

Ingesting Data into Amazon S3

Date published: 2022-02-24

Date modified: 2022-02-24

CLOUDERA

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Ingesting data into Amazon S3.....	4
Understand the use case.....	4
Meet the prerequisites.....	4
Build the data flow.....	5
Set up AWS for your ingest data flow.....	7
Create IDBroker mapping.....	8
Create controller services for your data flow.....	10
Configure the processor for your data source.....	13
Configure the processor for merging records.....	15
Configure the processor for your data target.....	16
Start the data flow.....	17
Verify data flow operation.....	18
Monitoring your data flow.....	18
Next steps.....	19

Ingesting data into Amazon S3

You can use an Apache NiFi data flow to ingest data into Amazon S3 object stores in CDP Public Cloud by following these steps.

Understand the use case

Learn how you can use a Flow Management cluster connected to a Streams Messaging cluster to build an end-to-end flow that ingests data to Amazon S3 storage. This example use case shows you how to use Apache NiFi to move data from Kafka to S3 buckets.

Why move data to object stores?

Cloud environments offer numerous deployment options and services. There are many ways to store data in the cloud, but the easiest option is to use object stores. Object stores are extremely robust and cost-effective storage solutions with multiple levels of durability and availability. You can include them in your data pipeline, both as an intermediate step and as an end state. Object stores are accessible to many tools and connecting systems, and you have a variety of options to control access.

Apache NiFi for cloud ingestion

As a part of Cloudera Data Flow in CDP Public Cloud, Flow Management clusters run Apache NiFi. You can use NiFi to move data from a range of locations into object stores. NiFi supports data transfer from nearly any type of data source to cloud storage solutions in a secure and governed manner. It also offers a scalable solution for managing data flows with guaranteed delivery, data buffering / pressure release, and prioritized queuing.

NiFi helps you to build a cloud ingestion data flow in a quick and user-friendly way, by simply dragging and dropping a series of processors on the NiFi user interface. When the data flow is ready, you can visually monitor and control the pipeline you have built.

This use case walks you through the steps associated with creating an ingest-focused data flow from Apache Kafka into Amazon S3 object stores. If you are moving data from a location other than Kafka, see the *Getting Started with Apache NiFi* for information about how to build a data flow, and about other data get and consume processor options.

Related Information

[Getting Started with Apache NiFi](#)

[Ingesting Data into Apache Kafka](#)

[Ingesting Data into Apache HBase in CDP Public Cloud](#)

[Ingesting data into Apache Hive in CDP Public Cloud](#)

[Ingesting Data into Apache Kudu in CDP Public Cloud](#)

[Ingesting Data into Azure Data Lake Storage](#)

Meet the prerequisites

Use this checklist to make sure that you meet all the requirements before you start building your data flow.

- You have a CDP Public Cloud environment.
- You have a CDP username and password set to access Data Hub clusters. The predefined resource role of this user is at least EnvironmentUser. This resource role provides the ability to view Data Hub clusters and set the FreeIPA password for the environment.
- Your user is synchronized to the CDP Public Cloud environment.
- You have a Flow Management Data Hub cluster running in your CDP Public Cloud environment.
- You have a Streams Messaging Data Hub cluster in the same CDP environment as the Flow Management cluster.

- Your CDP user has been added to the appropriate pre-defined Ranger access policies to allow access to the NiFi UI.
- You have set up a consumer group in Ranger, and you have a Ranger policy for Kafka allowing your CDP user to access this consumer group.
- You have a Ranger policy on your Kafka topic you will use as a data source allowing the CDP user to consume from the topic.
- You have created a target folder on your AWS Management Console for the data to be moved to the cloud.
- You have created an AWS role with policies attached allowing write access to the S3 bucket you want to use in your data flow.

Related Information

[Understanding roles and resource roles](#)

[Creating your first Flow Management cluster](#)

[Configuring Ranger policies for Apache NiFi](#)

[IDBroker](#)

[Amazon S3 Console User Guide](#)

Build the data flow

Learn how you can create an ingest data flow to move data from Kafka to S3 buckets. This involves opening Apache NiFi in your Flow Management cluster, adding processors and other data flow objects to your canvas, and connecting your data flow elements.

About this task

You can use the `PutHDFS` or `PutS3Object` processors to build your AWS ingest data flows. Regardless of the type of flow you are building, the first steps in building your data flow are generally the same. Open NiFi, add your processors to the canvas, and connect the processors to create the flow.

Procedure

1. Open NiFi in CDP Public Cloud.
 - a) To access the NiFi service in your Flow Management Data Hub cluster, navigate to Management Console service Data Hub Clusters .
 - b) Click the tile representing the Flow Management Data Hub cluster you want to work with.

- c) Click Nifi in the Services section of the cluster overview page to access the NiFi UI.

The screenshot displays the Cloudera Management Console interface for a cluster named 'cfm-nifi-cluster'. The left sidebar shows navigation options like Dashboard, Environments, Data Lakes, User Management, and Data Hub Clusters. The main content area shows cluster details: STATUS is 'Running', NODES is 4, and CREATED AT is 09/10/21, 02:24 PM GMT+2. The 'Services' section includes 'CM UI', 'NiFi' (highlighted with a red box), and 'NiFi Registry'. Below, the 'Management' table lists the cluster's ID, FQDN, and status. The 'NiFi' table shows the NiFi service's ID, FQDN, and status.

You will be logged into NiFi automatically with your CDP credentials.

2. Add the `ConsumeKafkaRecord_2_0` processor for data input in your data flow.
 - a) Drag and drop the processor icon into the canvas.

This displays a dialog that allows you to choose the processor you want to add.
 - b) Select the `ConsumeKafkaRecord_2_0` processor from the list.
 - c) Click Add or double-click the required processor type to add it to the canvas.



Note: This processor consumes messages from Kafka specifically built against the Kafka 2.0 Consumer API.

3. Add the `MergeRecord` processor.



Note: This processor merges together multiple record-oriented flowfiles into a single flowfile that contains all of the records of the input flowfiles. It works by creating bins and then adding flowfiles to these bins until they are full. Once a bin is full, all of the flowfiles are combined into a single output flowfile, and that flowfile is routed to the **merged** relationship.

When using the `ConsumeKafkaRecord_2_0` processor, you are pulling small-sized records, so it is practical to merge them into larger files before writing them to S3.

4. Add a processor for writing data to S3.

You have two options:

- `PutHDFS` processor: The HDFS client writes to S3 through the S3A API. This solution leverages centralized CDP security. You can use the usernames and passwords you set up in CDP for central authentication, and all requests go through IDBroker.
 - `PutS3Object` processor: This is an S3-specific processor that interacts with the Amazon S3 object store while still leveraging the authentication mechanism provided through IDBroker in CDP Public Cloud.
- a) Drag and drop the processor icon into the canvas.

In the dialog box you can choose which processor you want to add.
 - b) Select the processor of your choice from the list.
 - c) Click Add or double-click the required processor type to add it to the canvas.

5. Connect the processors to create the data flow by clicking the connection icon in the first processor, and dragging and dropping it on the second processor.

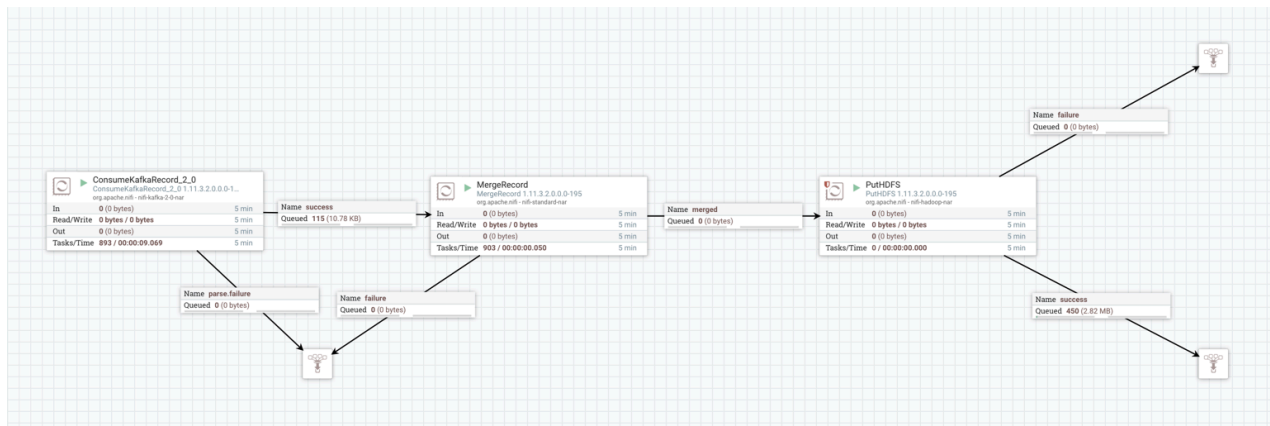
A Create Connection dialog appears with two tabs: Details and Settings. You can configure the connection's name, flowfile expiration time period, thresholds for back pressure, load balance strategy and prioritization.

- a) Connect `ConsumeKafkaRecord_2_0` with `MergeRecord`.
 - b) Add the **success** flow of the `ConsumeKafkaRecord_2_0` processor to the `MergeRecord` processor.
 - c) Click Add to close the dialog box and add the connection to your data flow.
 - d) Connect `MergeRecord` with your target data processor (`PutHDFS` / `PutS3Object`).
 - e) Add the **merged** flow of the `MergeRecord` processor to the target data processor.
 - f) Click Add to close the dialog box and add the connection to your data flow.
6. Optionally, you can add funnels to your flow.
 - a) `ConsumeKafkaRecord_2_0`: If any of the Kafka messages are pulled but cannot be parsed, the contents of the message will be written to a separate flowfile and that flowfile will be transferred to the `parse.failure` relationship. You can connect the failure queue coming from the processor to a failure funnel for parse failures.
 - b) `MergeRecord`: For merge failures, you can connect the failure queue coming from the processor to the failure funnel you used for `ConsumeKafkaRecord_2_0`.
 - c) `PutHDFS` / `PutS3Object`: You can add success and failure funnels at the end of the data flow and connect them with your target data processor. These funnels help you see where flow files are being routed when your flow is running.

If you want to know more about working with funnels, see the *Apache NiFi User Guide*.

Results

This example data flow has been created using the `PutHDFS` processor.



What to do next

Configure IDBroker mapping authorization.

Related Information

[Apache NiFi Documentation](#)

Set up AWS for your ingest data flow

Learn what you need to set on the AWS side to prepare for data ingestion.

Procedure

1. Create an S3 bucket that will be the target folder for ingesting data.
For instructions, see *Create your first S3 bucket*.
2. Create a policy to allow writing into your bucket.
For instructions, see *Creating IAM policies*.
3. Create an AWS IAM role.
You will use this AWS identity to delegate access to your S3 bucket.
4. Apply the policy you have created to grant access to your S3 bucket to the newly create IAM role.
5. Create a trust relationship between the AWS IAM role and the CDP IDBroker role that you create when you registered your environment to allow IDBroker to assume the AWS role.
For more information, see *Onboarding CDP users and groups for cloud storage (no RAZ)*.

What to do next

You are ready with preparing your AWS environment. Switch back to CDP and create an IDBroker mapping.

Related Information

[Create your first S3 bucket](#)

[Creating IAM policies](#)

[Onboarding CDP users and groups for cloud storage \(no RAZ\)](#)

Create IDBroker mapping

Learn how you can create the IDBroker mapping for the target data processors (`PutS3Object` and `PutHDFS`) of your data flow. To enable your CDP user to utilize the central authentication features CDP provides and to exchange credentials for AWS access tokens, you have to map your CDP user to the correct IAM role.

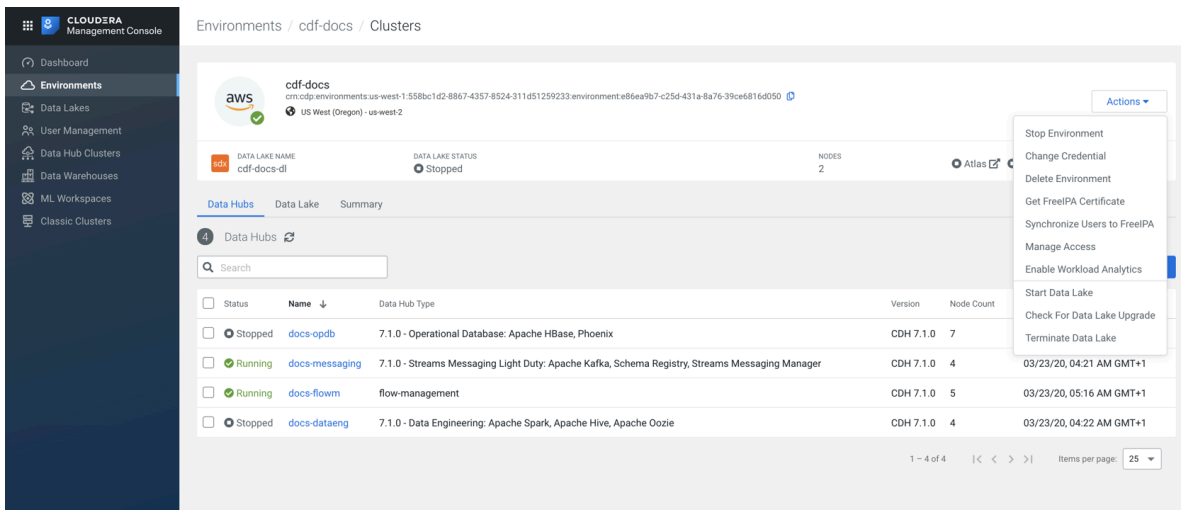
About this task

The option to add / modify the mappings is available from the Management Console in your CDP environment.

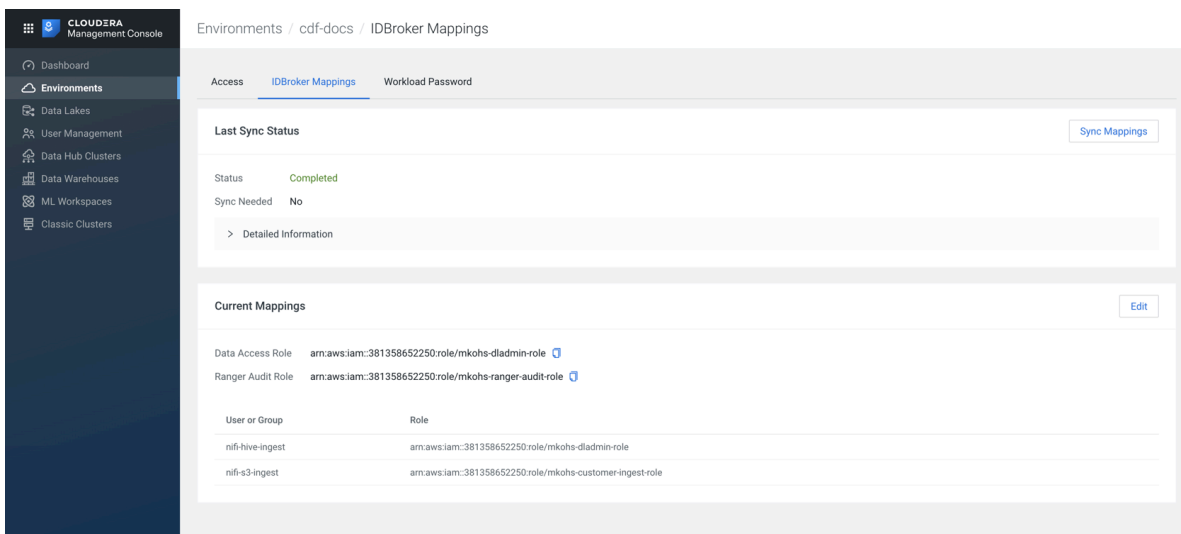
Procedure

1. Access IDBroker mappings in the Management Console.

a) To access IDBroker mappings in your environment, click **Actions Manage Access** .



b) Choose the IDBroker Mappings tab where you can provide mappings for users or groups and click **Edit**.



2. Add your CDP user account, a service account, or a user group and the corresponding AWS role that provides write access to your folder in your S3 bucket to the Current Mappings section by clicking the blue + sign.



Note: You can get the AWS IAM role Amazon Resource Name (ARN) from the Roles Summary page in AWS and can copy it into the IDBroker role field. The selected AWS IAM role must have a trust policy allowing IDBroker to assume this role.

3. Click Save and Sync.

What to do next

Configure controller services for your data flow.

Related Information

[IDBroker](#)

[Onboarding CDP users and groups for cloud storage](#)

[Create a provisioning credential for AWS](#)

Create controller services for your data flow

Learn how you can create and configure controller services for an S3 ingest data flow in CDP Public Cloud.

Controller services provide shared services that can be used by the processors in your data flow. You will use these Controller Services later when you configure your processors.

Procedure

1. To add a Controller Service to your flow, right-click on the canvas and select Configure from the pop-up menu. This displays the Controller Services Configuration window.
2. Select the Controller Services tab.

- Click the + button to display the Add Controller Service dialog.



- Select the required Controller Service and click Add.

Add Controller Service

Source: all groups (dropdown) | Displaying 5 of 67 | avro (dropdown)

Type	Version	Tags
AvroReader	1.9.2	comma, reader, record, values, ...
AvroRecordSetWriter	1.9.2	result, set, record, serializer, rec...
AvroSchemaRegistry	1.9.2	schema, registry, csv, json, avro
ConfluentSchemaRegistry	1.9.2	schema, registry, confluent, kaf...
HortonworksSchemaRegistry	1.9.2	schema, registry, hortonworks, ...

AvroReader 1.9.2 org.apache.nifi - nifi-record-serialization-services-nar

Parses Avro data and returns each Avro record as an separate Record object. The Avro data may contain the schema itself, or the schema can be externalized and accessed by one of the methods offered by the 'Schema Access Strategy' property.

CANCEL
ADD

- Perform any necessary Controller Service configuration tasks by clicking the Configure icon in the right-hand column.



- When you have finished configuring the options you need, save the changes by clicking the Apply button.

7. Enable the Controller Service by clicking the Enable button (flash) in the far-right column of the Controller Services tab.

Example

In this example the following controller services are used:

- AvroReader Controller Service
- CSVReader Controller Service
- CSVRecordSetWriter Controller Service
- AWSIDBrokerCloudCredentialsProvider Controller Service

You will use the record reader and record writer controller services when configuring the data source and merge record processors in your data flow.

You will use the AWS controller service when configuring the PutS3Object data target processor.

AvroReader Controller Service

This controller service parses Avro data and returns each Avro record as a separate record object.

Table 1: AvroReader Controller Service properties

Property	Description	Example value for ingest data flow
Schema Access Strategy	Specify how to obtain the schema to be used for interpreting the data.	HWX Content-Encoded Schema Reference
Schema Registry	Specify the Controller Service to use for the Schema Registry.	CDPSchemaRegistry
Schema Name	Specify the name of the schema to look up in the Schema Registry property.	customer

CSVReader Controller Service

This controller service parses your CSV-formatted data, returning each row in the CSV file as a separate record.

Table 2: CSVReader Controller Service Properties

Property	Description	Example value for ingest data flow
Schema Access Strategy	Specify how to obtain the schema to be used for interpreting the data.	Use String Fields From Header
Treat First Line as Header	Specify whether the first line of CSV should be considered a header or a record.	true

CSVRecordSetWriter Controller Service

This controller service writes the contents of a record set as CSV data.


Table 3: CSVRecordSetWriter Controller Service Properties

Property	Description	Example value for ingest data flow
Schema Write Strategy	Specify how the schema for a record should be added to the data.	Do Not Write Schema
Schema Access Strategy	Specify how to obtain the schema to be used for interpreting the data.	Use 'Schema Name' Property
Schema Name	Specify the name of the schema to look up in the Schema Registry property.	customer

AWSIDBrokerCloudCredentialsProvider Controller Service

This controller service defines credentials for the PutS3Object processor.

Table 4: CSVRecordSetWriter Controller Service Properties

Property	Description	Example value for ingest data flow
Configuration Resources	<p>Provide the path to the core site configuration file that contains IDBroker-related configuration to. The path is set as default. The only thing you have to configure is username and password.</p> <p> Note: The core-site.xml file is present on every Flow Management cluster. Cloudera Manager stores the right core-site.xml file in the same /etc directory for every cluster.</p>	/etc/hadoop/conf.cloudera.core_settings/core-site.xml
Username	Specify your username.	csso_jsmith
Password	<p>Specify your password.</p> <p>If you use your own username, provide the CDP workload password associated with your username.</p>	CDP workload password

What to do next

Configure the processors in your data flow.

Related Information

[Adding Controller Services for data flows](#)

[Apache NiFi Documentation](#)

Configure the processor for your data source

Learn how you can configure the `ConsumeKafkaRecord_2_0` data source processor for your S3 ingest data flow. You can set up a data flow to move data to Amazon S3 from many different locations. This example assumes that you are streaming data from Kafka and shows you the configuration for the relevant data source processor.

Procedure

1. Launch the Configure Processor window, by right clicking the `ConsumeKafkaRecord_2_0` processor and selecting Configure.

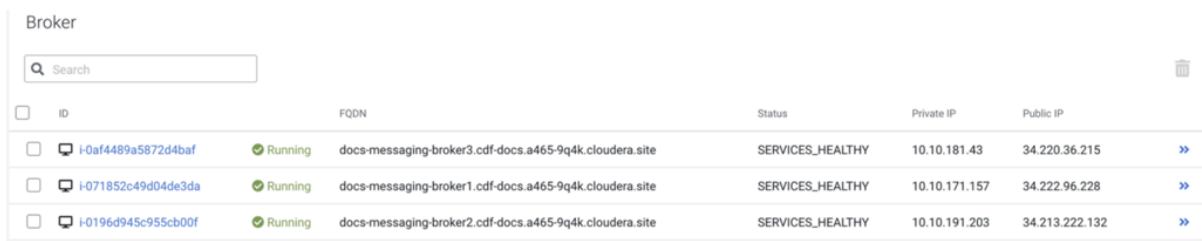
This gives you a configuration dialog with the following tabs: Settings, Scheduling, Properties, Comments.

2. Configure the processor according to the behavior you expect in your data flow.

- When you have finished configuring the options you need, save the changes by clicking the Apply button.

Make sure that you set all required properties, as you cannot start the processor until all mandatory properties have been configured.

In this example data flow, the data source is Kafka. You can create the modified Kafka broker URLs using the broker hostnames and adding port :9093 to the end of each FQDN. You can find the hostnames on the Streams Messaging cluster overview page when selecting the Hardware tab.



ID	FQDN	Status	Private IP	Public IP
f0af4489a5872d4baf	docs-messaging-broker3.cdf-docs.a465-9q4k.cloudera.site	SERVICES_HEALTHY	10.10.181.43	34.220.36.215
f071852c49d04de3da	docs-messaging-broker1.cdf-docs.a465-9q4k.cloudera.site	SERVICES_HEALTHY	10.10.171.157	34.222.96.228
f0196d945c955cb00f	docs-messaging-broker2.cdf-docs.a465-9q4k.cloudera.site	SERVICES_HEALTHY	10.10.191.203	34.213.222.132



Note: Property values can be parameterized. For example, you can create a parameter context to hold Kafka connection information and apply it to the Kafka Brokers property instead of adding the broker URLs individually.

The following table includes a description and example values for the properties required to configure the example ingest data flow. For a complete list of `ConsumeKafkaRecord_2_0` properties, see the *Apache Nifi Documentation*.

Table 5: ConsumeKafkaRecord_2_0 processor properties

Property	Description	Example value for ingest data flow
Kafka Brokers	Provide a comma-separated list of known Kafka Brokers. In the format: <host>:<port>	Docs-messaging-broker1.cdf-docs.a465-9q4k.cloudera.site:9093, docs-messaging-broker2.cdf-docs.a465-9q4k.cloudera.site:9093, docs-messaging-broker3.cdf-docs.a465-9q4k.cloudera.site:9093
Topic Name(s)	Provide the name of the Kafka Topic(s) to pull from.	Customer
Record Reader	Specify the Record Reader to use for incoming FlowFiles.	ReadCustomerAvroRoot
Record Writer	Specify the Record Writer to use in order to serialize the data before sending to Kafka.	CSVRecordSetWriter
Security Protocol	Specify the protocol used to communicate with Kafka brokers.	SASL_SSL
SASL Mechanism	Specify the SASL mechanism to use for authentication.	PLAIN
Username	Use your CDP workload username to set this Authentication property.	srv_nifi-kafka-ingest
Password	Use your CDP workload password to set this Authentication property.	password
SSL Context Service	Specify the SSL Context Service to use for communicating with Kafka. Use the pre-configured SSLContextProvider.	Default NiFi SSL Context Service

Property	Description	Example value for ingest data flow
Group ID	Provide the consumer group ID to identify consumers that are within the same consumer group.	nifi-s3-ingest



Note: If you want to move data to Amazon S3 from a location other than Kafka, see the *Apache NiFi Getting Started* for information about other data ingest processor options.

What to do next

Configure the processor for merging your records.

Related Information

[Configuring a processor](#)

[Apache NiFi Documentation](#)

[Getting Started with Apache NiFi](#)

Configure the processor for merging records

Learn how you can configure the `MergeRecord` processor for your S3 ingest data flow. You can use it to merge together multiple record-oriented flow files into a large flow file that contains all records of your Kafka data input.

Procedure

1. Launch the Configure Processor window, by right clicking the `MergeRecord` processor and selecting Configure.

This gives you a configuration dialog with the following tabs: Settings, Scheduling, Properties, Comments.

2. Configure the processor according to the behavior you expect in your data flow.
3. When you have finished configuring the options you need, save the changes by clicking the Apply button.

Make sure that you set all required properties, as you cannot start the processor until all mandatory properties have been configured.

In this example the following settings and properties are used:

Table 6: MergeRecord processor scheduling

Scheduling	Description	Example value for ingest data flow
Automatically Terminate Relationships		original

Table 7: MergeRecord processor properties

Property	Description	Example value for ingest data flow
RecordReader	Specify the Controller Service to use for reading incoming data.	ReadCustomerCSVRoot
RecordWriter	Specify the Controller Service to use for writing out the records.	CSVRecordSetWriter
Merge Strategy	Specify the algorithm used to merge records. The Bin-Packing Algorithm generates a FlowFile populated by arbitrarily chosen FlowFiles.	Bin-Packing Algorithm
Minimum Number of Records	Specify the minimum number of records to include in a bin.	900

Property	Description	Example value for ingest data flow
Maximum Number of Records	Specify the maximum number of Records to include in a bin.	1000

For a complete list of MergeRecord properties, see the *processor documentation*.

What to do next

Configure the processor for your data target.

Related Information

[Configuring a processor](#)

[Apache NiFi Documentation](#)

[Getting Started with Apache NiFi](#)

Configure the processor for your data target

Learn how you can configure the data target processor for your S3 ingest data flow. This example assumes that you are moving data to AWS S3 and shows you how to configure the corresponding processors.

Procedure

1. Launch the Configure Processor window by right clicking the processor you added for writing data to S3 (PutHDFS or PutS3Object) and selecting Configure.

This gives you a configuration dialog with the following tabs: Settings, Scheduling, Properties, Comments.

2. Configure the processor according to the behavior you expect in your data flow.

Make sure that you set all required properties, as you cannot start the processor until all mandatory properties have been configured.

Examples:

- You can use the following properties for PutS3Object:

Table 8: PutS3Object processor properties

Property	Description	Example value for ingest data flow
Object key		\${filename}
Bucket	Provide the name of your target bucket in AWS. This is not the bucket related to your data lake(s). You have to create it specifically for the data ingest.	ingest-idbroker-test


Property	Description	Example value for ingest data flow
AWS Credentials Provider Service	Provide here the controller service that is used to obtain AWS credentials provider.	AWSIDBrokerCloudCredentialsProviderControllerService

You can leave all other properties as default configurations.

For a complete list of `PutS3Object` properties, see the *processor documentation*.

- You can use the following properties for `PutHDFS`:

Table 9: PutHDFS processor properties

Property	Description	Example value for ingest data flow
Hadoop Configuration Resources	Specify the path to the core-site.xml configuration file. Make sure that the default file system (fs, default.FS) points to the S3 bucket you are writing to.  Note: The core-site.xml file used for the Hadoop Configuration Resources property is present on every Flow Management cluster. Cloudera Manager stores the right core-site.xml file in the same /etc directory for every cluster.	/etc/hadoop/conf.cloudera.core_settings/core-site.xml
Kerberos Principal	Specify the Kerberos principal (your username) to authenticate against CDP.	srv_nifi-s3-ingest
Kerberos Password	Provide the password that should be used for authenticating with Kerberos.	password
Directory	Provide the path to your target directory in AWS expressed in an S3A compatible path.	s3a://your path/customer

You can leave all other properties as default configurations.

For a complete list of `PutHDFS` properties, see the *processor documentation*.

- When you have finished configuring the options you need, save the changes by clicking Apply.

If you want to move data to a different location, review the other use cases in the *Cloudera Data Flow for Data Hub library*.

What to do next

Your data flow is ready to ingest data into AWS S3. Start the flow.

Related Information

[Configuring a processor](#)

[Apache NiFi Documentation](#)

[Data ingest use cases in Cloudera Data Flow for Data Hub](#)

Start the data flow

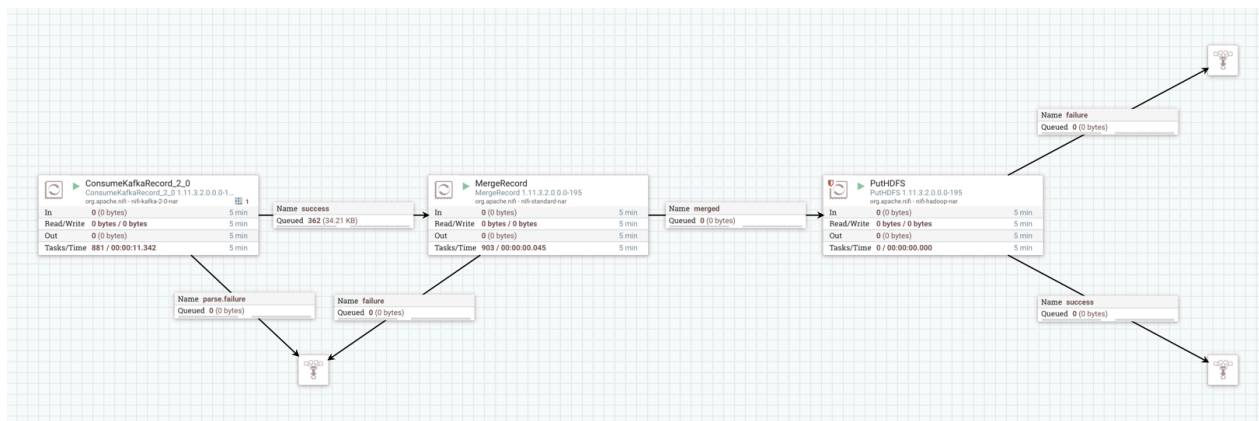
When your flow is ready, you can begin ingesting data into Amazon S3 buckets. Learn how to start your S3 ingest data flow.

Procedure

- To initiate your data flow, select all the data flow components you want to start.
- Click the Start icon in the Actions toolbar.

Alternatively, right-click a single component and choose Start from the context menu. Data should be read from Kafka and it should be written to the defined folder of your AWS S3 bucket

Results



What to do next

It is useful to check that data is running through the flow you have created.

Verify data flow operation

Learn how you can verify the operation of your S3 ingest data flow.

About this task

There are a number of ways you can check that data is running through the flow you have built.

Procedure

1. You can verify that NiFi processors are not producing errors.
2. You can look at the processors in the UI, where you can see the amount of data that has gone through them. You can also right click on the processors, or on connections to view status history.
3. You can check that the data generated appears in your AWS S3 bucket. To do this, return to the dedicated folder of your S3 bucket, where you should see your files listed.

You may have to refresh the page depending on your browser/settings.

Monitoring your data flow

Learn about the different monitoring options for your Amazon S3 ingest data flow in CDP Public Cloud.

You can monitor your data flow for information about health, status, and details about the operation of processors and connections. NiFi records and indexes data provenance information, so you can conduct troubleshooting in real time.

Data statistics are rolled up on a summary screen (the little table icon on the top right toolbar which lists all the processors). You can use the `MonitorActivity` processor to alert you, if for example you have not received any data in your flow for a specified amount of time.

If you are worried about data being queued up, you can check how much data is currently queued. Process groups also conveniently show the totals for any queues within them. This can often indicate if there is a bottleneck in your flow somewhere, and how far the data has got through that pipeline.

Another option to check that data has fully passed through your flow is to check out data provenance to see the full history of your data.

Next steps

Learn about the different options that you have after building a simple Amazon S3 ingest data flow in CDP Public Cloud.

Moving data to the cloud is one of the cornerstones of any cloud migration. Cloud environments offer numerous deployment options and services. This example data flow provides you with a model to design more complex data flows for moving and processing data as part of cloud migration efforts.

You can build a combination of on-premise and public cloud data storage. You can use this solution as a path to migrate your entire data to the cloud over time—eventually transitioning to a fully cloud-native solution or to extend your existing on-premise storage infrastructure, for example for a disaster recovery scenario. Cloud storage can provide secure, durable, and extremely low-cost options for data archiving and long-term backup for on-premise datasets.

You can also use cloud services without storing your data in the cloud. In this case you would continue to use your legacy on-premise data storage infrastructure, and work with on-demand, cloud-based services for data transformation, processing and analytics with the best performance, reliability and cost efficiency for your needs. This way you can manage demand peaks, provide higher processing power, and sophisticated tools without the need to permanently invest in computer hardware.