 記錄剖析器，或是 `SingleLineJson` 記錄剖析器隨著 `TimestampField` 鍵/值對一同指定時才是必要項目。

`Pattern`

指定必須比對潛在多行記錄中第一行的規則表達式。此鍵/值對只針對 `Regex` 記錄剖析器為必要項目。

`ExtractionPattern`

指定應使用具名群組的規則表達式。記錄會使用此規則表達式剖析，而具名群組則會構成剖析記錄的欄位。這些欄位接著會用來做為建構 JSON 或 XML 物件或文件的基礎，並接著由接收串流至不同各種 AWS 服務。此鍵/值對是選用的，並且可透過Regex記錄解析器和時間戳記解析器。

`Timestamp` 群組名稱會特別進行處理，因為它會向 `Regex` 剖析器指出每個日誌檔案中哪些欄位包含每一筆記錄的日期和時間。

`Delimiter`

指定分隔每一筆日誌記錄中每個項目的字元或字串。此鍵/值對必須 (且只能) 搭配 `Delimited` 記錄剖析器使用。請使用雙字元序列 `\t` 來表示定位字元。

`HeaderPattern`

指定規則表達式，比對日誌檔案中包含記錄標頭組的行。若日誌檔案不包含任何標頭資訊，請使用 `Headers` 鍵/值對來指定隱含標頭。`HeaderPattern` 鍵/值對是選用的，只針對 `Delimited` 記錄剖析器有效。

> **Note**
>
> 資料行的空白 (長度 0) 標頭項目會造成從 `DirectorySource` 剖析輸出的最終輸出中篩選出該資料行的資料。

`Headers`

指定使用指定分隔符號剖析之資料的資料行名稱。此鍵/值對是選用的，只針對 `Delimited` 記錄剖析器有效。

> **Note**
>
> 資料行的空白 (長度 0) 標頭項目會造成從 DirectorySource 剖析輸出的最終輸出中篩選出該資料行的資料。

`RecordPattern`

指定規則表達式，識別日誌檔案中包含記錄資料的行。除了由 HeaderPattern 所識別的選用標頭之外，不符合指定 RecordPattern 的行也會在記錄處理期間遭到忽略。此鍵/值對是選用的，只針對 Delimited 記錄剖析器有效。若沒有提供此項目，則根據預設會將任何不符合選用 HeaderPattern 或選用 CommentPattern 的行視為包含可剖析記錄資料的行。

`CommentPattern`

指定規則表達式，識別日誌檔案中應在剖析日誌檔案內資料前排除的行。此鍵/值對是選用的，只針對 Delimited 記錄剖析器有效。若沒有提供此項目，則根據預設會將任何不符合選用 HeaderPattern 的行視為包含可剖析記錄資料的行，除非有指定 RecordPattern。

`TimeZoneKind`

指定是否應將日誌檔案中的時間戳記視為本地時區或 UTC 時區。此為選用項目，其預設為 UTC。此鍵/值對有效的值僅限 Local 或 UTC。若沒有指定 TimeZoneKind，或是其值為 UTC，則永遠不會改變時間戳記。時間戳記會轉換為 UTC，當TimeZoneKind值為Local，且接收時間戳記的接收為 CloudWatch Logs，或是將剖析記錄傳送到其他接收。內嵌在訊息中的日期和時間不會轉換。

`SkipLines`

指定此項目時，會控制記錄剖析發生前於每個日誌檔案開頭要忽略的行數。此為選用項目，預設值為 0。

編碼

根據預設，Windows 版 Kinesis 代理程式可以自動偵測來自位元組標記的編碼。不過，在某些較舊的 Unicode 格式上，自動編碼可能無法正常運作。下列範例會指定資料流微軟 SQL 伺服器記錄檔所需的編碼。

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

如需編碼名稱的清單，請參閱編碼清單在微軟 .NET 文檔中。

解壓縮規則選項

您可以使用ExtractionRegexOptions以簡化規則表達式。這個鍵/值對是選用的。預設值為"None"。

以下範例指定"."運算式符合任何字元，包括\r\n。

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

如需解壓縮規則選項的可能欄位清單，請參閱規則表達式選項列舉在微軟 .NET 文檔中。

Regex 記錄剖析器

您可以使用 Regex 記錄剖析器，搭配 TimestampFormat、Pattern 和 ExtractionPattern 鍵/值對剖析非結構化文字日誌。例如，假設您的日誌檔案看起來如下：

```
[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'
```

您可以為 Pattern 鍵/值對指定以下規則表達式，協助將日誌檔案分成個別日誌記錄：

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

這個規則表達式符合以下序列：

1. 要評估的字串開頭。
2. 以方括弧包圍的一或多個文字字元。
3. 以方括弧包圍的時間戳記。時間戳記符合以下序列：
   a. 四位數年份
   b. 斜線
   c. 二位數月份
   d. 斜線

e. 二位數日期
f. 空白字元
g. 二位數小時
h. 冒號
i. 二位數分鐘
j. 冒號
k. 二位數秒鐘
l. 句號
m. 三位數毫秒

您可以為 TimestampFormat 鍵/值對指定以下格式，將文字時間戳記轉換成日期和時間：

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

您可以使用以下規則表達式來透過 ExtractionPattern 鍵/值對擷取日誌記錄的欄位。

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

此規則表達式符合以下序列中的群組：

1. Severity— 以方括弧包圍的一或多個文字字元。
2. TimeStamp— 請參閱上一個描述以取得時間戳記。
3. 跳過三個零或多個字元的未命名方括弧序列。
4. SubSystem— 以方括弧包圍的一或多個文字字元。
5. Module— 以方括弧包圍的一或多個文字字元。
6. 跳過一個零或多個字元的未命名方括弧序列。
7. 跳過一個未命名空格。
8. Message— 以單引號括住的零個或多個字元。

以下來源宣告會合併這些規則表達式及日期時間格式，提供完整說明給 Windows 版 Kinesis Agent，用來剖析這類日誌檔案。

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

> **Note**
>
> JSON 格式檔案中的反斜線必須使用額外的反斜線逸出。

如需規則表達式的詳細資訊，請參閱 Microsoft .NET 中的 Regular Expression Language - Quick Reference。

`Delimited` 記錄剖析器

您可以使用 `Delimited` 記錄剖析器來剖析其中有一致字元序列分隔每個資料列中每個資料行的半結構化日誌及資料檔案。例如，CSV 檔案使用逗號來分隔每個資料行，TSV 檔案則使用標籤。

假設您希望剖析網路政策伺服器所產生的 Microsoft NPS Database Format 日誌檔案。這類檔案看起來與以下內容相似：

```
"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,,,,
```

以下範例 appsettings.json 組態檔案包含一個 DirectorySource 宣告，使用 Delimited 記錄剖析器來將文字剖析為物件表示。它接著會將 JSON 格式資料串流至 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"
        }
    ]
}
```

串流至 Kinesis 資料火焰軟管的 JSON 格式資料看起來與以下內容相似：

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

```
    "Acct-Session-Id": "",
    "Acct-Authentic": "",
    "Acct-Session-Time": "",
    "Acct-Input-Packets": "",
    "Acct-Output-Packets": "",
    "Acct-Terminate-Cause": "",
    "Acct-Multi-Ssn-ID": "",
    "Acct-Link-Count": "",
    "Acct-Interim-Interval": "",
    "Tunnel-Type": "",
    "Tunnel-Medium-Type": "",
    "Tunnel-Client-Endpt": "",
    "Tunnel-Server-Endpt": "",
    "Acct-Tunnel-Conn": "",
    "Tunnel-Pvt-Group-ID": "",
    "Tunnel-Assignment-ID": "",
    "Tunnel-Preference": "",
    "MS-Acct-Auth-Type": "",
    "MS-Acct-EAP-Type": "",
    "MS-RAS-Version": "",
    "MS-RAS-Vendor": "",
    "MS-CHAP-Error": "",
    "MS-CHAP-Domain": "",
    "MS-MPPE-Encryption-Types": "",
    "MS-MPPE-Encryption-Policy": "",
    "Proxy-Policy-Name": "Use Windows authentication for all users",
    "Provider-Type": "1",
    "Provider-Name": "",
    "Remote-Server-Address": "",
    "MS-RAS-Client-Name": "",
    "MS-RAS-Client-Version": ""
}
```

SysLog 記錄剖析器

針對 SysLog 記錄剖析器，從來源剖析後的輸出包含以下資訊：

屬性	類型	描述
SysLogTimeStamp	字串	來自 syslog 格式日誌檔案的原始日期和時間。

屬性	類型	描述
Hostname	字串	syslog 格式日誌檔案所在的電腦名稱。
Program	字串	產生日誌檔案的應用程式或服務名稱。
Message	字串	應用程式或服務產生的日誌訊息。
TimeStamp	字串	ISO 8601 格式的剖析後日期及時間。

以下是轉換成 JSON 的 SysLog 資料範例：

```
{
    "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

以下是 DirectorySource 來源可用的鍵/值對摘要及與這些鍵/值對相關的 RecordParser。

鍵值名稱	RecordParser	備註
SourceType	所有項目的必要項目	其值必須為 DirectorySource
Directory	所有項目的必要項目	
FileNameFilter	所有項目的選用項目	
RecordParser	所有項目的必要項目	

鍵值名稱	RecordParser	備註
TimestampField	SingleLineJson 的選用項目	
TimestampFormat	Timestamp 的必要項目；若有指定 TimestampField ，則為 SingleLineJson 的必要項目	
Pattern	Regex 的必要項目	
ExtractionPattern	Regex 的選用項目	若接收指定 json 或 xml 格式，則為 Regex 的必要項目
Delimiter	Delimited 的必要項目	
HeaderPattern	Delimited 的選用項目	
Headers	Delimited 的選用項目	
RecordPattern	Delimited 的選用項目	
CommentPattern	Delimited 的選用項目	
TimeZoneKind	識別時間戳記欄位時，Regex、Timestamp 、SysLog 和 SingleLineJson 的選用項目	
SkipLines	所有項目的選用項目	

ExchangeLogSource 組態

ExchangeLogSource 類型會用於從 Microsoft Exchange 收集日誌。Exchange 會以數種不同類型的日誌格式產生日誌。此來源類型會剖析所有格式。雖然您可以使用 DirectorySource 類型搭配

Regex 記錄剖析器剖析他們，但使用 ExchangeLogSource 會簡單許多。這是因為您不需要設計及提供日誌檔案格式的規則表達式。以下是範例 ExchangeLogSource 宣告：

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

所有 Exchange 宣告都可提供下列鍵/值對：

`SourceType`

必須是常值字串 "ExchangeLogSource" (必要)。

`Directory`

指向包含日誌檔案目錄的路徑 (必要)。

`FileNameFilter`

選擇性地根據萬用字元檔案命名模式，限制收集日誌資料目錄中的檔案組。若沒有指定此鍵/值對，則根據預設，會收集目錄中所有檔案的日誌資料。

`TimestampField`

包含記錄日期和時間的資料行名稱。此鍵/值對是選用的，且若欄位名稱是 `date-time` 或 `DateTime`，則不需要指定此項目。否則該項目為必要項目。

W3SVCLogSource 組態

`W3SVCLogSource` 類型會用於從 Windows 的 Internet Information Services (IIS) 收集日誌。

以下是範例 `W3SVCLogSource` 宣告：

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

所有 W3SVCLogSource 宣告都可提供下列鍵/值對：

SourceType

必須是常值字串 "W3SVCLogSource" (必要)。

Directory

指向包含日誌檔案目錄的路徑 (必要)。

FileNameFilter

選擇性地根據萬用字元檔案命名模式，限制收集日誌資料目錄中的檔案組。若沒有指定此鍵/值對，則根據預設，會收集目錄中所有檔案的日誌資料。

UlsSource 組態

UlsSource 類型會用於從 Microsoft SharePoint 收集日誌。以下是範例 UlsSource 宣告：

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

所有 UlsSource 宣告都可提供下列鍵/值對：

SourceType

必須是常值字串 "UlsSource" (必要)。

Directory

指向包含日誌檔案目錄的路徑 (必要)。

FileNameFilter

選擇性地根據萬用字元檔案命名模式，限制收集日誌資料目錄中的檔案組。若沒有指定此鍵/值對，則根據預設，會收集目錄中所有檔案的日誌資料。

WindowsEventLogSource 組態

`WindowsEventLogSource` 類型會用於從 Windows 事件記錄服務收集事件。以下是範例 `WindowsEventLogSource` 宣告：

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

所有 `WindowsEventLogSource` 宣告都可提供下列鍵/值對：

`SourceType`

必須是常值字串 `"WindowsEventLogSource"` (必要)。

`LogName`

事件會從指定日誌收集。常見的值包含 `Application`、`Security` 和 `System`，但您可以指定任何有效的 Windows 事件日誌名稱。這個鍵/值對是必要的。

`Query`

選擇性限制要從 `WindowsEventLogSource` 輸出的事件。若沒有指定此鍵/值對，則根據預設會輸出所有事件。如需此值語法的資訊，請參閱 Windows 文件中的 Event Queries and Event XML。如需日誌層級定義的資訊，請參閱 Windows 文件中的 Event Types。

`IncludeEventData`

在此鍵/值對的值為 `"true"` 時，選擇性地啟用從指定 Windows 事件日誌收集及串流與事件相關聯的提供者限定事件資料。只有可成功序列化的事件資料才會包含在其中。此鍵/值對是選用的，且若沒有指定，便不會收集提供者限定事件資料。

> **Note**
>
> 包含事件資料可能會大幅增加從此來源串流的資料量。事件的大小上限可為 262,143 個位元組 (包含事件資料)。

來自 `WindowsEventLogSource` 的剖析輸出包含以下資訊：

屬性	類型	描述
EventId	Int	事件類型的識別符。
Description	字串	描述事件詳細資訊的文字。
LevelDisplayName	字串	事件的類別 (錯誤 (Error)、警告 (Warning)、資訊 (Information)、成功稽核 (Success Audit)、失敗稽核 (Failure Audit) 中的其中一項)。
LogName	字串	記錄事件的位置 (典型的值為 Application 、Security 和 System，但有許多可能值)。
MachineName	字串	記錄事件的電腦。
ProviderName	字串	記錄事件的應用程式或服務。
TimeCreated	字串	事件發生的時間 (以 ISO 8601 格式呈現)。
Index	Int	項目在日誌中的位置。
UserName	字串	建立該項目的人員 (若已知的話)。
Keywords	字串	事件的類型。標準值包含 AuditFailure (失敗的安全稽核事件)、AuditSuccess (成功的安全稽核事件)、Classic (使用 RaiseEvent 函數引發的事件)、CorrelationHint (傳輸事件)、SQM (服務品質機制事件)、WDI Context (Windows 診斷基礎設施內容事件)，以及 WDI Diag (Windows 診斷基礎設施診斷事件)。
EventData	物件清單	與日誌事件相關的選用提供者限定額外資料。只有在 IncludeEventData

屬性	類型	描述
		鍵/值對的值為 "true" 時，才會包含此項目。

以下是轉換成 JSON 的範例事件：

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

動態指標視窗選取來源組態

WindowsEventLogPollingSource會使用輪詢型機制，從事件記錄檔中收集符合設定參數的所有新事件。輪詢間隔會在 100 毫秒到 5000 毫秒之間動態更新，視上次輪詢期間收集的事件數目而定。以下是範例 WindowsEventLogPollingSource 宣告：

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

所有 WindowsEventLogPollingSource 宣告都可提供下列鍵/值對：

`SourceType`

必須是常值字串 `"WindowsEventLogPollingSource"` (必要)。

`LogName`

指定記錄。有效選項為：`Application`、`Security`、`System`或其他有效的記錄檔。

`IncludeEventData`

選用。時機`true`，指定包含以 JSON 和 XML 形式串流處理時的額外 EventData。預設為 `false`。

`Query`

選用。Windows 事件記錄檔支援使用 XPath 運算式查詢事件，您可以使用`Query`。如需詳細資訊，請參閱「」事件查詢和事件 XML在微軟文檔中。

`CustomFilters`

選用。以分號分隔的篩選器清單 (`;`。您可以指定以下篩選條件。

`ExcludeOwnSecurityEvents`

排除由 Kinesis 代理程式針對 Windows 本身產生的安全性事件。

WindowsETWEventSource 組態

`WindowsETWEventSource` 類型會使用名為 Windows 事件追蹤 (ETW) 的功能，用來收集應用程式和服務事件追蹤。如需詳細資訊，請參閱 Windows 文件中的 Event Tracing。

以下是範例 `WindowsETWEventSource` 宣告：

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

所有 `WindowsETWEventSource` 宣告都可提供下列鍵/值對：

`SourceType`

必須是常值字串 `"WindowsETWEventSource"` (必要)。

`ProviderName`

指定要用來收集追蹤事件的事件提供者。此項目必須是已安裝提供者的有效 ETW 提供者名稱。若要判斷已安裝哪些提供者，請在 Windows 命令提示視窗中執行以下內容：

```
logman query providers
```

`TraceLevel`

指定應收集的追蹤事件類別。允許值包含 `Critical`、`Error`、`Warning`、`Informational` 及 `Verbose`。明確的意義取決於所選取的 ETW 提供者。

`MatchAnyKeyword`

此值是 64 位元的數字，其中每個位元都代表一個個別關鍵字。每個關鍵字都描述了要收集事件類別。如需了解支援的關鍵字及其值，以及他們與 `TraceLevel` 相關的方式，請參閱該提供者的文件。例如，如需 CLR ETW 提供者的資訊，請參閱 Microsoft .NET Framework 文件中的 CLR ETW Keywords and Levels。

在先前的範例中，32768 (0x00008000) 代表 CLR ETW 提供者的 `ExceptionKeyword`，指示提供者收集拋出的異常相關資訊。雖然 JSON 並不原生支援十六進位常數，您可以藉由將他們置放在字串內，來為 `MatchAnyKeyword` 指定他們。您也可以指定數個常數，並以逗號分隔。例如，使用以下內容來指定 `ExceptionKeyword` 和 `SecurityKeyword` (0x00000400)：

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

為了確保為提供者啟用所有指定的關鍵字，多個關鍵字值會使用 OR 合併，並傳遞給該提供者。

來自 `WindowsETWEventSource` 的輸出包含每個事件的以下資訊：

屬性	類型	描述
`EventName`	字串	發生的事件種類。

屬性	類型	描述
ProviderName	字串	偵測到事件的提供者。
FormattedMessage	字串	事件的文字摘要。
ProcessID	Int	報告事件的程序。
ExecutingThreadID	Int	程序中報告事件的執行緒。
MachineName	字串	報告事件的桌面平台或伺服器名稱。
Payload	Hashtable	包含字串鍵及任何物件類型做為值的資料表。鍵是承載項目名稱，值則是承載項目的值。承載會取決於提供者。

以下是轉換成 JSON 的範例事件：

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

WindowsPerformanceCounterSource 組態

WindowsPerformanceCounterSource 類型會從 Windows 收集效能計數器指標。以下是範例 WindowsPerformanceCounterSource 宣告：

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

所有 WindowsPerformanceCounterSource 宣告都可提供下列鍵/值對：

SourceType

必須是常值字串 "WindowsPerformanceCounterSource" (必要)。

`Categories`

指定要從 Windows 收集的一組效能計數器指標群組。每個指標群組都包含下列鍵/值對：

`Category`

指定要收集的計數器指標組 (必要)。

`Instances`

當每個物件都具有一組唯一的效能計數器時，指定您感興趣的物件組。例如，當類別是 `LogicalDisk` 時，每個磁碟機都會有一組效能計數器。這個鍵/值對是選用的。您可以使用萬用字元 * 和 ? 來比對多個執行個體。若要彙整所有執行個體的值，請指定 `_Total`。

您也可以使用`InstanceRegex`，它接受包含*萬用字元做為執行個體名稱的一部分。

`Counters`

指定要為指定類別收集何種指標。這個鍵/值對是必要的。您可以使用萬用字元 * 和 ? 來比對多個計數器。您可以只使用名稱指定 `Counters`，或是使用名稱和單位。若沒有指定計數器單位，Windows 版 Kinesis Agent 會嘗試從名稱推斷單位。若這些推斷不正確，則可以明確指定單位。若您希望的話，您可以變更 `Counter` 名稱。計數器更複雜的表示是使用下列鍵/值對的物件：

`Counter`

計數器的名稱。這個鍵/值對是必要的。

`Rename`

向接收呈現的計數器名稱。這個鍵/值對是選用的。

`Unit`

與計數器關聯值的意涵。有效單位名稱的完整清單，請參閱MetricDatum中的Amazon CloudWatch API 參考。

以下是複雜計數器規格的範例：

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

WindowsPerformanceCounterSource只能搭配指定 Amazon CloudWatch 接收的管道使用。如果 Windows 內建指標的 Kinesis 代理程式也會串流至 CloudWatch，請使用不同的接收。在服務啟動後檢查 Kinesis 代理程式的 Windows 日誌，判斷已為計數器推斷何種單位，當尚未在WindowsPerformanceCounterSource宣告。使用 PowerShell 來判斷類別、執行個體和計數器的有效名稱。

若要查看所有類別的資訊 (包含與計數器組相關聯的計數器)，請在 PowerShell 視窗中執行此命令：

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

若要判斷計數器組中每個計數器有哪些可用的執行個體，請在 PowerShell 視窗中執行與以下範例相似的命令：

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

Counter 參數的值應為先前 Get-Counter -ListSet 命令呼叫所列出 PathsWithInstances 成員中的其中一個路徑。

Windows 內建指標來源適用的 Kinesis 代理程式

除了一般度量來源 (例如WindowsPerformanceCounterSource類型 (請參閱WindowsPerformanceCounterSource 組態)，CloudWatch 接收類型可以接收來自特殊來源的指標，而這種特殊來源會收集適用 Kinesis Windows 本身的相關指標。適用於 Windows 的 Kinesis 代理程式指標也可於KinesisTap類別的 Windows 效能計數器。

所以此MetricsFilterCloudWatch 接收宣告會指標要從內建的 Kinesis 代理程式 (適用於 Windows 指標來源的內建 Kinesis 代理程式) 串流至 CloudWatch 的指標。其值是一個字串，包含一或多個以分號分隔的篩選條件表達式；例如：

"MetricsFilter": "FilterExpression1;FilterExpression2"

符合一或多個篩選條件表達式的指標會串流至 CloudWatch。

單一執行個體指標的本質是全域的，而不會繫結於特定來源或接收。多個執行個體指標則會根據來源或接收宣告 Id，以維度的方式呈現。每個來源或接收類型都可以有不同的指標組。

如需 Windows 指標名稱的內建 Kinesis 代理程式清單，請參閱適用於視窗度量的 Kinesis 代理程式清單。

針對單一執行個體指標，篩選條件表達式是指標的名稱；例如：

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

針對多個執行個體指標，篩選條件表達式是指標的名稱，一個句號 (.)，然後是產生該指標來源或接收宣告的 Id。例如，假設有一個具備 MyFirehose Id 的接收宣告：

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

您可以使用旨在區分單一及多個執行個體指標的特殊萬用字元模式。

- 星號 (*) 會比對零或多個字元 (句號 (.) 除外)。
- 問號 (?) 則會比對一個字元 (句號除外)。
- 任何其他字元都只會與自身比對。
- _Total 是一種特殊字符，會彙整各維度間所有相符的多個執行個體值。

以下範例會比對所有單一執行個體指標：

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

因為星號不會比對句號字元，因此只會包含單一執行個體指標。

以下範例會比對所有多個執行個體指標：

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

以下範例會比對所有指標 (單一及多個)：

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

以下範例會彙整所有來源及接收的所有多個執行個體指標：

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

以下範例會彙整所有 Kinesis Data Firehose 接收的所有 Kinesis 資料火管指標：

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

以下範例會比對所有單一及多個執行個體錯誤指標：

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

以下範例會比對彙整自所有來源及接收的所有不可復原錯誤指標：

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

如需如何指定使用適用於 Windows 內建指標來源的 Kinesis 代理程式管道的資訊，請參閱[設定 Windows 度量管道的 Kinesis 代理程式]。

適用於視窗度量的 Kinesis 代理程式清單

以下是適用於 Windows 專用 Kinesis 代理程式可用的單一執行個體及多個執行個體指標清單。

單一執行個體指標

以下是可用的單一執行個體指標：

`KinesisTapBuildNumber`

適用於視窗的 Kinesis 代理程式版本號碼。

`PipesConnected`

有多少管道已將他們的來源成功連線至其接收。

`PipesFailedToConnect`

有多少管道將他們的來源連線至其接收失敗。

`SinkFactoriesFailedToLoad`

有多少接收類型並未成功載入至 Windows 的 Kinesis 代理程式。

`SinkFactoriesLoaded`

有多少接收類型已成功載入至 Windows 的 Kinesis 代理程式。

`SinksFailedToStart`

有多少接收並未成功開始 (通常是因為不正確的接收宣告)。

`SinksStarted`

有多少接收已成功開始。

`SourcesFailedToStart`

有多少來源並未成功開始 (通常是因為不正確的來源宣告)。

`SourcesStarted`

有多少來源已成功開始。

`SourceFactoriesFailedToLoad`

有多少來源類型並未成功載入 Windows 專用 Kinesis 代理程式。

`SourceFactoriesLoaded`

在 Windows 專用 Kinesis 代理程式中，成功載入了多少來源類型。

多個執行個體指標

以下是可用的多個執行個體指標：

DirectorySource 指標

`DirectorySourceBytesRead`

此 `DirectorySource` 的間隔期間讀取了多少位元組。

`DirectorySourceBytesToRead`

Windows 適用的 Kinesis 代理程式尚未由 Windows 專用的 Kinesis 代理程式讀取了多久已知位元組數。

`DirectorySourceFilesToProcess`

要檢查的已知檔案有多少，但尚未由 Kinesis 代理程式檢查。

`DirectorySourceRecordsRead`

此 `DirectorySource` 的間隔期間已讀取了多少記錄。

WindowsEventLogSource 指標

`EventLogSourceEventsError`

有多少 Windows 事件日誌事件並未成功讀取。

`EventLogSourceEventsRead`

有多少 Windows 事件日誌事件已成功讀取。

KinesisFirehose 接收指標

`KinesisFirehoseBytesAccepted`

間隔期間已接受了多少位元組。

`KinesisFirehoseClientLatency`

記錄產生與記錄串流至 Kinesis Data Firehose 服務之間經過了多久時間。

`KinesisFirehoseLatency`

Kinesis Data Firehose 服務記錄串流的開始與結束之間經過了多久時間。

`KinesisFirehoseNonrecoverableServiceErrors`

即使經過了重試，記錄仍無法在不出現錯誤的情況下傳送至 Kinesis Data Firehose 服務的次數。

`KinesisFirehoseRecordsAttempted`

嘗試串流至 Kinesis Data Firehose 服務的記錄數。

`KinesisFirehoseRecordsFailedNonrecoverable`

即使經過了重試，仍無法成功串流至 Kinesis Data Firehose 服務的記錄數。

`KinesisFirehoseRecordsFailedRecoverable`

經過重試之後，成功串流至 Kinesis Data Firehose 服務的記錄數。

`KinesisFirehoseRecordsSuccess`

在不進行重試的情況下，成功串流至 Kinesis Data Firehose 服務的記錄數。

`KinesisFirehoseRecoverableServiceErrors`

經過重試之後，記錄成功傳送至 Kinesis Data Firehose 服務的次數。

KinesisStream 指標

`KinesisStreamBytesAccepted`

間隔期間已接受了多少位元組。

`KinesisStreamClientLatency`

記錄產生與記錄串流至 Kinesis Data Streams 服務之間經過了多久時間。

`KinesisStreamLatency`

Kinesis Data Streams 服務記錄串流的開始與結束之間經過了多久時間。

`KinesisStreamNonrecoverableServiceErrors`

即使經過了重試，記錄仍無法在不出現錯誤的情況下傳送至 Kinesis Data Streams 服務的次數。

`KinesisStreamRecordsAttempted`

嘗試串流至 Kinesis Data Streams 服務的記錄數。

`KinesisStreamRecordsFailedNonrecoverable`

即使經過了重試，仍無法成功串流至 Kinesis Data Streams 服務的記錄數。

`KinesisStreamRecordsFailedRecoverable`

經過重試之後，成功串流至 Kinesis Data Streams 服務的記錄數。

`KinesisStreamRecordsSuccess`

在不進行重試的情況下，成功串流至 Kinesis Data Streams 服務的記錄數。

`KinesisStreamRecoverableServiceErrors`

經過重試之後，記錄成功傳送至 Kinesis Data Streams 服務的次數。

CloudWatchLog 指標

`CloudWatchLogBytesAccepted`

間隔期間已接受了多少位元組。

`CloudWatchLogClientLatency`

記錄產生與記錄串流至 CloudWatch Logs 服務之間經過了多久時間。

`CloudWatchLogLatency`

CloudWatch Logs 服務記錄串流的開始與結束之間經過了多久時間。

`CloudWatchLogNonrecoverableServiceErrors`

即使經過了重試，記錄仍無法在不出現錯誤的情況下傳送至 CloudWatch Logs 服務的次數。

`CloudWatchLogRecordsAttempted`

嘗試串流至 CloudWatch 服務的記錄數。

`CloudWatchLogRecordsFailedNonrecoverable`

即使經過了重試，仍無法成功串流至 CloudWatch Logs 服務的記錄數。

`CloudWatchLogRecordsFailedRecoverable`

經過重試之後，成功串流至 CloudWatch 服務的記錄數。

`CloudWatchLogRecordsSuccess`

在不進行重試的情況下，成功串流至 CloudWatch Logs 服務的記錄數。

`CloudWatchLogRecoverableServiceErrors`

經過重試之後，記錄成功傳送至 CloudWatch Logs 服務的次數。

CloudWatch Metrics

`CloudWatchLatency`

CloudWatch 服務指標串流的開始與結束之間平均經過了多久時間。

`CloudWatchNonrecoverableServiceErrors`

即使經過了重試，指標仍無法在不出現錯誤的情況下傳送至 CloudWatch 服務的次數。

`CloudWatchRecoverableServiceErrors`

指標在不出現錯誤的情況下傳送至 CloudWatch 服務的次數。

`CloudWatchServiceSuccess`

不進行重試，指標在不出現錯誤的情況下傳送至 CloudWatch 服務的次數。

書籤組態

根據預設，Windows 的 Kinesis 代理程式會將日誌記錄傳送至代理程式啟動後建立的接收。有時候傳送較早的日誌記錄會很有用，例如：在自動更新期間停止時所建立的日誌記錄。書籤功能會追蹤已傳送至接收的記錄。當 Windows 適用的 Kinesis 代理程式處於書籤模式並啟動時，它會將所有在 Windows 適用的 Kinesis 代理程式停止之後建立的日誌記錄，與任何之後建立的日誌記錄一同傳送。若要控制此行為，檔案類型來源宣告可以選擇性的包含下列鍵/值對：

`InitialPosition`

指定書籤的初始狀態。可能的值如下：

`EOS`

指定串流結束 (EOS)。只有在代理程式執行中時建立的日誌記錄，才會傳送至接收。

`0`

所有可用的日誌記錄及事件都會在初始時傳送。之後便會建立書籤，確保每個新的日誌記錄及在書籤建立後建立的事件最後都會傳送，無論 Kinesis 否在執行中。

`Bookmark`

書籤會在最新的日誌記錄或事件之後初始化。之後便會建立書籤，確保每個新的日誌記錄及在書籤建立後建立的事件最後都會傳送，無論 Kinesis 否在執行中。

書籤預設為皆啟用。檔案儲存在`%ProgramData%\Amazon\KinesisTap`目錄。

`Timestamp`

傳送 `InitialPositionTimestamp` 值 (定義如下) 之後建立的日誌記錄和事件。之後便會建立書籤，確保每個新的日誌記錄及在書籤建立後建立的事件最後都會傳送，無論 Kinesis 否在執行中。

`InitialPositionTimestamp`

指定您希望的最早日誌記錄或事件時間戳記。請只在 `InitialPosition` 具有 `Timestamp` 值時才指定此鍵/值對。

`BookmarkOnBufferFlush`

此設定可新增至任何可收藏的來源。當設定為`true`，可確保書籤更新只有在接收器成功將事件傳送到 AWS 時才會進行。您只能將單一接收器訂閱至來源。如果您要將記錄傳送至多個目的地，請複製來源，以避免資料遺失的潛在問題。

若 Windows 版 Kinesis Agent 已停止很長一段時間，您可能需要刪除那些書籤，因為已標記為書籤的日誌記錄及事件可能已不存在。指定 source id 的書籤檔案位於 %PROGRAMDATA%\Amazon\AWSKinesisTap*source id*.bm 中。

書籤無法在重新命名或截斷的檔案上運作。因為 ETW 事件及效能計數器的特性，他們無法標記為書籤。

目的地宣告

「目的地宣告」可指定日誌、事件和指標應該於何處、以任何形式傳送到各種 AWS 服務。下列各節說明了適用於微軟視窗的 Amazon Kinesis 代理程式可用內建目的地類型的組態。由於 Windows 適用的 Kinesis 代理程式是可延伸的，因此您可以新增自訂目的地類型。在每個目的地類型的相關組態宣告中，目的地類型通常需要唯一的鍵/值對。

所有目的地宣告都可以包含下列鍵/值對：

`Id`

可識別組態檔案中特定目的地的唯一字串 (必要)。

`SinkType`

此目的地的目的地類型名稱 (必要)。目的地類型可指定此目的地要串流的日誌、事件或指標資料目標。

`AccessKey`

指定要使用的 AWS 存取金鑰以授權存取與目的地類型相關聯的 AWS 服務。這個鍵/值對是選用的。如需詳細資訊，請參閱 目的地安全組態。

`SecretKey`

指定要使用的 AWS 秘密金鑰以授權存取與目的地類型相關聯的 AWS 服務。這個鍵/值對是選用的。如需詳細資訊，請參閱 目的地安全組態。

`Region`

指定哪個 AWS 區域包含要串流的目標資源。這個鍵/值對是選用的。

`ProfileName`

指定哪個 AWS 描述檔要用於身份驗證。這個指定鍵/值對是選用的，但若有指定，它會覆寫任何指定的存取金鑰和秘密金鑰。如需詳細資訊，請參閱 目的地安全組態。

`RoleARN`

指定要使用的 IAM 角色以存取與目的地類型相關聯的 AWS 服務。當 Windows 版 Kinesis Agent 在 EC2 執行個體上執行時，此選項非常實用，但不同的角色會比執行個體描述檔參考的角色更適合。例如，跨帳戶角色可用來鎖定與 EC2 執行個體不同之 AWS 帳戶中的資源目標。這個鍵/值對是選用的。

`Format`

指定要在串流之前套用至日誌和事件資料的序列化類型。有效值為 `json` 和 `xml`。當您必須進行資料管道的下游分析或需要特定形式的資料時，此選項很有幫助。這個指定鍵/值對是選用的，但若未指定，則會將來源的一般文字從目的地串流到與目的地類型相關聯的 AWS 服務。

`TextDecoration`

當未指定任何 `Format` 時，`TextDecoration` 會指定在串流日誌或事件記錄時應該包含哪些額外的文字。如需詳細資訊，請參閱 設定目的地宣告。這個鍵/值對是選用的。

`ObjectDecoration`

當指定 `Format` 時，`ObjectDecoration` 會指定在序列化和串流之前，日誌或事件記錄中應該包含哪些額外的資料。如需詳細資訊，請參閱 設定目的地宣告。這個鍵/值對是選用的。

`BufferInterval`

為了盡可能減少對目的地類型相關聯之 AWS 服務的 API 呼叫，Windows 專用 Kinesis Agent 會先緩衝多個日誌、事件或指標記錄之後再串流。這可以節省按 API 呼叫收費的服務成本。`BufferInterval` 可指定在串流到 AWS 服務之前應該緩衝記錄的最長時間 (秒為單位)。這個指定鍵/值對是選用的，但若有指定，請使用字串代表值。

`BufferSize`

為了盡可能減少對目的地類型相關聯之 AWS 服務的 API 呼叫，Windows 專用 Kinesis Agent 會先緩衝多個日誌、事件或指標記錄之後再串流。這可以節省按 API 呼叫收費的服務成本。`BufferSize` 可指定在串流到 AWS 服務之前要緩衝的記錄數量上限。這個指定鍵/值對是選用的，但若有指定，請使用字串代表值。

`MaxAttempts`

指定在串流持續失敗時要將日誌、事件和指標記錄串流到 AWS 服務的次數上限。這個鍵/值對是選用的。若有指定，請使用字串代表值。預設值為 3。

如需使用各種目的地的完整組態檔案範例，請參閱從 Windows 應用程式事件日誌串流到目的地。

主題

KinesisStream 目的地組態

所以此KinesisStream目的地類型會將日誌記錄和事件串流到 Kinesis Data Streams 服務。一般而言，資料串流到 Kinesis Data S流的資料會由使用各種 AWS 服務執行的一或多個自訂應用程式來處理。系統會將資料串流到使用 Kinesis Data Streams 設定的具名串流。如需詳細資訊，請參閱 。Amazon Kinesis Data Streams 開發者指南。

以下是範例 Kinesis Data Streams 目的地宣告：

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

所有 KinesisStream 目的地宣告都可以提供下列額外的鍵/值對：

SinkType

必須指定，而且值必須為常值字串 KinesisStream。

StreamName

指定 Kinesis 資料串流的名稱，以接收從KinesisStream水槽類型 (必要)。串流資料之前，請先在 AWS 管理主控台、AWS CLI 或透過使用 Kinesis Data Streams API 的應用程式來設定串流。

RecordsPerSecond

指定每秒串流到 Kinesis Data Streams 的記錄數目上限。這個鍵/值對是選用的。若有指定，請使用整數代表值。預設值為 1000 筆記錄。

BytesPerSecond

指定每秒串流到 Kinesis Data Streams 的位元組數目上限。這個鍵/值對是選用的。若有指定，請使用整數代表值。預設值為 1 MB。

此目的地類型的預設 BufferInterval 為 1 秒，而預設的 BufferSize 為 500 筆記錄。

KinesisFirehose 目的地組態

所以此KinesisFirehose目的地類型會將日誌記錄和事件串流到 Kinesis Data Firehose 服務。Kinesis Data Firehose 會將串流的資料交付給其他服務以儲存。接著，存放的資料通常會在資料管道後續階段分析。系統會將資料串流到使用 Kinesis Data Firehose 設定的具名交付串流。如需詳細資訊，請參閱 。Amazon Kinesis Data Firehose 開發者指南。

以下是範例 Kinesis Data Firehose 目的地宣告：

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

所有 KinesisFirehose 目的地宣告都可以提供下列額外的鍵/值對：

`SinkType`

必須指定，而且值必須為常值字串 `KinesisFirehose`。

`StreamName`

指定 Kinesis Data Firehose 交付串流的名稱，以接收從`KinesisStream`水槽類型 (必要)。串流資料之前，請先使用 AWS 管理主控台、AWS CLI 或透過使用 Kinesis Data Firehose API 的應用程式來設定交付串流。

`CombineRecords`

當設定為`true`，指定將多個小記錄合併成具有 5 KB 最大大小的大型記錄。這個鍵/值對是選用的。使用此函數組合的記錄由\n。如果您使用 AWS Lambda 轉換 Kinesis Data Firehose 記錄，則您的 Lambda 函數需要將分隔符號字元納入考量。

`RecordsPerSecond`

指定每秒串流到 Kinesis Data Streams 的記錄數目上限。這個鍵/值對是選用的。若有指定，請使用整數代表值。預設值為 5000 筆記錄。

`BytesPerSecond`

指定每秒串流到 Kinesis Data Streams 的位元組數目上限。這個鍵/值對是選用的。若有指定，請使用整數代表值。預設值為 5 MB。

此目的地類型的預設 `BufferInterval` 為 1 秒，而預設的 `BufferSize` 為 500 筆記錄。

CloudWatch Logs 目的地組態

所以此`CloudWatch`目的地類型會將指標串流到 CloudWatch Logs 服務。您可以在 AWS 管理主控台中檢視指標。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。

以下是 `CloudWatch` 目的地宣告的範例：

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

所有 `CloudWatch` 目的地宣告都可以提供下列額外的鍵/值對：

`SinkType`

必須指定，而且值必須為常值字串 `CloudWatch`。

`Interval`

指定 Windows 適用的 Kinesis Agent 向 CloudWatch 服務回報指標的頻率 (以秒為單位)。這個鍵/值對是選用的。若有指定，請使用整數代表值。預設值為 60 秒。如果您想要高解析度的 CloudWatch 指標，請指定 1 秒。

`Namespace`

指定要回報指標資料的 CloudWatch 命名空間。CloudWatch 命名空間會將指標集分組。這個鍵/值對是選用的。預設值為 `KinesisTap`。

`Dimensions`

指定 CloudWatch 維度，以用於隔離命名空間內的指標集。好比說，如果要提供每個桌面或伺服器的個別指標資料集，這會很有用。這個指定鍵/值對是選用的，但若有指定，值必須符合以下格式："key1=value1;key2=value2..."。預設值為 `"ComputerName={computername};InstanceId={instance_id}"`。此值支援目的地變數替換。如需詳細資訊，請參閱 設定目的地變數替換。

`MetricsFilter`

指定要將哪些指標從內建 Kinesis Agent 串流到 CloudWatch。如需內建 Kinesis Agent Windows 指標來源的詳細資訊，包括此鍵/值對的值語法詳細資訊，請參閱Windows 內建指標來源適用的 Kinesis 代理程式。

`CloudWatchLogs` 目的地組態

所以此`CloudWatchLogs`目的地類型會將日誌記錄和事件串流到 Amazon CloudWatch Logs。您可以在 AWS 管理主控台中檢視日誌，或透過資料管道的其他階段來處理這些日誌。系統會將資料串流到 CloudWatch 日誌中設定的具名日誌串流。系統會將日誌串流分為具名日誌群組。如需詳細資訊，請參閱 。Amazon CloudWatch Logs 使用者指南。

以下是範例 CloudWatch Logs 目的地宣告：

```
{
   "Id": "MyCloudWatchLogsSink",
   "SinkType": "CloudWatchLogs",
   "BufferInterval": "60",
   "BufferSize": "100",
```

```
    "Region": "us-west-2",
    "LogGroup": "MyTestLogGroup",
    "LogStream": "MyTestStream"
}
```

所有 CloudWatchLogs 目的地宣告都必須提供下列額外的鍵/值對：

SinkType

必須為常值字串 CloudWatchLogs。

LogGroup

指定 CloudWatch Logs 日誌群組的名稱，此群組包含由CloudWatchLogs接收器類型。如果指定的日誌群組不存在，Windows 版 Kinesis Agent 會嘗試建立它。

LogStream

指定 CloudWatch Logs 串流的名稱，以接收由CloudWatchLogs接收器類型。此值支援目的地變數替換。如需詳細資訊，請參閱 設定目的地變數替換。如果指定的日誌串流不存在，Windows 版 Kinesis 代理程式會嘗試建立它。

此目的地類型的預設 BufferInterval 為 1 秒，而預設的 BufferSize 為 500 筆記錄。緩衝大小上限為 10,000 筆記錄。

區域FileSystem目的地組態

接收器類型FileSystem會將日誌和事件記錄儲存到本機檔案系統上的檔案，而不是將它們串流到 AWS 服務。FileSystem水槽是有用的測試和診斷。例如，您可以使用此接收器類型來檢查記錄，然後再將記錄傳送至 AWS。

搭配FileSystem接收器，您也可以使用組態參數模擬批次處理、節流和錯誤重試，以模擬實際 AWS 接收器的行為。

來自所有來源的所有記錄連接至FileSystem接收器被保存到指定為FilePath。如果FilePath，則會將記錄儲存到名為的檔案*SinkId*.txt中的%TEMP%目錄，這通常是C:\Users*UserName*\AppData\Local\Temp，其中：*SinkId*是目的地的地的地的地的地的唯一識別符*UserName*是作用中使用者的 Windows 使用者名稱。

此接收器類型支援文字裝飾屬性。如需詳細資訊，請參閱 設定目的地宣告。

範例FileSystem目的地類型組態如下列範例所示。

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

所以此FileSystem組態由以下鍵/值對組成。

SinkType

必須為常值字串 FileSystem。

FilePath

指定儲存記錄的路徑和檔案。這個鍵/值對是選用的。如果未指定，則預設值為*TempPath**SinkId*.txt，其中：*TempPath*的資料夾是存放在%TEMP%變數和*SinkId*是目的地的地的地的地的地唯一識別符。

Format

指定事件的格式是json或xml。此金鑰值組是選用的，不區分大小寫。如果省略，事件會以純文字寫入檔案。

TextDecoration

僅適用於以純文字撰寫的事件。這個鍵/值對是選用的。

ObjectDecoration

僅適用於其中的事件：Format已設定為json。這個鍵/值對是選用的。

進階使用 — 記錄節流和失敗模擬

FileSystem可以通過模擬記錄節流模擬 AWS 接收器的行為。您可以使用下列索引鍵/值對來指定記錄節流和失敗模擬屬性。

通過獲取目標文件的鎖並防止寫入，您可以使用FileSystem匯入以模擬和檢查網路失敗時 AWS 匯入的行為。

下列範例會顯示FileSystem使用模擬屬性的規劃。

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

`RequestsPerSecond`

可選，並指定為字符串類型。若省略，則預設值為"5"。控制接收器處理的要求速率 (也就是寫入檔案)，而非記錄數目。適用於 Windows 的 Kinesis 代理程式會對 AWS 終端節點發出批次請求，因此請求可能包含多筆記錄。

`BufferSize`

可選，並指定為字串類型。指定接收器在儲存至檔案之前批次處理的事件記錄數目上限。

`MaxBatchSize`

可選，並指定為字符串類型。指定接收器在儲存至檔案之前批次處理的事件記錄資料量上限 (以位元組為單位)。

最大記錄速率限制是`BufferSize`，它決定每個請求的記錄數目上限，以及`RequestsPerSecond`。您可以使用下列公式計算每秒記錄速率限制。

`記錄=BufferSize*RequestsPerSecond`

在上述範例中給定組態值，最大記錄速率為每秒 1000 筆記錄。

目的地安全組態

設定身份驗證

若要讓 Windows 專用 Kinesis Agent 將日誌、事件和指標串流到 AWS 服務，存取必須經過身份驗證。為 Windows 版的 Kinesis 代理程式提供驗證有幾種方法。您的執行方式取決於的 Windows Kinesis Agent 執行方式以及特定組織的特定安全需求。

- 如果是在 Amazon EC2 主機上執行的 Kinesis Agent，提供身份驗證最安全與最簡單的方法是建立 IAM 角色並提供足夠權限以存取 AWS 服務所需的操作，並建立參考該角色的 EC2 執行個體描述

檔。如需建立執行個體描述檔的資訊，請參閱使用執行個體描述檔。如需要將哪些政策連接至 IAM 角色的資訊，請參閱設定授權。

在建立執行個體描述檔之後，您可以將其與任何使用 Kinesis Agent 的 EC2 執行個體建立關聯。如果執行個體已有相關聯的執行個體描述檔，您即可將適當政策連接至該執行個體描述檔相關聯的角色。

- 如果是 Windows 的 Kinesis Agent 在某個帳戶的 EC2 主機上執行，但目的地的目標資源位於不同的帳戶，您可以建立一個跨帳戶存取的 IAM 角色。如需詳細資訊，請參閱「」教學課程：使用 IAM 角色將存取權委派給不同 AWS 帳戶。在建立跨帳戶角色之後，請為跨帳戶角色指定 Amazon Resource Name (ARN) 做為`RoleARN`目的地宣告中的地金鑰值對。即會 Kinesis 存取與該目的地類型相關聯的 AWS 資源時，嘗試擔任指定的跨帳戶角色。
- 如果是在 Amazon EC2 之外執行的 Kinesis Agent，例如現場部署)，您可以使用幾個選項：
  - 如果可以將現場部署伺服器或桌面機器註冊為 Amazon EC2 Systems Manager 受管的執行個體，請使用下列程序來設定身份驗證：
    1. 使用在混合式環境中設定 AWS Systems Manager 所述的程序，以建立服務角色、建立受管執行個體的啟用，並安裝 SSM 代理程式。
    2. 將適當的政策連接至服務角色，以讓 Windows 版 Kinesis Agent 存取從所設目的地串流資料所需的資源。如需要將哪些政策連接至 IAM 角色的資訊，請參閱設定授權。
    3. 使用設定`ProfileRefreshingAWSCredentialProvider`重新整理 AWS 登入資料來重新整理 AWS 登入資料。

    這是針對非 EC2 執行個體的建議方法，因為 SSM 和 AWS 會安全地管理登入資料。
  - 如果可接受以特定使用者身分執行的 AWSKinesis Agent 服務，而非預設的系統帳戶，請使用下列程序：
    1. 在要使用 AWS 服務的 AWS 帳戶中，建立 IAM 使用者。在建立過程中，擷取此使用者的存取金鑰和秘密金鑰。您在此程序的後續步驟中將需要這些資訊。
    2. 將政策連接至 IAM 使用者，以授權存取必要服務的必要作業。如需要將哪些政策連接至 IAM 使用者的資訊，請參閱設定授權。
    3. 變更每個桌面或伺服器的 AWSKinesisTap 服務，使其以特定使用者身分執行而非預設的系統帳戶。
    4. 使用稍早記錄的存取金鑰和秘密金鑰，在軟體開發套件存放區中建立描述檔。如需詳細資訊，請參閱設定 AWS 登入資料。
    5. 更新 `%PROGRAMFILES%\Amazon\AWSKinesisTap` 目錄中的 `AWSKinesisTap.exe.config` 檔案，以指定您在先前步驟建立的描述檔名稱。如需詳細資訊，請參閱設定 AWS 登入資料。

針對不得為受管執行個體的非 EC2 主機，這是建議的方法，因為系統會為特定主機和特定使用者加密登入資料。

- 如果必須以預設系統帳戶執行適用於 Windows 的 Kinesis Agent 服務，您必須使用共享登入資料檔案。這是因為系統帳戶沒有 Windows 使用者描述檔，無法啟用軟體開發套件存放區。系統不會加密共享登入資料檔案，因此我們不建議此做法。如需如何使用共用組態檔的資訊，請參閱設定 AWS 登入資料中的適用於 .NET 的 AWS 開發套件。如果您使用此方法，建議您使用 NTFS 加密，並限制共享組態檔案的檔案存取。管理平台應該輪換金鑰，而共享組態檔案必須在金鑰輪換時更新。

雖然您可以直接在目的地宣告中提供存取金鑰和秘密金鑰，但不建議此方法，因為系統不會加密宣告。

設定授權

將適當的政策連接至 IAM 使用者或角色，以便 Windows 版 Kinesis Agent 用來串流資料到 AWS 服務：

Kinesis Data Streams

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

若要限制特定區域、帳戶或串流名稱的授權，請將 ARN 中的適當星號取代為特定值。如需詳細資訊，請參閱使用 IAM 控制對 Amazon Kinesis Data Streams 資源的存取中「Kinesis Data Streams 的 Amazon Resource Name (ARN)」。

Kinesis Data Firehose

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

若要限制特定區域、帳戶或交付串流名稱的授權，請將 ARN 中的適當星號取代為特定值。如需詳細資訊，請參閱「」控制 Amazon Kinesis Data Firehose 的存取中的Amazon Kinesis Data Firehose 開發者指南。

CloudWatch

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

如需詳細資訊，請參閱「」管理 CloudWatch 資源存取許可概觀中的Amazon CloudWatch Logs 使用者指南。

含現有日誌群組和日誌串流的 CloudWatch 日誌

```
{
```

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

若要限制存取特定區域、帳戶、日誌群組或日誌串流，請將 ARN 中適當星號取代為適當的值。如需詳細資訊，請參閱「」管理 CloudWatch 日誌資源存取許可概觀中的Amazon CloudWatch Logs 使用者指南。

CloudWatch Logs 與 Windows 專用 Kinesis 代理程式的額外許可，以建立日誌群組和日誌串流

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

若要限制存取特定區域、帳戶、日誌群組或日誌串流，請將 ARN 中適當星號取代為適當的值。如需詳細資訊，請參閱「」管理 CloudWatch 日誌資源存取許可概觀中的Amazon CloudWatch Logs 使用者指南。

EC2 標籤變數擴展所需的許可

使用變數擴展與 `ec2tag` 變數字首時，必須具備 `ec2:Describe*` 許可。

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

> **Note**
>
> 您可以將多個陳述式合併成單一政策，只要該政策中每個陳述式的 `Sid` 是唯一的即可。如需建立政策的詳細資訊，請參閱建立 IAM 政策中的IAM 使用者指南。

設定ProfileRefreshingAWSCredentialProvider重新整理 AWS 登入資料

如果您針對混合環境使用 AWS Systems Manager 來管理 AWS 登入資料，Systems Manager 會在c:\Windows\System32\config\systemprofile\.aws\credentials。如需混合式環境的 Systems Manager 的詳細資訊，請參閱為混合環境設定 AWS Systems Manager中的AWS Systems Manager 使用者指南。

由於 AWS .net 開發套件不會自動取得新的登入資料，因此我們會提供ProfileRefreshingAWSCredentialProvider外掛程式來重新整理認證。

您可以使用CredentialRef屬 AWS 以參考Credentials定義，其中CredentialType屬性會設定為ProfileRefreshingAWSCredentialProvider如下列範例所示，所示。

```
{
    "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
    }]
}
```

認證定義由下列屬性組成，做為索引鍵/值組。

`Id`

定義了接收器定義可以使用指定的字符串CredentialRef來參考此認證組態。

`CredentialType`

設定到常值字串ProfileRefreshingAWSCredentialProvider。

`Profile`

選用。預設值為 `default`。

`FilePath`

選用。指定 AWS 登入資料檔案的路徑。如果省略，`%USERPROFILE%/.aws/credentials` 為預設值。

`RefreshingInterval`

選用。重新整理認證的頻率 (以秒為單位)。如果省略，`300` 為預設值。

設定目的地宣告

您可選擇在目的地宣告中包含鍵/值對，以指定其他要串流到各種 AWS 服務的資料，藉此強化從來源收集的記錄。

`TextDecoration`

當目的地宣告中未指定任何 `Format` 時，請使用此鍵/值對。該值是用於變數替換發生時的特殊格式字串。例如，假設您針對目的地提供 `"{ComputerName}:::{timestamp:yyyy-MM-dd HH:mm:ss}:::{_record}"` 的 `TextDecoration`。當來源發出的日誌記錄包含文字 `The system has resumed from sleep.`，且該來源已透過管道連線到目的地，則會將文字 `MyComputer1:::2017-10-26 06:14:22:::The system has resumed from sleep.` 串流到與目的地類型相關聯的 AWS 服務。`{_record}` 變數會參考來源交付的原始文字記錄。

`ObjectDecoration`

當目的地宣告中指定 `Format` 時，請使用此鍵/值對新增其他資料，之後再進行記錄序列化。例如，假設您針對指定 JSON `Format` 的目的地提供 `"ComputerName={ComputerName};DT={timestamp:yyyy-MM-dd HH:mm:ss}"` 的 `ObjectDecoration`。系統會將產生的 JSON 串流到與目的地類型相關聯的 AWS 服務，包含下列鍵/值對與來源的原始資料：

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

如需使用 `ObjectDecoration` 的範例，請參閱 教學課程：使用適用於視窗的 Kinesis 代理程式將 JSON 日誌檔串流到 Amazon S3。

ObjectDecorationEx

指定一個表示式，它允許更靈活的資料萃取和格式相比，ObjectDecoration。當水槽的格式是可以使用此字段json。表示式語法如下所示。

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

例如，以下ObjectDecorationEx屬性：

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

轉換文字記錄：

System log message

進入 JSON 對象，如下所示，並使用表達式返回的值：

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

如需有關制定運算式的詳細資訊，請參閱撰寫運算式的秘訣。大部分的ObjectDecoration聲明應該使用新語法工作，但時間戳變量除外。A{timestamp:yyyyMMdd}欄位ObjectDecoration會將其表示為{format(_timestamp,'yyyyMMdd')}中的ObjectDecorationEx。

TextDecorationEx

指定一個表示式，它允許更靈活的資料萃取和格式相比，TextDecoration，如下列範例所示。

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

您可以使用TextDecorationEx來組成 JSON 對象。使用 '@ {'以逸出開放的大括號，如下列範例所示。

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

如果連線到目的地之來源的來源類型為 DirectorySource，則目的地可以使用下列三個額外變數：

_FilePath

日誌檔案的完整路徑。

_FileName

檔案的檔案名稱和副檔名。

_Position

整數，其表示記錄在日誌檔中的位置。

當您使用的來源會從連線到目的地 (其可將所有記錄串流到單一串流) 的多個檔案收集日誌記錄時，這些變數非常有用。將這些變數插入串流記錄時，可啟用資料管道的下游分析，以按檔案和記錄在每個檔案中的位置來排序記錄。

撰寫運算式的秘訣

運算式可為下列任何一項：

- 變數表示式。
- 常量表達式，例如'hello'、1、1.21、null、true、false。
- 呼叫函數的叫用表達式，如下列範例所示。

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

特殊字元

需要兩個反斜杠來轉義特殊字符。

Nesting

函數呼叫可以巢狀，如下列範例所示。

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

Variables

有三種變數：局部、中繼和全域。

- 局部變數開頭為$例如$message。它們用於解析事件對象的屬性，如果事件是字典的條目，或者如果事件是 JSON 對象的屬性。如果局部變量包含空格或特殊字符，請使用帶引號的局部變量，如$'date created'。
- 中繼變數以下劃線開頭（_），並用於解析事件的元數據。所有事件類型都支援下列中繼變數。

  _timestamp

  事件的時間戳記。

  _record

  事件的原始文字表示。

  記錄事件支援下列其他中繼變數。

  _filepath

  _filename

  _position

  _linenumber

- 全域變數解析環境變量，EC2 實例元數據或 EC2Tag。為獲得更高的效能，我們建議您使用字首來限制搜尋範圍，例如{env:ComputerName}、{ec2:InstanceId}，以及{ec2tag:Name}。

內建函數

Windows 專用的 Kinesis 代理程式支援以下內建功能。如果任何參數是NULL並且該函數不是為了處理NULL，一個NULL物件會傳回。

```
//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)
```

設定目的地變數替換

KinesisStream、KinesisFirehose 與 CloudWatchLogs 目的地宣告必須使用 LogStream 或 StreamName 鍵/值對。這些鍵/值可包含由 Windows 版 Kinesis Agent 自動解析的變數參考。適用於CloudWatchLogs，LogGroup鍵/值對也必須使用，並可包含由適用於 Windows 的 Kinesis Agent 自動解析的變數參考。您可以使用範本 {*prefix*:*variablename*} 來指定變數，其中 *prefix*: 是選用的。支援的字首如下：

- env— 變數參考會解析為相同名稱的環境變數值。
- ec2— 變數參考會解析為相同名稱的 EC2 實體中繼資料。
- ec2tag— 變數參考會解析為相同名稱的 EC2 執行個體標籤值。若要存取執行個體標籤，則必須具備 ec2:Describe* 許可。如需詳細資訊，請參閱 EC2 標籤變數擴展所需的許可。

如果未指定字首，且環境變數含有 variablename 的相同名稱，則變數參考會解析為環境變數的值。否則，如果 variablename 是 instance_id 或 hostname，則變數參考會解析為相同名稱的 EC2 中繼資料值。否則，就不會解析變數參考。

以下是使用變數參考的有效鍵/值對範例：

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

CloudWatchLogs 目的地器宣告支援特殊格式的時間戳記變數，以讓來源原始日誌或事件記錄的時間戳記，更改日誌串流的名稱。格式是 {timestamp:timeformat}。請參閱下列範例：

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

如果日誌或事件記錄於 2017 年 6 月 5 日產生，則先前範例中 LogStream 鍵/值對的值會解析為 "LogStream_20170605"。

如獲授權，則 CloudWatchLogs 目的地類型可根據產生的名稱視需要自動建立新的日誌串流。您無法針對其他目的地類型執行上述動作，因為它們需要串流名稱以外的其他組態。

文字和物件裝飾中也會發生特殊的變數替換。如需詳細資訊，請參閱 設定目的地宣告。

設定目的地佇列

如果記錄因為暫時性連線問題無法串流到與下列目的地類型相關聯的 AWS 服務，KinesisStream、KinesisFirehose 與 CloudWatchLogs 目的地宣告可以選擇啟用記錄佇列。若要啟用佇列並在連線恢復時自動重試串流，請在目的地宣告中使用下列鍵/值對：

QueueType

指定要使用的佇列機制。唯一支援的值是 file，這表示記錄應該排入檔案佇列。若要啟用 Windows 版 Kinesis 代理程式的佇列功能，這個索引鍵/值組是必要的。如果未指定，則預設行為是僅會排入記憶體佇列，並在達到記憶體佇列限制時無法串流。

`QueuePath`

指定資料夾的路徑，該資料夾包含佇列記錄的檔案。這個鍵/值對是選用的。預設值是 `%PROGRAMDATA%\KinesisTap\Queue\SinkId`，其中 SinkId 是識別符 (您已指派為目的地宣告的 `Id` 值)。

`QueueMaxBatches`

將記錄排入串流佇列時，限制 Windows 專用 Kinesis Agent 可以使用的空間總數限制。空間量限制為此鍵/值對的值乘以每批次的最大位元組數。`KinesisStream`、`KinesisFirehose` 與 `CloudWatchLogs` 目的地類型的每批次最大位元組分別為 5 MB、4 MB 和 1 MB。當達到此限制時，任何串流失敗都無法排入佇列，且會回報為無法復原的失敗。這個鍵/值對是選用的。預設值為 10,000 個批次。

設定目的地的代理

若要針對所有存取 AWS 服務的 Kinesis 代理程式設定代理，請編輯位於`%Program Files%\Amazon\KinesisTap\AWSKinesisTap.exe.config`。如需說明，請參閱`proxy`中的章節適用於 .NET 的 AWS 開發套件組態檔案參考中的適用於 .NET 開發人員的 AWS 開發套件。

在更多接收屬性中配置解析變數

下列範例顯示使用目的地組態`Region`環境變數的值`Region`屬性金鑰值對。適用於`RoleARN`，它指定 EC2 標籤鍵`MyRoleARN`，它會計算與該鍵關聯的值。

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

在 AWS 接收器中使用 RoleARN 屬性時設定 AWS STS 區域端點

此功能只有在您在 Amazon EC2 上使用 KinesStap 並使用`RoleARN`屬性，以擔任外部 IAM 角色，以使用目的地 AWS 服務進行身份驗證。

透過設定`UseSTSRegionalEndpoints`至`true`，您可以指定代理程式使用區域端點 (例如`https://sts.us-east-1.amazonaws.com`) 而不是全域端點 (例如`https://sts.amazonaws.com`。使用區域 STS 端點可減少作業的往返延遲，並限制全域端點服務失敗的影響。

為 AWS 接收器設定 VPC 端點

您可以在接收器組態中指定 VPC 端點`CloudWatchLogs`、`CloudWatch`、`KinesisStreams`，以及`KinesisFirehose`接收器類型。VPC 端點可讓您不需要網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線，即可將 VPC 私密連線至支援的 AWS 服務以及採用 AWS PrivateLink 技術的 VPC 端點服務。VPC 中的執行個體不需要公有 IP 地址，即可與服務中的資源通訊。VPC 與另一個服務之間的流量都會保持在 Amazon 網路的範圍內。如需詳細資訊，請參閱「」VPC 端點中的Amazon VPC 使用者指南。

您可 VPC 使用`ServiceURL`屬性如下列範例所示，`CloudWatchLogs`接收器組態。將的值設定為`ServiceURL`設定為顯示在VPC 端點詳細資訊索 VPC 標籤。

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

設定代理伺服器的替代方式

此功能可讓您使用 AWS 開發套件 (而非 .NET) 內建的代理支援，在接收器組態中設定代理伺服器。以前，將代理程式設定為使用 Proxy 的唯一方法是使用 .NET 的原生功能，該功能會自動透過 Proxy 檔案中定義的 Proxy 路由所有 HTTP/S 要求。

如果您目前正在將代理程式與 Proxy 伺服器搭配使用，則不需要變更即可使用此方法。

您可以使用`ProxyHost`和`ProxyPort`內容以設定替代 Proxy，如下列範例所示。

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

管道宣告

使用管道宣告來連接來源（請參閱來源宣告）到一個接收器（請參閱目的地宣告），以適用於 Microsoft Windows 的 Amazon Kinesis Kinesis 代理。管道宣告是以 JSON 物件來表示。Windows 版 Kinesis 代理程式啟動之後，系統即會從指定的管道來源收集日誌、事件或指標。然後，它們會被串流到使用該管道相關聯目的地的各種 AWS 服務。

下列為範例 管道宣告：

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

主題

- 設定管道
- 設定 Windows 度量管道的 Kinesis 代理程式

設定管道

所有管道宣告可以包含下列鍵/值對：

`Id`

指定管道的名稱 (必要)。此項目在組態檔案中必須是唯一的。

`Type`

指定當日誌資料從來源傳輸到目的地時管道要套用的轉換類型 (如果有)。唯一支援的值為 `RegexFilterPipe`。這個值可讓規則表達式篩選日誌記錄的基礎文字表示。使用篩選功能時只會將相關的日誌記錄向下游傳送到資料管道，而可以降低傳輸和儲存成本。這個鍵/值對是選用的。預設值是不提供轉換。

`FilterPattern`

指定 `RegexFilterPipe` 管道的規則表達式，以用於篩選來源收集的日誌記錄，之後再傳輸至目的地。當規則表達式符合記錄的基礎文字呈現時，即會透過 `RegexFilterPipe` 類型管道傳輸日誌記錄。您仍可以使用 `RegexFilterPipe` 機制，來篩選在 `DirectorySource` 宣告中使用 `ExtractionPattern` 鍵/值對而產生的結構化日誌記錄。這是因為此機制會依據剖析之前的原始

文字表示來運作。此鍵/值對是選用的，但如果管道指定 RegexFilterPipe 類型，您就必須提供此鍵/值對。

下列為範例 RegexFilterPipe 管道宣告：

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

`SourceRef`

指定來源宣告的名稱 (Id 鍵/值對的值)，以定義為管道收集日誌、事件和指標資料的來源 (必要)。

`SinkRef`

指定目的地宣告的名稱 (Id 鍵/值對的值)，以定義為管道接收日誌、事件和指標資料的目的地 (必要)。

`IgnoreCase`

選用。接受值`true`或`false`。當設定為`true`，則正則表達式將以不區分大小寫的方式匹配記錄。

`Negate`

選用。接受值`true`或`false`。當設定為`true`，管道將轉發不符合規則表達式。

如需使用 RegexFilterPipe 管道類型的完整組態檔案範例，請參閱 使用管道。

設定 Windows 度量管道的 Kinesis 代理程式

您可以使用名為的內建指標來源`_KinesisTapMetricsSource`，可產生關於 Windows 適用的 Kinesis 代理程式的度量。如果有`CloudWatch`接收器聲明與`Id`的`MyCloudWatchSink`，以下範例管道宣告會將 Windows 產生的指標傳輸到該目的地：

```
{
   "Id": "KinesisAgentMetricsToCloudWatch",
   "SourceRef": "_KinesisTapMetricsSource",
```

```
    "SinkRef": "MyCloudWatchSink"
}
```

如需 Windows 專用的 Kinesis 代理程式內建指標來源的詳細資訊，請參閱Windows 內建指標來源適用的 Kinesis 代理程式。

如果組態檔案也會串流 Windows 效能計數器指標，則不建議您對 Windows 指標和 Windows 效能計數器指標使用相同的目的地，而 Kinesis 使用個別的管道和目的地。

設定自動更新

使用appsettings.json組態檔案，以啟用適用於微軟視窗的 Amazon Kinesis 代理程式自動更新，以及適用於視窗的 Kinesis 代理程式的組態檔案。若要控制更新行為，請在組態檔案 Sources、Sinks 以及 Pipes 相同層級中指定 Plugins 鍵/值對。

Plugins 鍵/值對可指定其他要使用的一般功能；這些功能不特別歸在來源、目的地及管道類別中。例如，有個外掛程式可用來更新 Windows 版 Kinesis 代理程式，而且有個外掛程式可用來更新appsettings.json組態檔案。外掛程式會以 JSON 物件表示，且一律具備 Type 鍵/值對。Type 決定可針對外掛程式指定其他哪些鍵/值對。目前支援下列外掛程式類型：

PackageUpdate

指定 Windows 版 Kinesis 組態檔案，應該定期檢查套件版本組態檔案。如果套件版本檔案指出應該安裝不同版本的 Windows 版本的 Kinesis 代理程式，則 Windows 版本的 Kinesis 代理程式會下載該版本並加以安裝。PackageUpdate 外掛程式鍵/值對包含：

Type

此值必須是字串 PackageUpdate，且是必要的。

Interval

指定檢查套件版本檔案的頻率以尋找任何變更 (以分鐘為單位、字串表示)。這個鍵/值對是選用的。如果未指定，預設值為 60 分鐘。如果值小於 1，就不會進行任何更新檢查。

PackageVersion

指定套件版本 JSON 檔案的位置。檔案可位於檔案共享 (file://)、網站 (http://) 或 Amazon S3 (s3://。例如，值為s3://mycompany/config/agent-package-version.json表示適用於 Windows 的 Kinesis 代理程式應該檢查config/agent-package-version.json檔案mycompanyAmazon S3 儲存貯體。您應該根據該檔案的內容來執行更新。

> **Note**
>
> 的值`PackageVersion`中鍵/值對對對 Amazon S3 會區分大小寫。

下列為套件版本檔案的內容範例：

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

所以此`Version`中鍵/值對會指定應該安裝哪個版本的 Windows 版本的 Kinesis 代理程式 (如果尚未安裝)。`PackageUrl` 中的 `{Version}` 變數參考可解析您為 `Version` 鍵/值對指定的值。在此範例中，會將變數解析為字串 `1.0.0.106`。提供此變數解析時，您即可在版本套件版本檔案的單一位置中存放所需的特定版本。您可以使用多個套件版本檔案，來控制採用新版本的 Windows 版 Kinesis Agent 進度，在大型部署之前先驗證新版本。若要復原 Windows 專用 Kinesis 代理程式的部署，請將一或多個套件版本檔案變更為指定您環境中已知可用的 Windows 舊版 Kinesis 代理程式。

變數替換會影響 `PackageVersion` 鍵/值對的值，以推動不同套件版本檔案的自動選取。如需變數替換的詳細資訊，請參閱設定目的地變數替換。

`AccessKey`

指定要使用哪個存取金鑰來驗證 Amazon S3 中的套件版本檔案存取權。這個鍵/值對是選用的。我們不建議您使用此鍵/值對。如需其他建議的身份驗證方法，請參閱設定身份驗證。

`SecretKey`

指定要使用哪個秘密金鑰來驗證 Amazon S3 中的套件版本檔案存取權。這個鍵/值對是選用的。我們不建議您使用此鍵/值對。如需其他建議的身份驗證方法，請參閱設定身份驗證。

`Region`

指定要使用哪個區域端點從 Amazon S3 存取套件版本檔案。這個鍵/值對是選用的。

`ProfileName`

指定要使用哪個安全性描述檔來驗證 Amazon S3 中的套件版本檔案存取權。如需詳細資訊，請參閱 設定身份驗證。這個鍵/值對是選用的。

`RoleARN`

指定要擔任哪個角色來驗證跨帳戶案例中 Amazon S3 套件版本檔案存取權。如需詳細資訊，請參閱 設定身份驗證。這個鍵/值對是選用的。

如果未指定任何 `PackageUpdate` 外掛程式，則不會檢查任何套件版本檔案來判斷是否需要更新。

`ConfigUpdate`

指定 Windows 版 Kinesis 代理程式應定期檢查是否有更新的`appsettings.json`組態檔案存放在檔案共享、網站或 Amazon S3 中。如果存在更新的組態檔案，即會下載並安裝 Kinesis 組態檔案。`ConfigUpdate`鍵/值對包含下列項目：

`Type`

此值必須是字串 `ConfigUpdate`，且是必要的。

`Interval`

指定檢查新組態檔案的頻率 (以分鐘為單位、字串表示)。這個鍵/值對是選用的；如果未指定，預設值為 5 分鐘。如果值小於 1，則不會檢查組態檔案更新。

`Source`

指定要在何處尋找更新的組態檔案。檔案可位於檔案共享 (`file://`)、網站 (`http://`) 或 Amazon S3 (`s3://`。例如，值為`s3://mycompany/config/appsettings.json`表示 Windows 版 Kinesis 代理程式應該檢查`config/appsettings.json`檔案`mycompany`Amazon S3 儲存貯體。

> **Note**
>
> 的值`Source`Amazon S3 中鍵/值對會區分大小寫。

變數替換會影響 `Source` 鍵/值對的值，以推動不同組態檔案的自動選取。如需變數替換的詳細資訊，請參閱設定目的地變數替換。

`Destination`

指定組態檔案要存放在本機電腦的位置。這可以是相對路徑、絕對路徑，或是包含環境變數參考的路徑，例如 `%PROGRAMDATA%`。如果是相對路徑，即會相對於 Windows 版 Kinesis 代理程式安裝的位置。一般而言，值應該是 `.\appsettings.json`。這個鍵/值對是必要的。

`AccessKey`

指定要使用哪個存取金鑰來驗證 Amazon S3 中的組態檔案存取權。這個鍵/值對是選用的。我們不建議您使用此鍵/值對。如需其他建議的身份驗證方法，請參閱設定身份驗證。

`SecretKey`

指定要使用哪個秘密金鑰來驗證 Amazon S3 中的組態檔案存取權。這個鍵/值對是選用的。我們不建議您使用此鍵/值對。如需其他建議的身份驗證方法，請參閱設定身份驗證。

`Region`

指定要使用哪個區域端點從 Amazon S3 存取組態檔案。這個鍵/值對是選用的。

`ProfileName`

指定要使用哪個安全描述檔來驗證 Amazon S3 中的組態檔案存取權。如需詳細資訊，請參閱 設定身份驗證。這個鍵/值對是選用的。

`RoleARN`

指定要擔任哪個角色來驗證跨帳戶案例中的 Amazon S3 中的組態檔案存取權。如需詳細資訊，請參閱 設定身份驗證。這個鍵/值對是選用的。

如果未指定任何 `ConfigUpdate` 外掛程式，則不會檢查任何組態檔案來判斷是否需要組態檔案更新。

下列為示範使用 `PackageUpdate` 和 `ConfigUpdate` 外掛程式的範例 `appsettings.json` 組態檔案。在此範例中，有個套件版本檔案位於`mycompany`名為的 Amazon S3 儲存貯體`config/agent-package-version.json`。系統大約每隔 2 小時會檢查此檔案是否有任何變更。如果套件版本檔案中指定不同版本的 Windows Kinesis Agent，即會從套件版本檔案指定的位置來安裝指定的代理程式版本。

除此之外，還有一個`appsettings.json`組態檔案儲存在`mycompany`名為的 Amazon S3 儲存貯體`config/appsettings.json`。系統大約每隔 30 分鐘會比較該檔案與目前的組態檔案。如果不同，即會從 Amazon S3 下載更新的組態檔案，並將其安裝到`appsettings.json`組態檔案。

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

適用於視窗設定範例的 Kinesis 代理程式

所以此appsettings.json組態檔案是一種 JSON 文件 Amazon Kinesis 可控制如何收集日誌、事件和指標。它也可以控制 Windows 專用 Kinesis Agent 如何轉換這些資料並將其串流到各種 AWS 服務。如需組態檔案中來源、目的地和管道宣告的詳細資訊，請參閱來源宣告、目的地宣告以及管道宣告。

下列各節包含各種不同類型案例的組態檔案範例。

主題

從各種來源串流到 Kinesis Data Streams

以下為範例示範：`appsettings.json`組態檔案示範如何將日誌和事件從各種來源串流到 Kinesis Data Streams，以及從 Windows 效能計數器串流到 Amazon CloudWatch 指標。

`DirectorySource`、`SysLog` 記錄剖析器

以下檔案會將 syslog 格式日誌記錄從所有檔案串流到，其中包含`.log`副檔名為`C:\LogSource\`目錄中的`SyslogKinesisDataStream`Kinesis Data Streams 會將 us-east-1 區域中的 Kinesis Data stream 串流。您可以建立書籤，以確保即使代理程式關閉並於稍後重新啟動時，仍會傳送日誌檔的所有資料。自訂應用程式可以讀取及處理來自 `SyslogKinesisDataStream` 串流的記錄。

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

```
      "Id": "SyslogDS2KSSink",
      "SourceRef": "SyslogDirectorySource",
      "SinkRef": "KinesisStreamSink"
    }
  ]
}
```

DirectorySource、SingleLineJson 記錄剖析器

以下檔案會將 JSON 格式日誌記錄從具有.log副檔名為C:\LogSource\目錄中的JsonKinesisDataStreamKinesis Data Streams 會將 us-east-1 區域中的 Kinesis Data stream 串流。串流之前，系統會將 ComputerName 和 DT 索引鍵的鍵/值對新增到每個 JSON 物件，包括電腦名稱和處理記錄的日期與時間值。自訂應用程式可以讀取及處理來自 JsonKinesisDataStream 串流的記錄。

```
{
  "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

以下檔案會將 Microsoft Exchange 產生的日誌記錄和存放在具有.log擴充功能C:\temp\ExchangeLog\目錄中的ExchangeKinesisDataStream在 us-east-1 區域中的 Kinesis Data stream，以 JSON 格式串流。雖然 Exchange 日誌不是 JSON 格式，但 Windows 適用 Kinesis 代理程式可以剖析這些日誌並轉換為 JSON。串流之前，系統會將 ComputerName 和 DT 索引鍵的鍵/值對新增到每個 JSON 物件，其中包含電腦名稱和處理記錄的日期與時間值。自訂應用程式可以讀取及處理來自 ExchangeKinesisDataStream 串流的記錄。

```
{
  "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

以下檔案會將 Internet Information Services (IIS) (存放在這些檔案的標準位置中) 串流到IISKinesisDataStreamKinesis Data Streams 會將 us-east-1 區域中的 Kinesis Data stream 串流。自訂應用程式可以讀取及處理來自 IISKinesisDataStream 串流的記錄。IIS 是一種 Windows web 伺服器。

```
{
  "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 與查詢

下列檔案會從 Windows 系統事件記錄檔資料流記錄事件，其層級為Critical或Error(小於或等於 2)串流到SystemKinesisDataStream在 us-east-1 區域中的 Kinesis Data stream，以 JSON 格式串流。自訂應用程式可以讀取及處理來自 SystemKinesisDataStream 串流的記錄。

```
{
```

```
  "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

以下檔案會將 Microsoft 通用語言執行平台 (CLR) 的例外狀況和安全事件串流到`ClrKinesisDataStream`在 us-east-1 區域中的 Kinesis Data stream，以 JSON 格式串流。自訂應用程式可以讀取及處理來自 `ClrKinesisDataStream` 串流的記錄。

```
{
  "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

以下檔案會 CloudWatch 開啟檔案總數、重新啟動後嘗試登入總數、磁碟每秒讀取數，以及可用磁碟空間百分比的效能計數器串流至 us-east-1 區域中的區域。您可以在 CloudWatch 中繪製這些指標的圖表、從圖表建置儀表板，以及設定警示以在超過閾值時傳送通知。

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

從 Windows 應用程式事件日誌串流到目的地

以下為範例示範：appsettings.json組態檔案示範將 Windows 應用程式事件日誌串流到適用於微軟視窗的 Amazon Kinesis 代理程式中的各種目的地。如需使用 KinesisStream 和 CloudWatch 目的地類型的範例，請參閱從各種來源串流到 Kinesis Data Streams。

KinesisFirehose

下列檔案串流Critical或ErrorWindows 應用程式記錄檔事件到WindowsLogFirehoseDeliveryStreamKinesis Data Firehose 交付串流位於 us-east-1 區域中。如果與 Kinesis Data Firehose 的連線中斷，系統會先將事件排入記憶體佇列。若有必要，系統會接著將它們排入磁碟檔案上的佇列，直到恢復連線。然後，事件即可解除佇列狀態，並後接任何新事件一起傳送。

您可以根據資料管道要求，設定 Kinesis Data Firehose，將串流資料存放到多種不同類型的儲存體與分析服務。

```
{
```

```
  "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

下列檔案串流Critical或ErrorWindows 應用程式 CloudWatch Logs 事件串流到MyServiceApplicationLog-Group日誌群組。每個串流名稱開頭為 Stream-。結尾為串流建立時的四位數年份、二位數月份和二位數日期，全部串連在一起 (例如，Stream-20180501 是 2018 年 5 月 1 日建立的串流)。

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

使用管道

以下範例 appsettings.json 組態檔案示範如何使用管道相關的功能。

此範例會將日誌項目從c:\LogSource\到ApplicationLogFirehoseDeliveryStreamKinesis Data Firehose 交付串流。它只包含符合 FilterPattern 鍵/值對所指定規則表達式的字行。具體來說，日誌檔中只有以10或11會將其串流到 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),.*"
    }
  ]
}
```

使用多個來源和管道

以下範例 appsettings.json 組態檔案示範如何使用多個來源和管道。

此範例會將應用程式、安全性和系統 Windows 事件日誌串流到EventLogStreamKinesis Data Firehose 交付串流使用三個來源、三個管道和單一目的地。

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

設定遙測

為了啟用更佳的支援，根據預設，Microsoft Windows 用 Amazon Kinesis 代理程式會收集代理程式操作的相關統計資料，並傳送至 AWS。此資訊並未包含任何個人識別資訊。它不會包含任何您收集或串流至 AWS 服務的資料。我們會每 60 分鐘收集約 1—2 KB 的這項指標資料。

您可以退出收集及傳輸這些統計資料。若要執行此作業，請將以下鍵/值對新增至與來源、接收和管道相同層級的 `appsettings.json` 組態檔案：

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

例如，以下組態檔案會設定來源、接收和管道，同時停用遙測：

```
{
  "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"
    }
}
```

我們會在啟用遙測時收集下列指標：

`ClientId`

安裝軟體時自動指派的唯一 ID。

`ClientTimestamp`

收集遙測的日期和時間。

`OSDescription`

作業系統的描述。

`DotnetFramework`

目前的 dotnet 框架版本。

`MemoryUsage`

Kinesis 代理程式使用的記憶體量 (MB)。

`CPUUsage`

Windows CPU 用量百分比 (小數)。例如，0.01 表示 1%。

`InstanceId`

如果 Windows 適用的 Kinesis 代理程式正在 Amazon EC2 執行個體上執行，則為 Amazon EC2 執行個體識別碼。

`InstanceType (string)`

如果 Windows 適用的 Kinesis 代理程式正在 Amazon EC2 執行個體上執行，則為 Amazon EC2 執行個體類型。

此外，我們還會收集適用於視窗度量的 Kinesis 代理程式清單中所列的指標。

教學課程：使用適用於視窗的 Kinesis 代理程式將 JSON 日誌檔串流到 Amazon S3

本教學示範使用 Amazon Kinesis vice (Windows) 設定資料管道的詳細步驟。

本教學包含下列步驟：

- 針對 Windows 使用 Kinesis Agent 將 JSON 格式的日誌檔案串流至Amazon Simple Storage Service (Amazon S3)透過Amazon Kinesis Data Firehose。如需有關 Windows 專用 Kinesis 代理程式的詳細資訊，請參閱什麼是 Amazon Kinesis Agent，適用於微軟視窗？。
- 增強日誌資料，再使用物件裝飾進行串流。如需詳細資訊，請參閱 設定目的地宣告。
- 使用Amazon Athena搜尋特定類型的日誌記錄。

Prerequisites

若您尚未擁有 AWS 帳戶，請遵循設定 AWS 帳戶來獲得一個。

主題

- 步驟 1：設定 AWS 服務
- 步驟 2：安裝、設定及執行適用於視窗的 Kinesis 代理程式
- 步驟 3：在 Amazon S3 中查詢日誌資料
- 後續步驟

步驟 1：設定 AWS 服務

遵循這些步驟，準備您的環境，使用 Amazon Simple Storage Service (Amazon S3)。如需詳細資訊和事前準備，請參閱教學課程：將 JSON 日誌檔案串流至 Amazon S3。

使用 AWS 管理主控台配置 AWS Identity and Access Management (IAM)、Amazon S3、Kinesis Data Firehose 管和 Amazon Elastic Compute Cloud (Amazon EC2)，以準備從 EC2 執行個體到 Amazon S3 的串流日誌資料。

主題

- 設定 IAM 政策及角色

- 建立 Amazon S3 儲存貯體
- 建立 Kinesis Data Firehose 交付串流
- 建立 Amazon EC2 執行個體以執行適用於視窗的 Kinesis 代理程式
- 後續步驟

設定 IAM 政策及角色

建立以下政策，授權 Windows 版 Kinesis Agent 將記錄串流至特定 Kinesis Data Firehose 交付串流：

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

Replace*region*，並包含將建立 Kinesis Data Firehose 交付串流的 AWS 區域名稱 (`us-east-1`，例如）。將 *account-id* 替換成將建立交付串流 AWS 帳戶的 12 位數帳戶 ID。

在導覽列中，選擇支援，然後Support 中心。您目前登入的 12 位數帳戶號碼 (ID) 會顯示在Support 中心導覽窗格。

使用以下程序建立政策。將政策命名為 `log-delivery-stream-access-policy`。

使用 JSON 政策編輯器建立政策

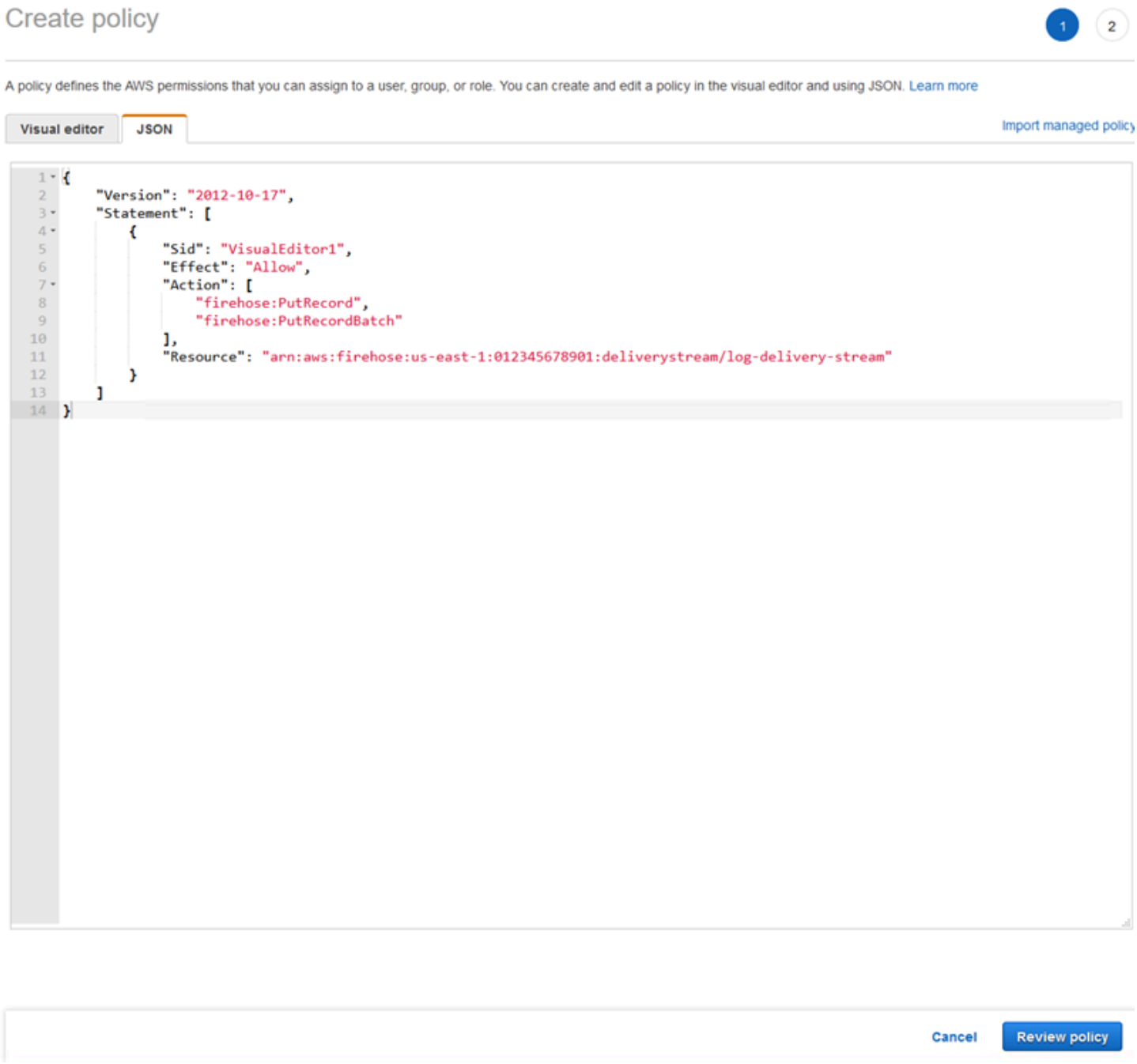
1. 登入 AWS 管理主控台，然後前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在左側的導覽窗格中，選擇 Policies (政策)。

如果這是您第一次選擇 Policies (政策)，將會顯示 Welcome to Managed Policies (歡迎使用受管政策) 頁面。選擇 Get Started (開始使用)。

3. 在頁面頂端，選擇 Create policy (建立政策)。
4. 請選擇 JSON 索引標籤。
5. 輸入 JSON 政策文件。如需 IAM 政策語言的詳細資料，請參閱IAM JSON 政策參考中的IAM 使用者指南。
6. 完成時，選擇 Review policy (檢閱政策)。Policy Validator (政策檢查工具) 會回報任何語法錯誤。

> **Note**
>
> 您可以隨時切換 Visual editor (視覺化編輯器) 與 JSON 標籤。但是，如果您進行變更或選擇檢閱政策中的Visual editor (視覺化編輯器)索引標籤，IAM 可能會調整您的政策結構以針對視覺化編輯器進行最佳化。如需詳細資訊，請參閱「」調整政策結構中的IAM 使用者指南。

7. 在 Review policy (檢閱政策) 頁面上，為您正在建立的政策輸入 Name (名稱) 與 Description (描述) (選用)。檢閱政策 Summary (摘要) 來查看您的政策所授予的許可。然後選擇 Create policy (建立政策) 來儲存您的工作。



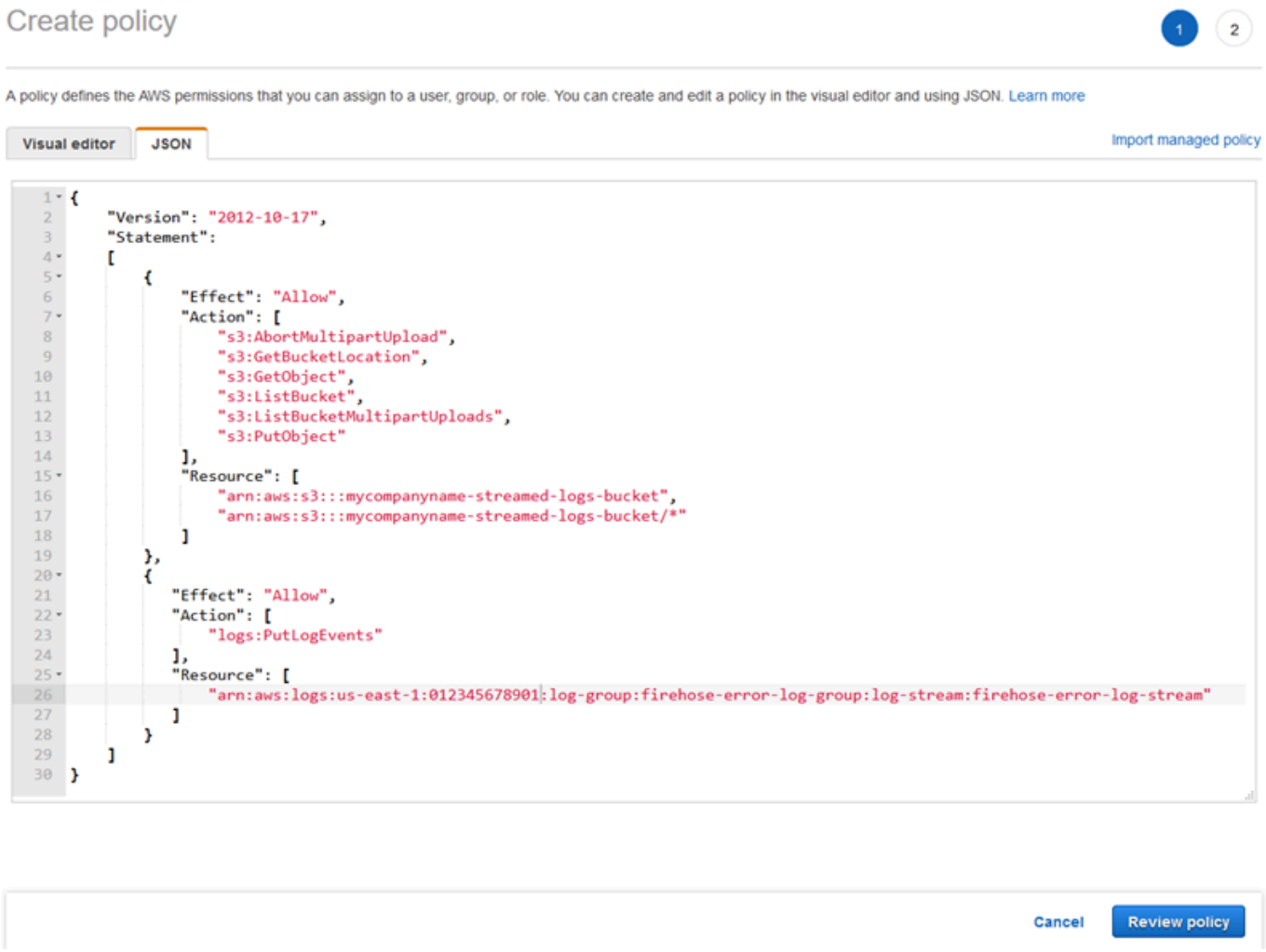
建立 Kinesis Data Firehose 存取 S3 儲存貯體的角色

1. 使用先前的程序，建立名為 `firehose-s3-access-policy` 的政策，使用以下 JSON 定義：

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

將 *bucket-name* 替換成存放日誌的唯一儲存貯體名稱。Replace*region*，並使用將建立 CloudWatch Logs 群組和日誌串流的 AWS 區域。這些是用來記錄透過 Kinesis Data Firehose 將資料串流至 Amazon S3 期間所發生的任何錯誤。將 *account-id* 替換成將建立日誌群組及日誌串流帳戶的 12 位數帳戶 ID。



2. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 選擇AWS 服務角色類型，然後選擇Kinesis服務。
4. 選擇Kinesis Data Firehose針對使用案例，然後選擇下一頁: 許可。
5. 在搜尋方塊中，輸入**`firehose-s3-access-policy`**、選擇政策，然後選擇下一頁: 檢閱。
6. 在 Role name (角色名稱) 方塊中，輸入 **`firehose-s3-access-role`**。
7. 選擇 Create Role (建立角色)。

建立角色，以與將執行 Windows S3 執行個體的執行個體描述檔建立關聯

1. 在 IAM 主控台的導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
2. 選擇AWS 服務角色類型，然後選擇EC2。
3. 選擇下一頁: 許可。
4. 在搜尋方塊中，輸入 **`log-delivery-stream-access-policy`**。

5. 選擇政策，然後選擇下一頁: 檢閱。
6. 在 Role name (角色名稱) 方塊中，輸入 **kinesis-agent-instance-role**。
7. 選擇 Create Role (建立角色)。

建立 Amazon S3 儲存貯體

建立 Kinesis Data Firehose 串流日誌的 S3 儲存貯體。

建立用於存放日誌的 S3 儲存貯體

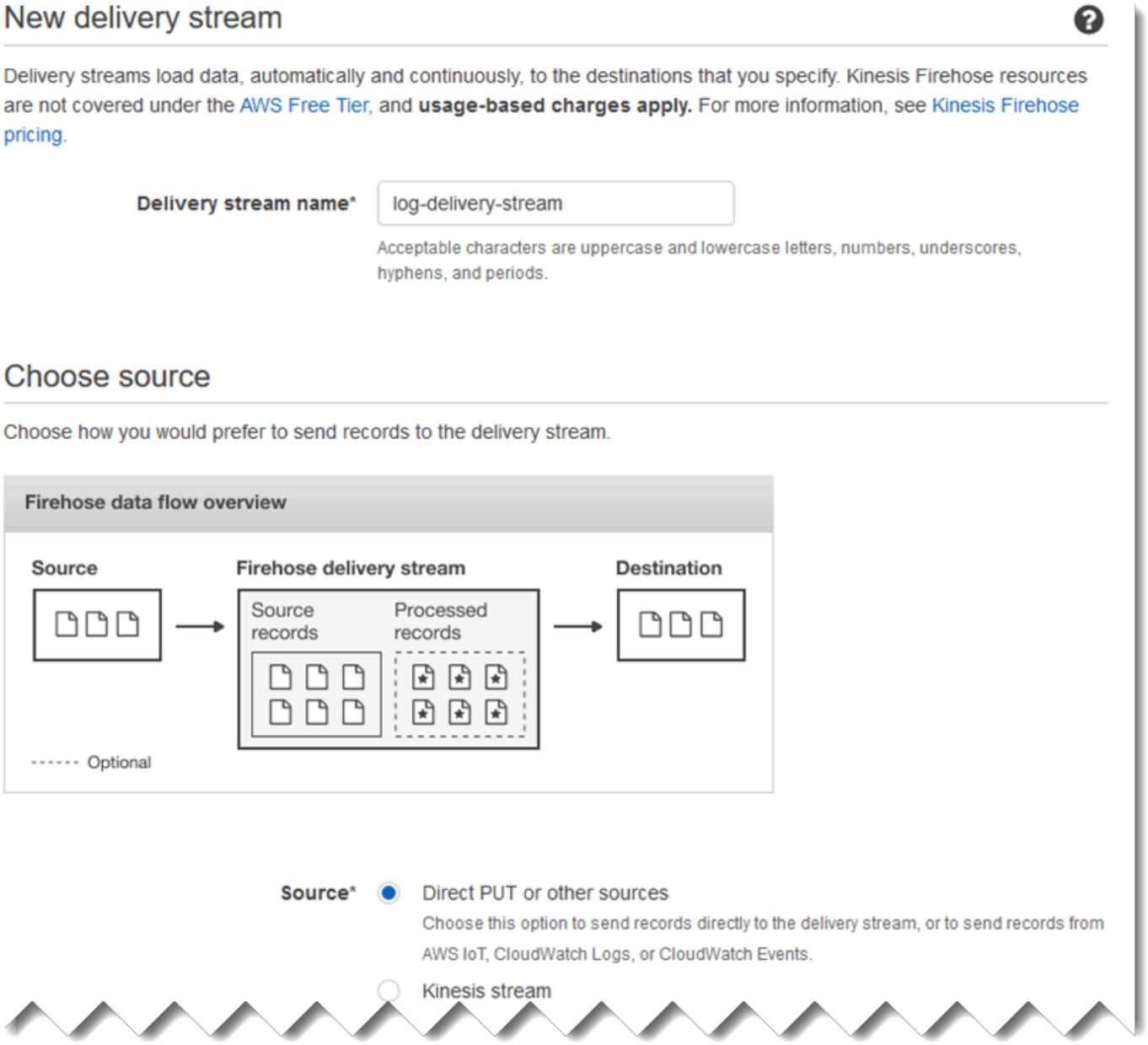
1. 前往 https://console.aws.amazon.com/s3/ 開啟 Amazon S3 主控台。
2. 選擇 Create bucket (建立儲存貯體)。
3. 在 Bucket name (儲存貯體名稱) 方塊中，輸入您在設定 IAM 政策及角色中所選擇的唯一 S3 儲存貯體名稱。
4. 選擇應建立儲存貯體的區域。這通常是和您意圖建立 Kinesis Data Firehose 交付串流和 Amazon S3 執行個體區域相同的區域。
5. 選擇 Create (建立)。

建立 Kinesis Data Firehose 交付串流

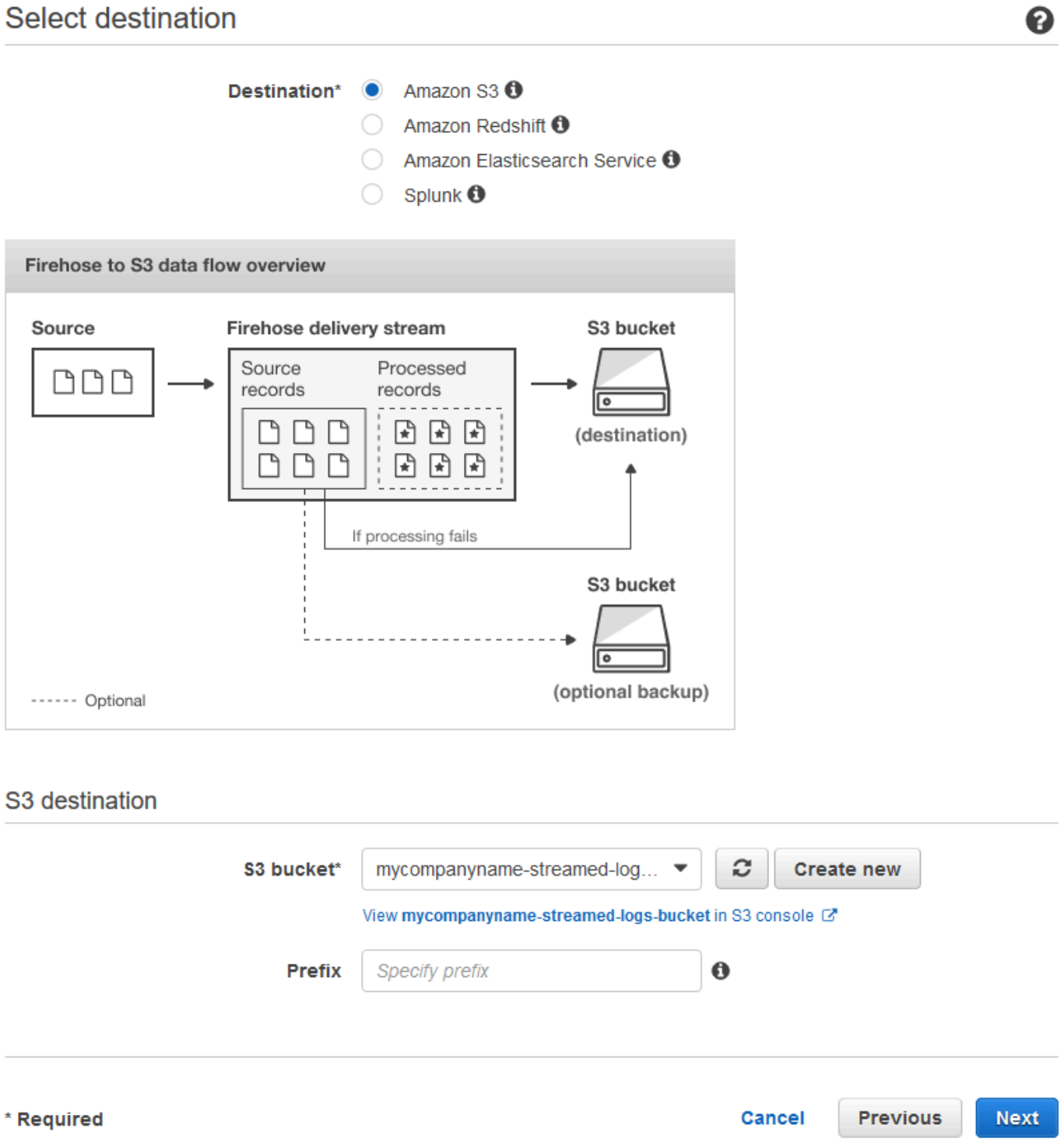
建立 Kinesis Data Firehose 交付串流，將串流記錄存放在 Amazon S3 的 Kinesis Data Firehose 交付串流。

建立 Kinesis Data Firehose 交付串流

1. 開啟 Kinesis Data Firehose 主控台，網址為https://console.aws.amazon.com/firehose/。
2. 選擇 Create Delivery Stream (建立交付串流)。
3. 在 Delivery stream name (交付串流名稱) 方塊，輸入 **log-delivery-stream**。
4. 針對 Source (來源)，選擇 Direct PUT or other sources (直接 PUT 或其他來源)。

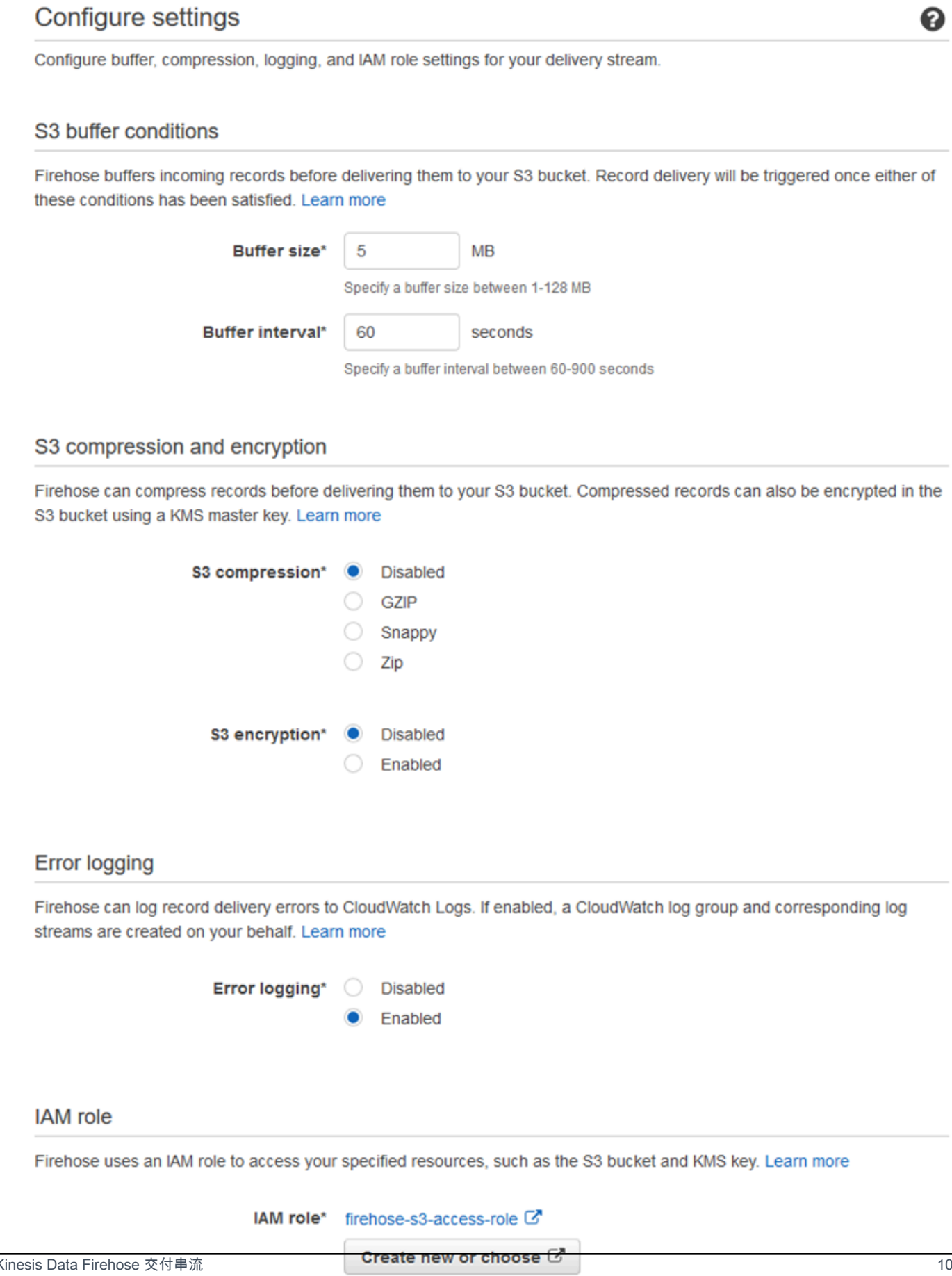


5. 選擇下一步。
6. 再次選擇 Next (下一步)。
7. 針對目標，選擇Amazon S3。
8. 針對 S3 bucket (S3 儲存貯體)，選擇您在建立 Amazon S3 儲存貯體中建立的儲存貯體名稱。



9. 選擇下一步。
10. 在 Buffer interval (緩衝間隔) 方塊中，輸入 **`60`**。
11. 在 IAM role (IAM 角色) 下方，選擇 Create new or choose (建立新項目或選擇)。
12. 針對 IAM role (IAM 角色)，選擇 `firehose-s3-access-role`。

13. 選擇 Allow (允許)。


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. 選擇下一步。
15. 選擇 Create Delivery Stream (建立交付串流)。

建立 Amazon EC2 執行個體以執行適用於視窗的 Kinesis 代理程式

建立 EC2 執行個體，使用 Windows 版本 Kinesis Agent，透過 Kinesis Data Firehose 串流日誌記錄。

建立 EC2 執行個體

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 遵循 Amazon EC2 Windows 執行個體入門中的說明，使用下列額外步驟：
   - 針對執行個體的 IAM role (IAM 角色)，選擇 `kinesis-agent-instance-role`。
   - 如果您尚未擁有公用網際網路連線的虛擬私有雲端 (VPC)，請遵循使用 Amazon EC2 進行設定中的Amazon EC2 執行個體使用者指南。
   - 建立或使用安全群組，限制對執行個體的存取只能來自您的電腦，或是您組織的電腦。如需詳細資訊，請參閱「」使用 Amazon EC2 進行設定中的Amazon EC2 執行個體使用者指南。
   - 若您指定現有的金鑰對，請確認您有權存取金鑰對的私有金鑰。或者，建立新的金鑰對並將私有金鑰儲存在安全的位置。
   - 在繼續之前，請等待執行個體進入執行中狀態，並完成全部兩項運作狀態檢查。
   - 您的執行個體需要一個公有 IP 地址。如果尚未配置，請依照彈性 IP 地址中的Amazon EC2 執行個體使用者指南。

後續步驟

步驟 2：安裝、設定及執行適用於視窗的 Kinesis 代理程式

步驟 2：安裝、設定及執行適用於視窗的 Kinesis 代理程式

在此步驟中，您會使用 AWS 管理主控台遠端連線至您在建立 Amazon EC2 執行個體以執行適用於視窗的 Kinesis 代理程式。您接著會在執行個體上安裝 Amazon Kinesis imple Storage Agent，建立及部署適用於 Windows 的 Kinesis Agent 的組態檔案，然後啟動智慧型手機服務。

1. 透過 Remore 桌面通訊協定 (RDP) 遠端連線至執行個體，請遵循步驟 2：連結到您的執行個體中的Amazon EC2 執行個體使用者指南。

2. 在執行個體上，使用 Windows 伺服器管理員停用使用者和系統管理員的 Microsoft Internet Explorer 增強式安全性設定。如需詳細資訊，請參閱 Microsoft TechNet 網站上的 How To Turn Off Internet Explorer Enhanced Security Configuration。
3. 在執行個體上，安裝並設定適用於 Windows 的 Kinesis 代理程式。如需詳細資訊，請參閱 安裝適用於視窗的 Kinesis 代理程式。
4. 在執行個體上，使用記事本建立 Windows 組態檔的 Kinesis Agent (Kinesis Agent)。將檔案儲存至 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json。將以下內容新增至組態檔案：

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

這個檔案會設定 Windows 的 Kinesis 代理程式，以便從c:\logsource\目錄 (source) 新增至 Kinesis Data Firehose 交付串流 (名為log-delivery-stream(sink。在每個日誌記錄串流至 Kinesis Data Firehose 前，它會使用兩個包含電腦名稱和時間戳記的額外鍵/值對增加。

5. 建立 c:\LogSource\ 目錄，然後使用記事本在該目錄中建立包含以下內容的 test.log 檔案：

```
{ "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. 在提高權限執行的 PowerShell 工作階段中，使用以下命令啟動 AWSKinesisTap 服務：

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

7. 使用檔案總管，瀏覽至 %PROGRAMDATA%\Amazon\AWSKinesisTap\logs 目錄。開啟最近的日誌檔案。日誌檔案看起來應該會和以下內容相似：

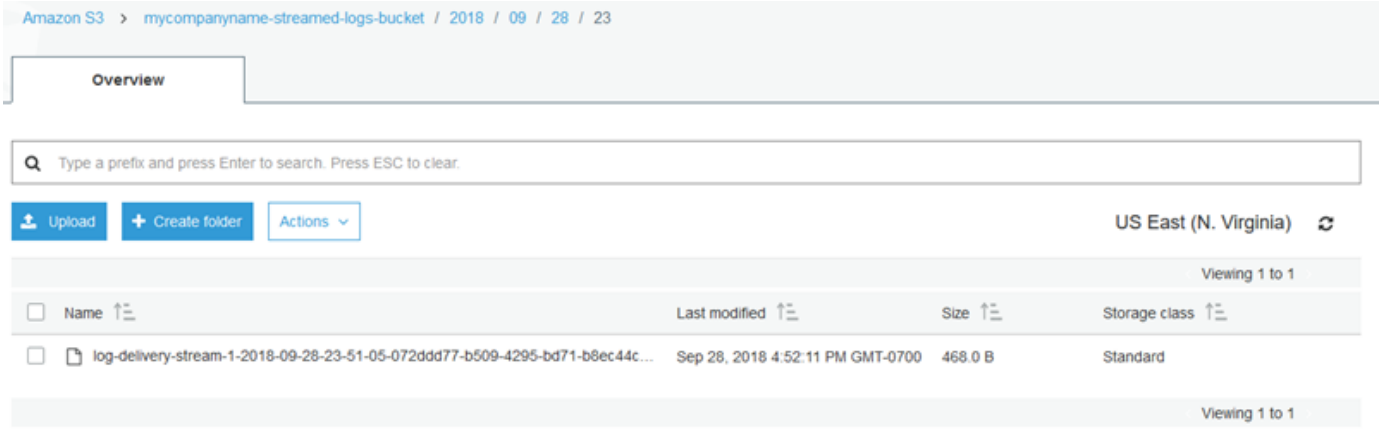
```
2018-09-28 23:51:02.2472 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.AWS.AWSEventSinkFactory.
2018-09-28 23:51:02.2784 Amazon.KinesisTap.Hosting.LogManager INFO Registered
 factory Amazon.KinesisTap.Windows.PerformanceCounterSinkFactory.
2018-09-28 23:51:02.5753 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.
```

此日誌檔案會指出服務已啟動，並且現在已會從 c:\LogSource\ 目錄收集日誌記錄。每一行都會剖析為單一 JSON 物件。電腦名稱和時間戳記的鍵/值對會新增至每個物件。然後，它會串流至 Kinesis Data Firehose。

8. 在一兩分鐘內，導覽至您在中建立的 Amazon S3 儲存貯體建立 Amazon S3 儲存貯體使用 AWS 管理主控台。確認您已在主控台上選擇正確的區域。

   在該儲存貯體中，有一個目前年份的資料夾。開啟該資料夾來顯示目前月份的資料夾。開啟該資料夾來顯示目前日期的資料夾。開啟該資料夾來顯示目前小時 (UTC) 的資料夾。開啟該資料夾來顯示名稱開頭為 log-delivery-stream 的一或多個項目。



9. 開啟最新項目的內容，確認日誌記錄已成功存放在 Amazon S3 中，並包含所需的增強。若一切都設定正確，內容看起來會和以下內容相似：

```
{"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. 如需解決任何下列問題的資訊，請參閱針對微軟視窗的 Amazon Kinesis 代理程式疑難排解：

    - Windows 日誌檔案的 Kinesis Agent 包含錯誤。
    - Amazon S3 中的預期資料夾或項目不存在。

- Amazon S3 項目的內容不正確。

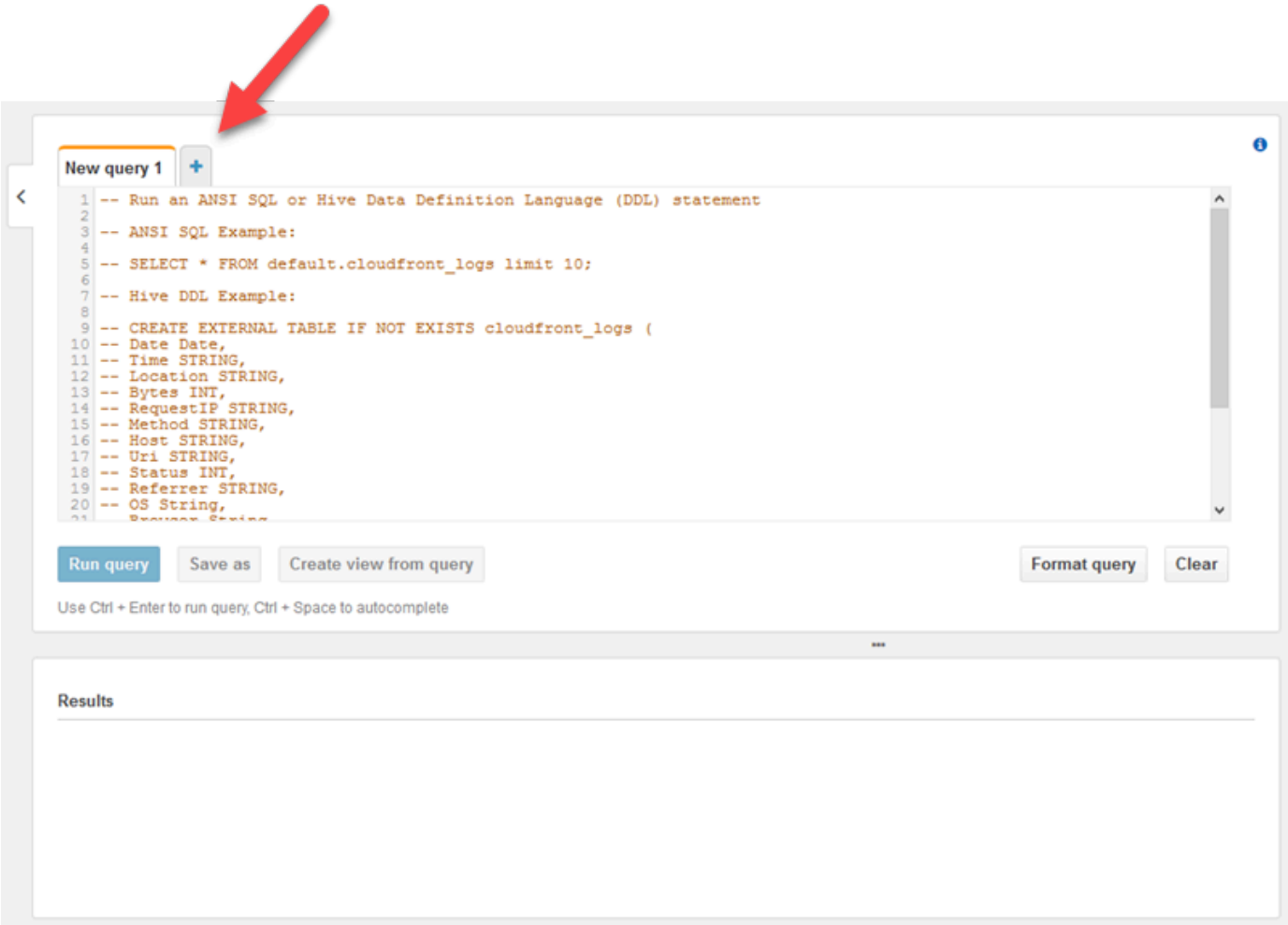
後續步驟

步驟 3：在 Amazon S3 中查詢日誌資料

步驟 3：在 Amazon S3 中查詢日誌資料

在這個 Amazon Kinesis 微軟視窗的最後一個步驟教學課程，您使用 Amazon Athena Service (Amazon S3) 中存放在 Amazon Simple Storage Service (Amazon S3) 日誌資料。

1. 前往 https://console.aws.amazon.com/athena/ 開啟 Athena 主控台。
2. 選擇加號 (+)，建立新的查詢視窗。

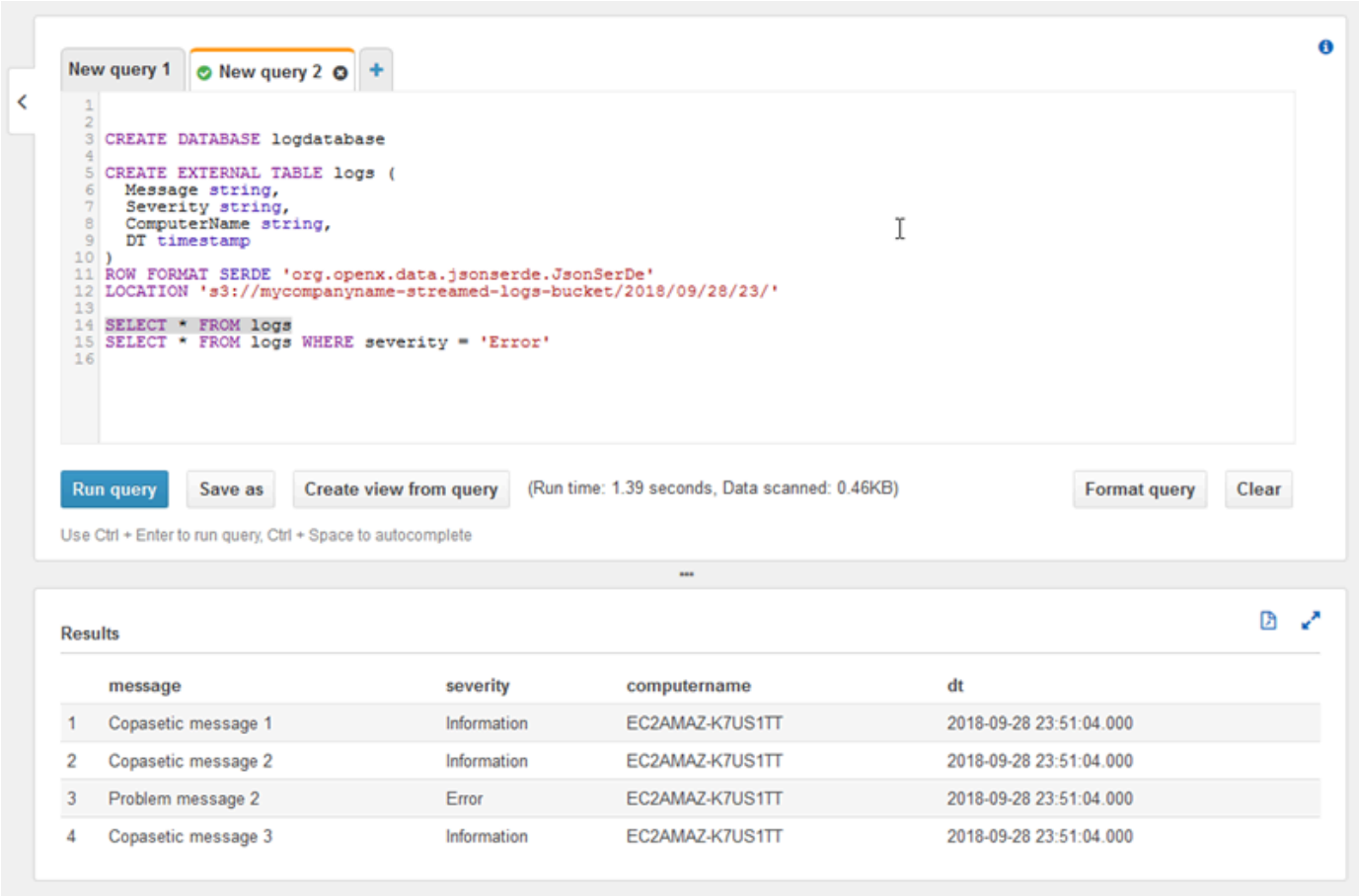


3. 將以下文字輸入查詢視窗：

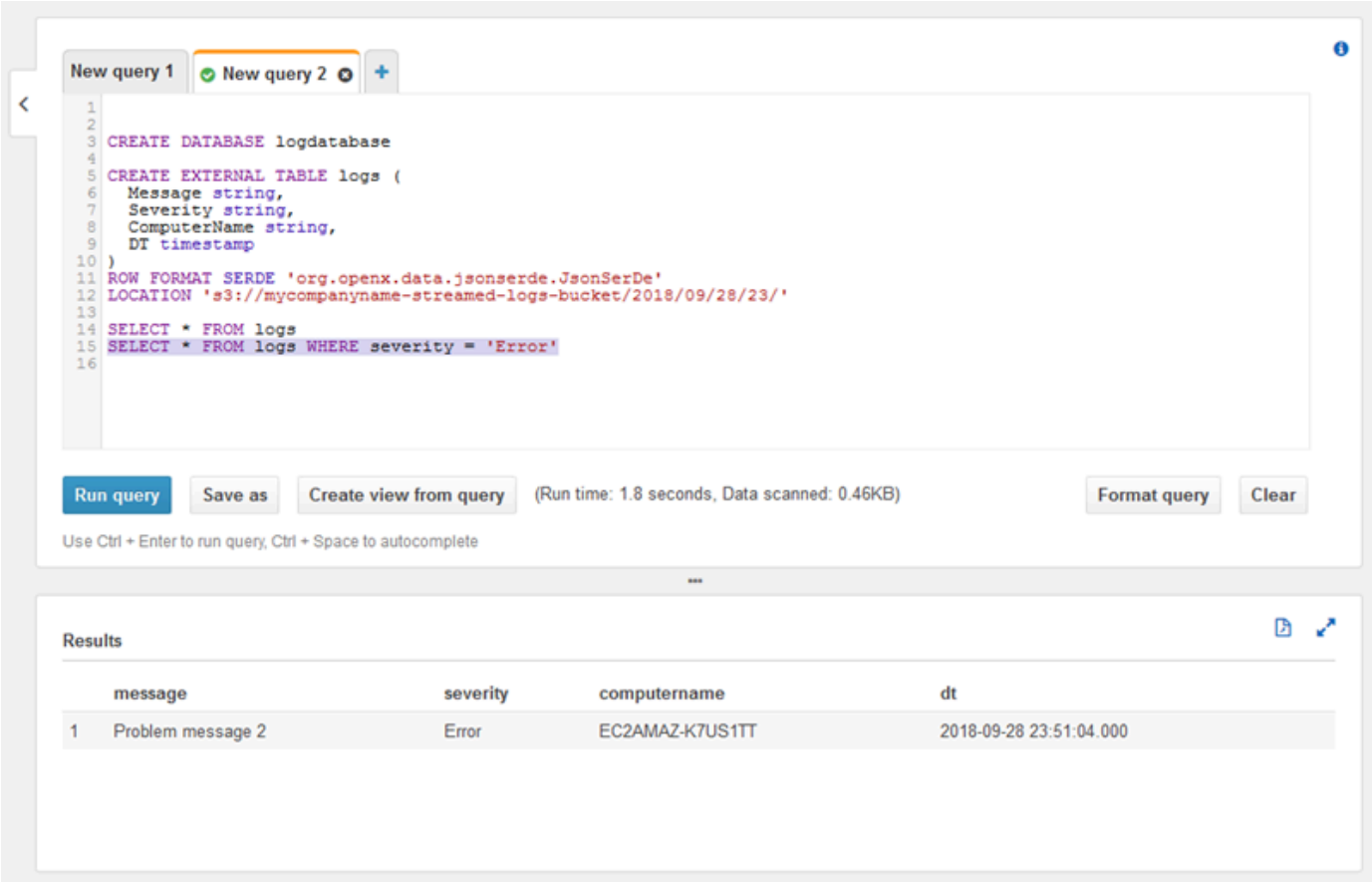
```
CREATE DATABASE logdatabase

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'
```

將 *bucket* 替換成您在建立 Amazon S3 儲存貯體中建立的儲存貯體名稱。Replace*year*、*month*、*day*和*hour*，其中包含 Amazon S3 日誌檔案建立時的年份、月份、日期及小時 (UTC)。

4. 選取 CREATE DATABASE 陳述式的文字，然後選擇 Run query (執行查詢)。這會在 Athena 中建立日誌資料庫。
5. 選取 CREATE EXTERNAL TABLE 陳述式的文字，然後選擇 Run query (執行查詢)。這會建立一個參考包含日誌資料 S3 儲存貯體的 Athena 資料表，並將 JSON 的結構描述映射至 Athena 資料表的結構描述。
6. 選取第一個 SELECT 陳述式的文字，然後選擇 Run query (執行查詢)。這會顯示資料表中所有的資料列。



7. 選取第二個 SELECT 陳述式的文字，然後選擇 Run query (執行查詢)。這會只顯示資料表中代表包含 `Error` 層級嚴重性日誌記錄的資料列。這類查詢會從潛在的大型日誌記錄集中尋找有趣的日誌記錄。



後續步驟

使用 AWS 管理主控台清除在教學期間建立的資源：

1. 終止 EC2 執行個體 (請參閱 Amazon EC2 Windows 執行個體入門中的步驟 3)。

   > **Important**
   >
   > 如果您啟動的執行個體不在AWS 免費方案，您將需要在終止它之前支付執行個體的費用。

2. 刪除 Kinesis Data Firehose 交付串流。

   a. 開啟 Kinesis Data Firehose 主控台，網址為https://console.aws.amazon.com/firehose/。
   b. 選擇您建立的交付串流。
   c. 選擇 Delete (刪除)。
   d. 選擇 Delete delivery stream (刪除交付串流)。

3. 刪除 S3 儲存貯體。如需說明，請參閱「」如何刪除 S3 儲存貯體？中的Amazon Storage Service 主控台使用者指南。

如需詳細資訊，請參閱下列主題：

- 設定適用於微軟視窗的 Amazon Kinesis
- 什麼是 Amazon Kinesis Data Firehose？
- 什麼是 Amazon S3？
- 什麼是 Amazon Athena？

針對微軟視窗的 Amazon Kinesis 代理程式疑難排解

使用下列說明來診斷及修正使用適用於微軟視窗的 Amazon Kinesis 代理程式時的問題。

主題

桌面平台或伺服器沒有任何資料串流至預期的 AWS 服務

Symptoms

當您檢查各種設為從 Windows 專用 Kinesis 代理接收資料串流的 AWS 服務所託管的日誌、事件和指標時，Windows 專用 Kinesis 代理程式並未串流任何資料。

Causes

此問題有數個可能的原因：

- 來源、接收或管道設定不正確。
- Windows Kinesis 代理程式的身份驗證設定不正確。
- Windows Kinesis 代理程式的授權設定不正確。
- `DirectorySource` 宣告中所提供的規則表達式有錯誤。
- 針對 `DirectorySource` 宣告指定了不存在的目錄。
- 提供給 AWS 服務的是無效值，因此拒絕了 Windows Kinesis 代理程式的請求。
- 接收正在參考在指定或隱含 AWS 區域中不存在的資源。
- 為 `WindowsEventLogSource` 宣告指定的查詢無效。
- 為來源 `InitialPosition` 鍵/值對指定的值無效。
- `appsettings.json` 組態檔案不符合該檔案的 JSON 結構描述。

- 資料正在串流至與在 AWS 管理主控台中選取的區域不同的區域。
- Windows 適用的 Kinesis 代理程式並未正確安裝或並未在執行中。

Resolutions

若要解決沒有串流資料的問題，請執行下列步驟：

1. 請檢查 Kinesis 代理程式的%PROGRAMDATA%\Amazon\AWSKinesisTap\logs目錄中。搜尋字串 ERROR。
   a. 若來源或接收並未載入，請執行以下作業：
      i. 檢查錯誤訊息，並尋找來源或接收的 Id。
      ii. 檢查對應至 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中 Id 的來源或接收宣告，查看是否有任何與所找到錯誤訊息相關的錯誤。如需詳細資訊，請參閱 設定適用於微軟視窗的 Amazon Kinesis。
      iii. 修正與錯誤相關的任何組態檔案問題。
      iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。
   b. 若錯誤訊息指出並未找到管道的 SourceRef 或 SinkRef，請執行以下作業：
      i. 記下管道的 Id。
      ii. 檢查 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中對應至所記下 Id 的管道宣告。確認 SourceRef 和 SinkRef 鍵/值對的值已正確拼為您意圖參考的來源和接收宣告 Id。修正任何錯誤或拼字錯誤。若組態檔案中沒有來源或接收宣告，請新增宣告。如需詳細資訊，請參閱 設定適用於微軟視窗的 Amazon Kinesis。
      iii. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。
   c. 若錯誤訊息指出特定 IAM 使用者或角色並未獲得授權執行特定操作，請執行以下作業：
      i. 確認適用於 Windows 的 Kinesis 代理程式正在使用正確的 IAM 使用者或角色。如果不是，請檢閱目的地安全組態，並調整 Windows 適用 Kinesis 代理程式的身份驗證方式，確保使用的是正確的 IAM 使用者或角色。
      ii. 若使用的是正確的 IAM 使用者或角色，請使用 AWS 管理主控台，檢查與使用者或角色相關聯的政策。確保使用者或角色針對 Windows 用 Kinesis 代理程式存取的所有 AWS 資源，皆具備所有在錯誤訊息中提及的許可。如需詳細資訊，請參閱 設定授權。
      iii. 停止並啟動 AWSKinesisTap 服務。然後，檢查最近的日誌檔案，驗證已解決安全問題。
   d. 若錯誤訊息指出在剖析包含於 %PROGRAMFILES%\Amazon\AWSKinesisTap \appsettings.json 組態檔案中的規則表達式時發生引數錯誤，請執行以下作業：

i. 檢查組態檔案中的規則表達式。

ii. 驗證規則表達式的語法。您可以使用數個網站驗證規則表達式，或是使用下列命令列來檢查 DirectorySource 來源宣告的規則表達式：

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

將 *sourceId* 替換成具有不正確規則表達式 DirectorySource 來源宣告的 Id 鍵/值對值。

iii. 對組態檔案中的規則表達式進行任何必要的修正，使其有效。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

e. 若錯誤訊息指出在剖析並未包含於 %PROGRAMFILES%\Amazon\AWSKinesisTap \appsettings.json 組態檔案中的規則表達式時發生引數錯誤，並且該錯誤與特定接收有關，請執行以下作業：

i. 找出組態檔案中的接收宣告。

ii. 驗證明確與 AWS 服務相關的鍵/值對使用符合該服務驗證規則的名稱。例如，CloudWatch Logs 群組名稱必須只能包含由規則表達式指定的一組字元。[\.\-_/#A-Za-z0-9]+。

iii. 修正接收宣告鍵/值對中的任何無效名稱，然後確認那些資源已適當在 AWS 中設定。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

f. 若錯誤訊息指出來源或接收因 null 或遺失參數而無法載入，請執行以下作業：

i. 記下來源或接收的 Id。

ii. 在 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中找出符合所記下 Id 的來源或接收宣告。

iii. 檢閱在來源或接收宣告中提供的鍵/值對，並與相關接收類型 設定適用於微軟視窗的 Amazon Kinesis 文件中的來源或接收類型需求進行比較。將任何遺失的必要鍵/值對新增至來源或接收宣告。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

g. 若錯誤訊息指出目錄名稱無效，請執行以下作業：

i. 在 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中找出無效的目錄名稱。

ii. 驗證此目錄存在並包含應串流的日誌檔案。

iii. 修正組態檔案中所指定目錄名稱內的任何錯字或錯誤。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

h. 若錯誤訊息指出資源不存在：

i. 找出 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中接收宣告內不存在資源的資源參考。

ii. 使用 AWS 管理主控台在接收宣告內應使用的正確 AWS 區域中找出資源。將它與組態檔案中指定的內容進行比較。

iii. 變更組態檔案中的接收宣告，使其擁有正確的資源名稱和正確的區域。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

i. 若錯誤訊息指出針對特定 WindowsEventLogSource 的查詢無效，請執行以下作業：

i. 在 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中，找出與錯誤訊息具有相同 Id 的 WindowsEventLogSource 宣告。

ii. 驗證來源宣告中 Query 鍵/值對的值符合 Event queries and Event XML。

iii. 對查詢進行任何變更，使其合規。

iv. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

j. 若錯誤訊息指出其中具有無效的初始位置，請執行以下作業：

i. 在 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案中，找出與錯誤訊息具備相同 Id 的來源宣告。

ii. 變更來源宣告中 InitialPosition 鍵/值對的值，使其符合所允許的值，如書籤組態中所述。

iii. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

2. 確認 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案與 JSON 結構描述相符。

a. 在命令提示視窗中，呼叫下列各行：

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

b. 修正針對 %PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json 組態檔案偵測到的任何問題。

c. 停止並啟動 AWSKinesisTap 服務。然後檢查最近的日誌檔案，驗證已解決組態問題。

3. 變更記錄日誌層級，嘗試取得更詳細的記錄日誌資訊。

a. 使用以下內容替換 %PROGRAMFILES%\Amazon\AWSKinesisTap\nlog.xml 組態檔案：

```
<?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. 停止並啟動 AWSKinesisTap 服務。然後檢查最新的日誌檔案，查看日誌中是否有額外的訊息可協助診斷及解決問題。

4. 驗證您在 AWS 管理主控台中正確區域內查看資源。

5. 驗證 Windows 代理程式的 Kinesis 代理程式已安裝並在執行中。

a. 在 Windows 中，選擇 Start (啟動)，然後導覽至 Control Panel (控制台)、Administrative Tools (系統管理工具)、Services (服務)。

b. 尋找 AWSKinesisTap 服務。

c. 若沒有看到 AWSKinesis 服務，請使用Amazon Kinesis 代理程式入門。

d. 若能看到服務，請判斷服務是否正在執行中。若不在執行中，請開啟服務的內容 (以滑鼠右鍵按一下) 選單，然後選擇 Start (啟動)。

e. 檢查 %PROGRAMDATA%\Amazon\AWSKinesisTap\logs 目錄中最新的日誌檔案，驗證服務已啟動。

適用對象

此資訊適用於 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.115 版及更新版本的 Kinesis 代理

預期資料有時候會遺失

Symptoms

大多數的時候都能成功串流資料的 Kinesis 代理程式，但有時候會遺失某些資料。

Causes

此問題有數個可能的原因：

- 並未使用書籤功能。
- 根據這些服務目前的組態，超過 AWS 服務的資料傳輸速率限制。
- AWS 服務的 API 呼叫費率限制，根據目前的appsettings.json組態檔案和 AWS 帳戶限制。

Resolutions

若要解決遺失資料的問題，請執行下列步驟：

1. 考慮使用書籤組態中說明的書籤功能。它可協助確保所有資料最終都會傳送，即使 Windows 適用 Kinesis 代理程式停止並啟動時也一樣。
2. 針對 Windows 的內建指標使用 Kinesis 代理程式，探索問題：
   a. 啟用 Windows 指標的 Kinesis 代理程式串流，如設定 Windows 度量管道的 Kinesis 代理程式。
   b. 若一或多個接收有大量不可復原的錯誤，請判斷每秒傳送多少位元組或記錄。然後判斷其是否仍在串流資料區域及帳戶中，針對那些 AWS 服務所設定的限制範圍內。

c. 若超過限制，請減少傳送的資料量或速率、請求提高限制，或是增加分片 (若適用的話)。

d. 在進行調整之後，繼續監控 Windows 專用的 Kinesis 代理程式，確保情況已解決。

如需 Kinesis Data Streams 限制的詳細資訊，請參閱Kinesis Data Streams 限制中的Kinesis Data Streams 開發人員指南。如需 Kinesis 資料 Firehose 限制的詳細資訊，請參閱Amazon Kinesis Data Firehose 限制。

適用對象

此資訊適用於 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.115 版及更新版本的 Kinesis 代理

資料以不正確的格式抵達

Symptoms

資料以不正確的格式抵達 AWS 服務。

Causes

此問題有數個可能的原因：

- `appsettings.json` 組態檔案中接收宣告 `Format` 鍵/值對的值不正確。
- `DirectorySource` 宣告中 `RecordParser` 鍵/值對的值不正確。
- `DirectorySource` 宣告中使用 `Regex` 記錄剖析器的規則表達式不正確。

Resolutions

若要解決不正確格式化的問題，請執行下列步驟：

1. 檢閱 `%PROGRAMFILES%\Amazon\AWSKinesisTap\appsettings.json` 組態檔案中的接收宣告。
2. 確認已為每個接收宣告指定 `Format` 鍵/值對正確的值。如需詳細資訊，請參閱 目的地宣告。
3. 若具有 `DirectorySource` 宣告的來源已使用管道與針對 `Format` 鍵/值對指定 `xml` 或 `json` 值的接收連線，請確認那些資源已為 `RecordParser` 鍵/值對指定下列其中一個值：
   - `SingleLineJson`

- Regex
- SysLog
- Delimited

其他記錄剖析器都是僅限文字類型，無法與需要 XML 或 JSON 格式化的接收搭配並正常運作。

4. 若日誌記錄無法由 DirectorySource 來源類型正確地進行剖析，請在命令提示視窗中呼叫下列各行，驗證在 DirectorySource 宣告中指定的時間戳記和規則表達式鍵/值對：

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

將 *sourceID* 替換成並未正常運作 DirectorySource 來源宣告 Id 鍵/值對的值。修正任何由 ktdiag.exe 報告的問題。

適用對象

此資訊適用於 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.115 版及更新版本的 Kinesis 代理

效能問題

Symptoms

在安裝並啟動 Windows 版 Kinesis 代理程式後，應用程式和服務的延遲增加。

Causes

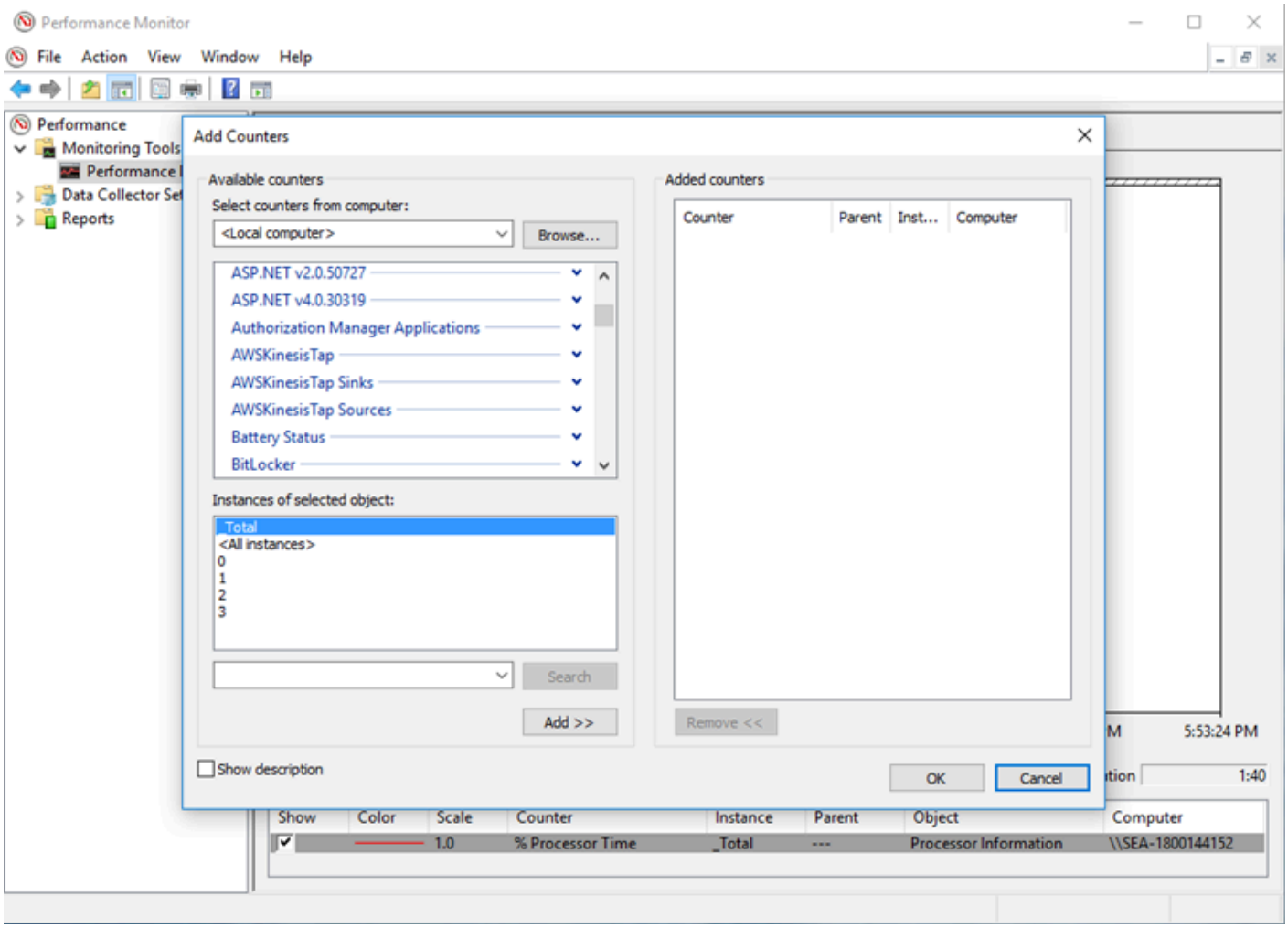
此問題有數個可能的原因：

- Windows 適用 Kinesis 代理程式執行的機器沒有足夠的容量串流所需的資料量。
- 不必要的資料正在串流至一或多個 AWS 服務。
- Windows 適用 Kinesis 代理程式正在將資料串流至並未針對高資料傳輸速率進行設定的 AWS 服務。
- Windows Kinesis 代理程式正在 API 呼叫速率限制過低的帳戶中，呼叫 AWS 服務的操作。

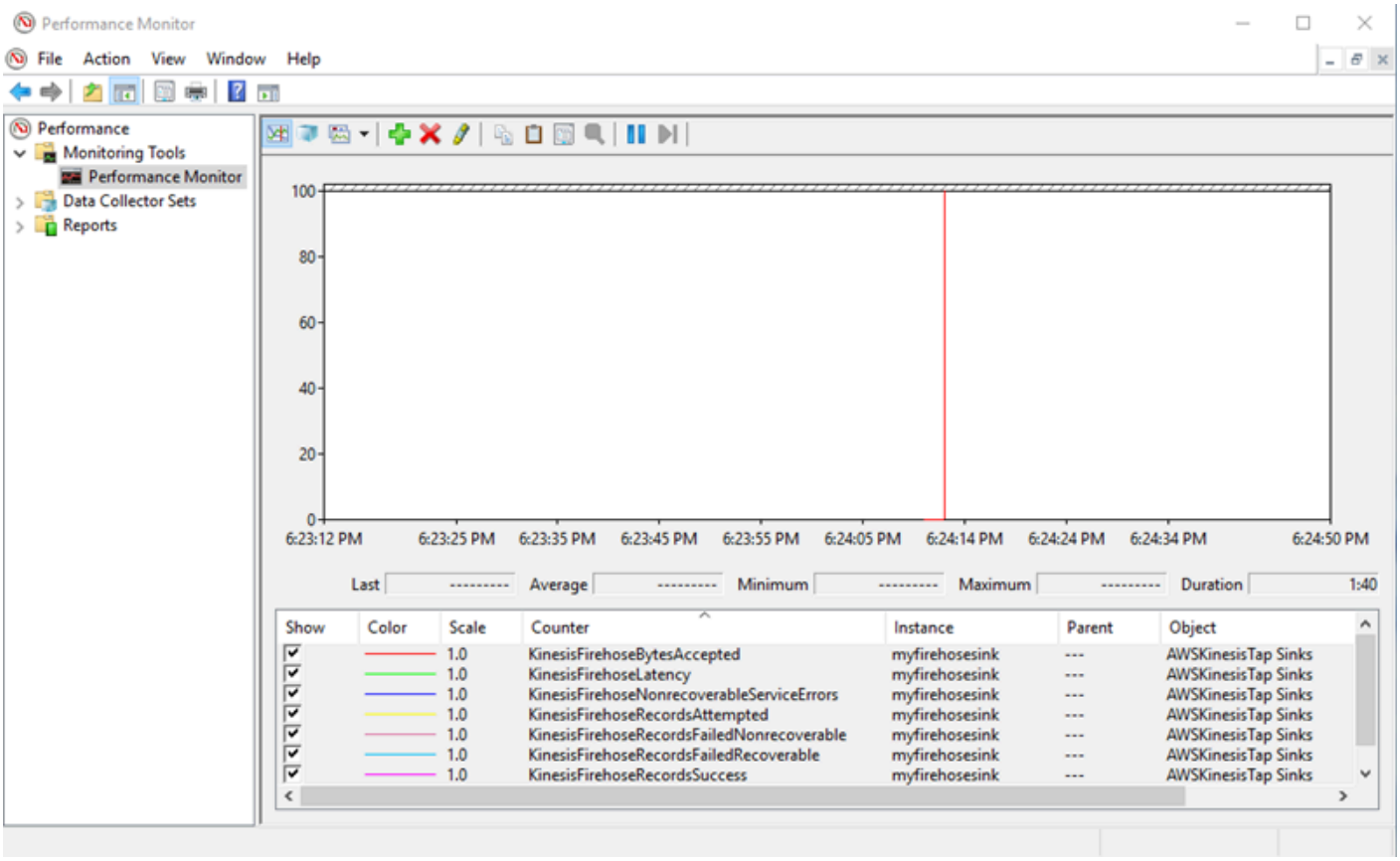
Resolutions

若要解決效能問題，請執行下列步驟：

1. 使用 Windows 資源監控應用程式，檢查記憶體、CPU、磁碟和網路用量。若您需要使用 Windows 專用 Kinesis Agent 串流大量資料，取決於組態，您可能需要在某些區域內佈建容量較高的機器。
2. 您可能可以使用篩選，減少記錄資料的數量：
   - 請參閱 WindowsEventLogSource 組態中的 `Query` 鍵/值對。
   - 請參閱設定管道中的管道篩選。
   - 請參閱中的 Amazon CloudWatch 指標篩選CloudWatch Logs 目的地組態。
3. 使用 Windows 效能監控應用程式，檢視 Windows 指標的 Kinesis 代理程式，或將那些指標串流至 CloudWatch (請參閱Windows 內建指標來源適用的 Kinesis 代理程式。在 Windows 效能監控應用程式中，您可以為 Windows 接收和來源的 Kinesis 代理程式新增計數器。他們會列在 AWSKinesisTap Sinks (AWSKinesisTap 接收) 和 AWSKinesisTap Sources (AWSKinesisTap 來源) 下方。



例如，若要診斷 Kinesis Data Firehose 效能問題，請新增Kinesis Firehose效能計數器。



若有大量的可還原的錯誤，請檢查最新 Kinesis 代理程式，查看%PROGRAMDATA%\Amazon\AWSKInesisTap\logs目錄中。若 KinesisStream 或 KinesisFirehose 接收發生調節，請執行以下作業：

- 若因為串流資料的速度過快導致發生調節，請考慮提高 Kinesis 資料串流的碎片數。如需詳細資訊，請參閱「」重新分片、擴展和平行處理中的Kinesis Data Streams 開發人員指南。
- 考慮提高 Kinesis Data Streams 的 API 呼叫限制，或增加接收的緩衝區大小 (若 API 呼叫遭到調節)。如需詳細資訊，請參閱「」Kinesis Data Streams 限制中的Kinesis Data Streams 開發人員指南。
- 若資料串流的速度過快，請考慮請求提高 Kinesis Data Firehose 交付串流的速率限制。或者，若 API 呼叫遭到調節，請請求提高 API 呼叫限制 (請參閱 Amazon Kinesis Data Firehose 限制) 或增加接收的緩衝區大小。
- 在增加 Kinesis Data Streams 的碎片數，或增加 KKinesis Data Firehose 串流的速率限制後，請修訂 Windows 專用的 Kinesis 代理程式。appsettings.json組態檔案，增加接收的每秒記錄數或每秒位元組數。否則，Windows 版 Kinesis 代理程式將無法利用增加的限制。

適用對象

此資訊適用於 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.115 版及更新版本的 Kinesis 代理

磁碟空間不足

Symptoms

適用於 Windows 的 Kinesis 代理程式正在一或多個磁碟機的磁碟空間非常少的機器上執行。

Causes

此問題有數個可能的原因：

- Windows 記錄日誌組態檔案的 Kinesis 代理程式不正確。
- 適用於 Windows 持久性佇列的 Kinesis 代理程式設定不正確。
- 某些其他的應用程式或服務正在使用磁碟空間。

Resolutions

若要解決磁碟空間問題，請執行下列步驟：

- 若包含 Windows 日誌檔案 Kinesis 代理程式磁碟上的磁碟空間過低，請檢查日誌檔案目錄 (通常是`%PROGRAMDATA%\Amazon\AWSKinesisTap\logs`。確認保留的日誌檔案數量合理，且日誌檔案的大小也合理。您可以藉由編輯 Windows Kinesis 記錄檔的位置、保留期和詳細程度，控制 Windows 記錄檔的位置、保留期和詳細程度。`%PROGRAMFILES%\Amazon\AWSKinesisTap\Nlog.xml`組態檔案。
- 當啟用接收排入佇列功能時，請檢查使用該功能的接收宣告。確認 `QueuePath` 鍵/值對參考具有足夠空間，可包含使用 `QueueMaxBatches` 鍵/值對所指定批次數量上限的磁碟機。若無法執行該作業，請減少 `QueueMaxBatches` 鍵/值對的值，使資料能輕易容納在指定磁碟機的剩餘磁碟空間內。
- 使用 Windows 檔案總管，找出使用磁碟空間的檔案，並傳輸或刪除多餘的檔案。變更使用大量磁碟空間的應用程式或服務組態。

適用對象

此資訊適用於 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.115 版及更新版本的 Kinesis 代理

故障診斷工具

除了驗證組態檔案，您也能使用ktdiag.exe工具，該工具可提供數種其他的功能，用於診斷及解決設定和使用針對 Windows 的 Kinesis 代理程式時的問題。ktdiag.exe 工具位於 %PROGRAMFILES%\Amazon\AWSKinesisTap 目錄中。

- 若您認為包含特定檔案模式的日誌檔案正在寫入目錄，但並未由適用於 Windows 的 Kinesis 代理程式處理，請使用/w參數，以確認偵測到這些變更。例如，假設您預期具有 *.log 檔案名稱模式的日誌檔案正在寫入 c:\foo 目錄。您可以在執行 ktdiag.exe 工具時使用 /w 開關，指定該目錄及檔案模式：

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

  若正在寫入日誌檔案，您可以看到與以下內容相似的輸出：

  ```
  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
  ```

  若沒有發生這類輸出，表示寫入日誌時發生應用程式或服務問題，或者發生安全組態問題 (而非 Windows 版 Kinesis Agent)。若發生這類輸出，但 Windows 版 Kinesis 代理程式仍未明顯地處理日誌，請參閱。桌面平台或伺服器沒有任何資料串流至預期的 AWS 服務。

- 有時候，日誌只會偶爾寫入，但驗證 Windows 版 Kinesis 代理程式是否正常運作很有用。請使用 /log4net 開關來模擬使用 Log4net 程式庫寫入日誌的應用程式；例如：

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

  這會將 Log4net 樣式的日誌檔案寫入 c:\foo\log2.log 日誌檔案，並持續新增新的日誌項目，直到按下任意鍵為止。您可以在檔案名稱後方選擇性地指定使用額外的開關，設定數種選項：

鎖定：-lm、-li 或 -le

您可以指定下列其中一項鎖定開關，控制鎖定日誌檔案的方式：

-lm

會在日誌檔案上使用最低數量的鎖定，提供日誌檔案最大程度的存取。

-li

只有相同程序內的執行緒可同時存取日誌。

-le

一次只能有一個執行緒存取日誌。此為預設值。

-tn:*##*

指定寫入日誌項目間的*##*數。預設為 1000 毫秒 (1 秒)。

-sm:*###*

指定每個日誌項目的*###*數。預設為 1000 位元組。

-bk:*number*

指定一次要寫入的日誌項目*##*。預設為 1。

- 有時候模擬寫入 Windows 事件日誌的應用程式會很有用。使用 /e 開關來將日誌項目寫入 Windows 事件日誌；例如：

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

這會將日誌項目寫入 Windows 應用程式事件日誌，直到按下任意鍵為止。您可以選擇性地在日誌的名稱後方指定下列額外選項：

-tn:*##*

指定寫入日誌項目間的*##*數。預設為 1000 毫秒 (1 秒)。

-sm:*###*

指定每個日誌項目的*###*數。預設為 1000 位元組。

-bk:*number*

指定一次要寫入的日誌項目*##*。預設為 1。

為視窗外掛程式建立 Kinesis 代理程式

在大多數情況下，不需要為微軟視窗外掛程式建立 Amazon Kinesis Kinesis 代理程式。針對 Windows 的 Kinesis Agent 可以高度地進行設定，並包含了強大的來源和接收，例如`DirectorySource`和`KinesisStream`，這對於大多數情況來說已經足夠了。如需現有來源和接收的詳細資訊，請參閱設定適用於微軟視窗的 Amazon Kinesis。

針對不尋常的案例，便可能需要使用自訂外掛程式擴充針對 Windows 的 Kinesis Agent。其中某些案例包含以下項目：

- 使用 `Regex` 或 `Delimited` 記錄剖析器封裝複雜的 `DirectorySource` 宣告，使其易於套用在許多不同類型的組態檔案中。
- 建立並非以檔案為基礎，或是超過現有記錄剖析器所提供剖析功能的創新來源。
- 目前不支援建立 AWS 服務的接收。

主題

開始使用適用於 Windows 外掛程式的 Kinesis 代理程式

自訂外掛程式並不特殊。所有現有的來源和接收都使用和 Windows 相同的機制，和自訂外掛程式相同，在啟 Kinesis 時進行載入，並且會在讀取`appsettings.json`組態檔案。

當 Windows 版 Kinesis 代理程式啟動時，便會發生以下順序：

1. 針對 Windows 的 Kinesis Assembly 會掃描安裝目錄內的組件 (`%PROGRAMFILES%\Amazon\AWSKinesisTap`），用於實現`IFactory<T>`中定義的介面`Amazon.KinesisTap.CoreAssembly`。此界面是在`Amazon.KinesisTap.Core\Infrastructure\IFactory.cs`，以取得 Windows 原始程式碼。
2. 針對 Windows 的 Kinesis Agent 會載入包含這些類別的組件，並呼叫`RegisterFactory`這些類的方法。

3. 適用於視窗的 Kinesis 代理程式會載入appsettings.json組態檔案。針對組態檔案中的每個來源和接收，檢查 SourceType 和 SinkType 鍵/值對。若有使用相同名稱做為 SourceType 和 SinkType 鍵/值對值註冊的工廠，便會在這些工廠上呼叫 CreateInstance 方法。將組態和其他資訊做為 IPluginContext 物件傳遞給 CreateInstance 方法。CreateInstance 方法會負責設定及初始化外掛程式。

若要讓外掛程式正常運作，必須擁有建立外掛程式的註冊工廠類別，並且也必須定義外掛程式的類別本身。

針對 Windows 來源碼的 Kinesis 代理程式位於https://github.com/awslabs/kinesis-agent-windows。

為視窗外掛程式工廠實作 Kinesis 代理程式

遵循下列步驟來實作針對 Windows 外掛程式工廠的 Kinesis 代理程式。

建立適用於視窗外掛程式工廠的 Kinesis 代理程式

1. 建立瞄準 .NET Framework 4.6 的 C# 程式庫專案。
2. 新增 Amazon.KinesisTap.Core 組件的參考。此組件位於%PROGRAMFILES%\Amazon\AWSKinesisTap目錄 Kinesis 行 Windows 安裝。
3. 使用 NuGet 安裝 Microsoft.Extensions.Configuration.Abstractions 套件。
4. 使用 NuGet 安裝 System.Reactive 套件。
5. 使用 NuGet 安裝 Microsoft.Extensions.Logging 套件。
6. 建立工廠類別，針對來源實作 IFactory<IEventSource>，或針對接收實作 IFactory<IEventSink>。新增 RegisterFactory 和 CreateInstance 方法。

   例如，下列程式碼會建立針對 Windows 外掛程式工廠的 Kinesis Agent，建立產生隨機資料的來源：

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

若您最後希望增強工廠來建立不同類型的執行個體，switch 陳述式會用於 CreateInstance 方法中。

若要建立接收工廠來建立不採取任何動作的接收，請使用與以下內容相似的類別：

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

```
}
```

為視窗外掛程式來源實作 Kinesis 代理程式

遵循下列步驟來實作針對 Windows 外掛程式來源的 Kinesis 代理程式。

建立 Windows 外掛程式來源的 Kinesis 代理程式

1. 新增類別，實作 IEventSource<out T> 界面至先前為來源所建立的專案。

   例如，使用以下程式碼來定義產生隨機資料的來源：

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

在此範例中，RandomSource 類別會繼承自 EventSource<T> 類別，因為它提供了 Id 屬性。雖然此範例不支援書籤，但此基礎類別在實作該功能上也相同有用。信封可提供一種方式存放中繼資料，並包裝任意資料以串流至接收。RandomData 類別會在下一個步驟中定義，並代表來自此來源輸出物件的類型。

2. 新增類別至先前定義的專案，該專案包含了從來源串流的資料。

   例如，隨機資料的容器可定義如下：

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

3. 編譯先前定義的專案。
4. 將組件複製到 Windows 的 Kinesis 代理程式的安裝目錄。
5. 建立或更新appsettings.json組態檔案，並將它置放在適用於 Windows 的 Kinesis Agent 的安裝目錄內。
6. 停止並啟動 Windows 的 Kinesis 代理程式。
7. 檢查目前的 Kinesis 代理程式是否有 Windows 記錄檔 (通常位於%PROGRAMDATA%\Amazon\AWSKinesisTap\logs目錄)，確認自訂來源外掛程式沒有任何問題。

8. 確認資料抵達所需的 AWS 服務。

如需如何擴展DirectorySource功能來實作剖析特定日誌格式，請參閱Amazon.KinesisTap.Uls\UlsSourceFactory.cs和Amazon.KinesisTap.Uls\UlsLogParser.cs，以取得 Windows 原始程式碼。

如需如何建立來源，以提供書籤功能的範例，請參閱Amazon.KinesisTap.Windows\WindowsSourceFactory.cs和Amazon.KinesisTap.Windows\EventLogSource.cs，以取得 Windows 原始程式碼。

為視窗外掛程式接收器實作 Kinesis 代理程式

遵循下列步驟來實作針對 Windows 外掛程式接收的 Kinesis 代理程式。

若要建立 Windows 外掛程式接收器的 Kinesis 代理程式

1. 新增類別至先前定義的專案，實作 IEventSink 界面。

   例如，以下程式碼會實作接收，除了記錄記錄抵達之外不採取任何動作，並隨即捨棄記錄。

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

在此範例中，NullSink 接收類別繼承自 EventSink 類別，因為它提供了將記錄轉換成不同序列化格式 (例如 JSON 和 XML) 的能力。

2. 編譯先前定義的專案。
3. 將組件複製到 Windows 的 Kinesis 代理程式的安裝目錄。
4. 建立或更新appsettings.json組態檔案，並將它置放在適用於 Windows 的 Kinesis Agent 的安裝目錄內。例如，若要使用 RandomSource 和 NullSink 自訂外掛程式，您可以使用以下 appsettings.json 組態檔案：

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

此組態會建立來源，傳送 RandomData 的執行個體，其中包含每 10 秒便會設為 0 至 50 間隨機數字的 RandomValue。它會建立一個接收，將傳入的 RandomData 執行個體轉換成 JSON，記錄該 JSON，然後捨棄執行個體。請務必在先前使用範例組態檔案的定義專案中包含兩個範例工廠 (RandomSource 來源類別及 NullSink 接收類別)。

5. 停止並啟動 Windows 的 Kinesis 代理程式。
6. 檢查目前的 Kinesis 代理程式是否有 Windows 記錄檔 (通常位於%PROGRAMDATA%\Amazon\AWSKinesisTap\logs目錄)，確認自訂接收外掛程式沒有任何問題。
7. 確認資料抵達所需的 AWS 服務。因為範例 NullSink 不會串流至 AWS 服務，您可以查看指出已接收記錄的日誌訊息，驗證接收已正常運作。

   例如，您應該會看到與以下內容相似的日誌檔案：

```
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}.
```

若您要建立存取 AWS 服務的接收，有些基礎類別可能會有所幫助。對於使用`AWSBufferedEventSink`基底類別，請參閱`Amazon.KinesisTap.AWS\CloudWatchLogsSink.cs`在原始程 Kinesis 碼中。

適用於微軟 Windows 的 Amazon Kinesis 代理程式的文件歷史記錄

API 版本：2018-10-15

下表說明Amazon Kinesis 微軟視窗代理程式使用指南(本文件)。

update-history-change	update-history-description	update-history-date
主要文件更新	新增 MSI 安裝指示。已更新目錄來源配置並添加視窗七個記錄格式來源。針對接收器組態，新增本機檔案系統同步設定；設定檔重新整理清單識別提供者；關於文字裝飾的資訊、解析接收屬性中的變數、設定接收器的 STS 區域端點、設定 VPC 端點，以及設定替代 Proxy 伺服器。對於管，已加入規劃屬性。	2021 年 2 月 23 日
更新文件	已更新的主題，指出 Amazon S3 位置指定區分大小寫。	2018 年 11 月 7 日
初始發行	第一次發行的《適用於 Windows 的 Kinesis 代理程式使用者指南》。	2018 年 11 月 5 日

AWS 詞彙表

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

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。