StreamSets

Reference Guide

Publication date 9th November 2020

All rights reserved.

Table of Contents

1. Introduction 4
    1.1. Compatibility 4
    1.2. Deployment 4
2. Architecture 5
3. Installation 6
    3.1. Pre-requisites 6
    3.2. Installation procedure 6
    3.3. Installing the Privitar data connector plug-in 6
    3.4. Installing the Token Vault driver 7
    3.5. Restart StreamSets 7
    3.6. Confirm that the Privitar data connector is available 7
4. Configuration 9
    4.1. Configuration procedure 9
    4.2. Create a Data Flow job in Privitar 10
    4.3. Record details about the Data Flow job 10
    4.4. Create a StreamSets Data Pipeline 10
    4.5. Configure the Origin and Destination Components 11
    4.6. Configure the Privitar Processor 12
    4.7. Usage and Setup for Data Flow UnMasking Jobs 13
5. Configuring users 14
    5.1. Creating an API user to run Data Flow jobs 14
    5.2. Masking Jobs 15
    5.3. UnMasking Jobs 16
6. Configuration options 18
    6.1. General 18
    6.2. Authentication 19
    6.3. Data Flow Job 19
    6.4. Advanced settings 20
7. Supported Data Types 21

1. Introduction

The StreamSets integration for the Privitar Data Privacy Platform can be used to apply a Data Flow Job consuming the records from a StreamSets Origin stage and streaming the results to a StreamSets Destination stage along a StreamSets pipeline.

The integration is supplied as a StreamSets Data Processor that you will be able to install in an existing StreamSets environment. You will be able to reuse the components you have already built for your StreamSets pipeline such as Origin and Destination stages.

1.1. Compatibility

The Privitar StreamSets data connector is compatible with StreamSets V3.16.0.

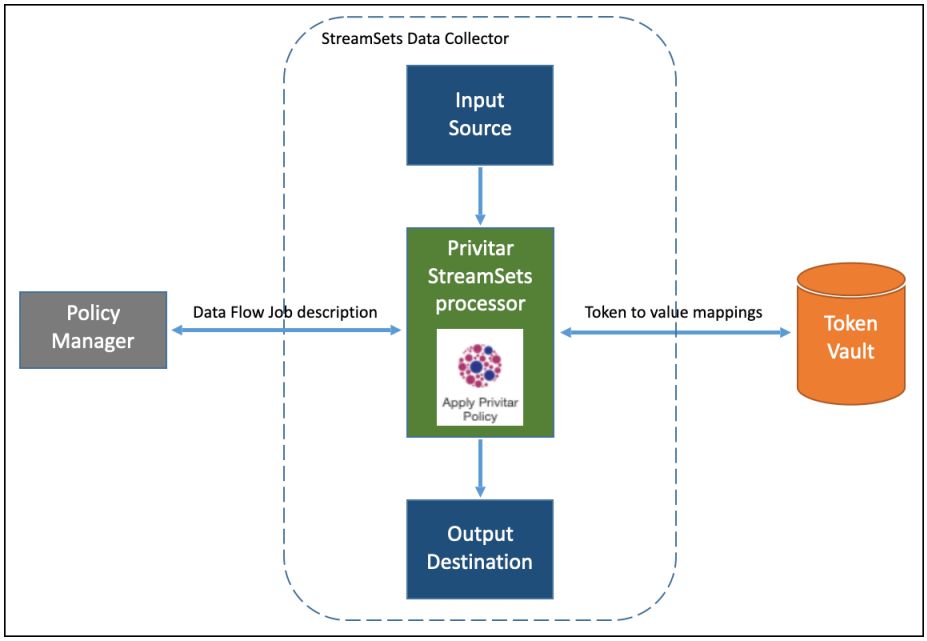
That is, the StreamSets Privitar Data Processor has been built and tested using this version of StreamSets.

1.2. Deployment

The StreamSets integration for the Privitar platform is supplied as a plug-in for StreamSets and can be deployed in an existing StreamSets environment.

2. Architecture

A simplified architectural diagram showing how the Privitar StreamSets Processor interfaces to the Privitar Policy Manager and Token Vault is shown below:



The process for de-identifying data in StreamSets using the Privitar StreamSets data processor (Apply Privitar Policy) is:

1. A data pipeline setup is initiated from the **StreamSets Data Collector**.
2. The data from the StreamSets **Input source** (Origin) is fed to the **Privitar StreamSets processor**.
3. The Privitar StreamSets processor contacts the **Privitar Policy Manager** for details about the Data Flow job that StreamSets has configured the processor to run on the input data.
4. The Privitar StreamSets processor applies the Data Flow Job details (Policy, PDD descriptions, Token Vault locations) on the input data to de-identify the data.
5. If the Data Flow Job uses a Policy that is configured to use Consistent Tokenization for certain rules, it will connect to the configured **Token Vault** to read/write tokens. (To improve performance, a local in-memory cache is used by the Privitar StreamSets processor.)
6. The **Processed data** is written out by the Privitar processor to a PDD in the Output destination (Destination).

The process for re-identifying (UnMasking) data in StreamSets using the Privitar StreamSets processor (**Apply Privitar Unmasking**) is the same. In this case the Input source would be a Privitar PDD.

3. Installation

This section describes the installation procedure for the Privitar StreamSets Data processor.

3.1. Pre-requisites

It is assumed that you have installed the following software:

- StreamSets v3.16.0
- Privitar Data Privacy Platform v3.8.0 (or later)

3.2. Installation procedure

1. Install the plug-in.
2. Install the necessary Token Vault drivers.
3. Restart StreamSets.
4. Confirm that the plug-in is available in StreamSets as a Processor.

3.3. Installing the Privitar data connector plug-in

The Privitar StreamSets data processor is provided as a tar file called:

`privitar-data-flow-streamsets-<x.x.x>.tar`

where <x.x.x> is the version of the Privitar platform. For example:

`privitar-data-flow-streamsets-3.8.0.tar`

To un-tar the file and install the plug-in (assuming you are using v3.8 of the Privitar platform):

1. Copy the tar file into the StreamSets directory that is defined by the StreamSets environment variable:

   `USER_LIBRARIES_DIR`

   (You can discover the definition of this variable using the env command from StreamSets.)
2. Un-tar the file, using the command:

   `tar -xvf privitar-data-flow-streamsets-3.8.0.tar`

   This command creates a directory called:

   `USER_LIBRARIES_DIR/privitar-data-flow-streamsets/`

   The Privitar StreamSets Data Processor jar file is located in:

   `privitar-data-flow-streamsets/lib/privitar-data-flow-streamsets-3.8.0.jar`

3.4. Installing the Token Vault driver

Drivers are required by the Privitar plug-in to connect to the Privitar Token vault. These drivers need to be added to the same location as the jar file for the Privitar StreamSets data processor. That is:

```
USER_LIBRARIES_DIR/privitar-data-flow-streamsets/lib/
```

The drivers to include are specific to the type of database you are using to store the Privitar Token Vault. For a StreamSets processing environment, the following types of Token Vault are supported:

- Relational Database (JDBC) including PostgreSQL (v9.6 and later) and Oracle (11g, 12c and later.)
- HBase v2.2.x and later

For JDBC drivers, you can use the drivers that are provided from PostgreSQL and Oracle vendors.

For HBase, Privitar provides a custom version of the HBase driver that can be used with the HBase Token Vault. The driver that needs to be used depends on the Hadoop Vendor that the HBase Token Vault is running on:

Hadoop vendor	Privitar HBase driver jar name
Google Bigtable	`privitar-hbase-bigtable-driver-[version].jar`
Cloudera CDH6 - HBase	`privitar-hbase-cdh6-driver-[version].jar`

For more information on accessing the Privitar HBase drivers, contact support@privitar.com.

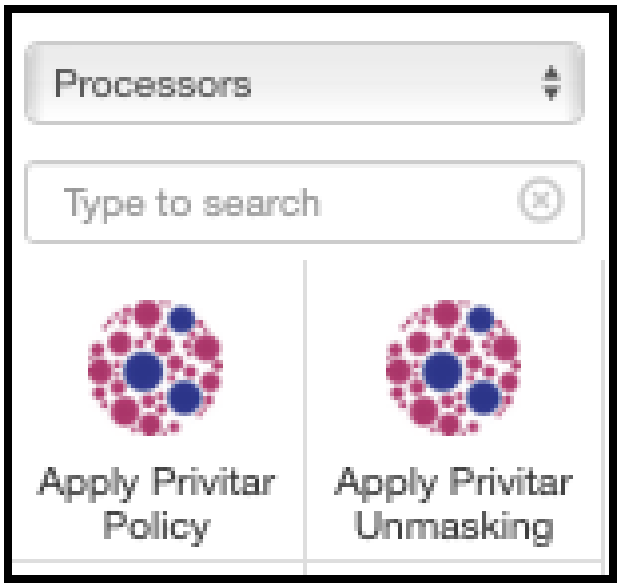
3.5. Restart StreamSets

To restart StreamSets:

1. Select the Administration icon in the top-right corner of the StreamSets main page.
2. Select **Restart** from the menu.

3.6. Confirm that the Privitar data connector is available

The Privitar StreamSets Data Processor should now be available from StreamSets in the Processors list box. For example:



Two processing components are available:

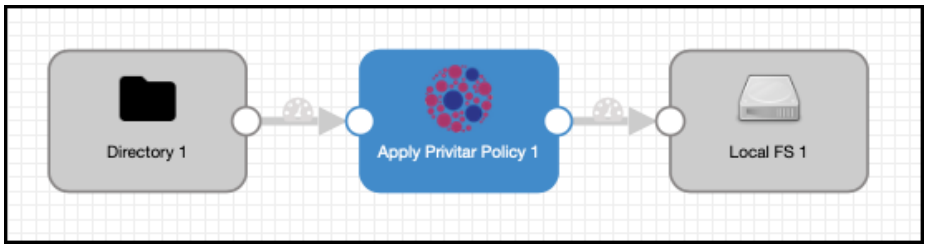
- **Apply Privitar Policy** - use this component to apply a Privitar Policy (de-identify data).
- **Apply Privitar Unmasking** - use this component to Unmask (re-identify data).

4. Configuration

This section describes how to use the Privitar StreamSets data processor in a StreamSets pipeline for **Data Flow Masking** Jobs.

A pipeline consists of stages that represent the origin and destination of the pipeline, and any additional processing that you want to perform. The Privitar StreamSets data processor can be included in between these two stages and is fully compatible with all the different types of components that can be used in each stage of the processing pipeline.

The example pipeline described in this section is a simple pipeline that is used for example only. In this example, data is processed from an input directory, de-identified by the Privitar processor and written out to an output directory. The setup is shown in StreamSets below:



To create this example pipeline from StreamSets, you need to create a Data Flow job in Privitar and then construct a pipeline in StreamSets that will use the Privitar processor to de-identify the data.

NOTE
In this example, StreamSets connects to Privitar using Basic authentication, which requires the creation of an API user that has the authorization rights to run Data Flow jobs. For more information about configuring users in Privitar to run Data Flow Jobs, refer to Configuring users [14].

4.1. Configuration procedure

To configure the Privitar processor, follow the procedure below:

1. Create a Data Flow job in Privitar.
2. Record details about the Data Flow job required by StreamSets.
3. Create a StreamSets Data Pipeline.
4. Configure the Origin and Destination components.
5. Configure the Privitar Processor.

4.2. Create a Data Flow job in Privitar

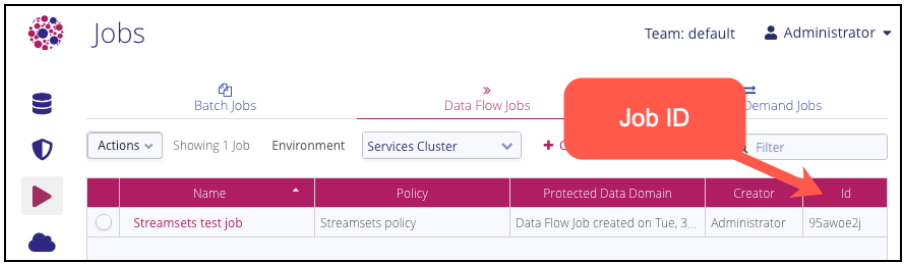
To create a Data Flow job in Privitar, refer to the *Privitar Data Privacy Platform User Guide* for general instructions about creating a Data Flow job. Creating a job involves creating a Schema, creating a Policy of rules that will be applied to the data in the Schema and finally defining a Data Flow job to process the data.

When creating the Schema, it is important to ensure that the StreamSets data types defined in the Schema are supported by the Privitar processor. The processor will map StreamSets data types to Privitar data types and this may cause errors at the processing stage if the defined Privitar Schema does not match the data types of the underlying input file. For more information about the mapping of data types between StreamSets and Privitar, refer to Supported Data Types.

4.3. Record details about the Data Flow job

You will need the following details from the Privitar platform when setting up the StreamSets pipeline:

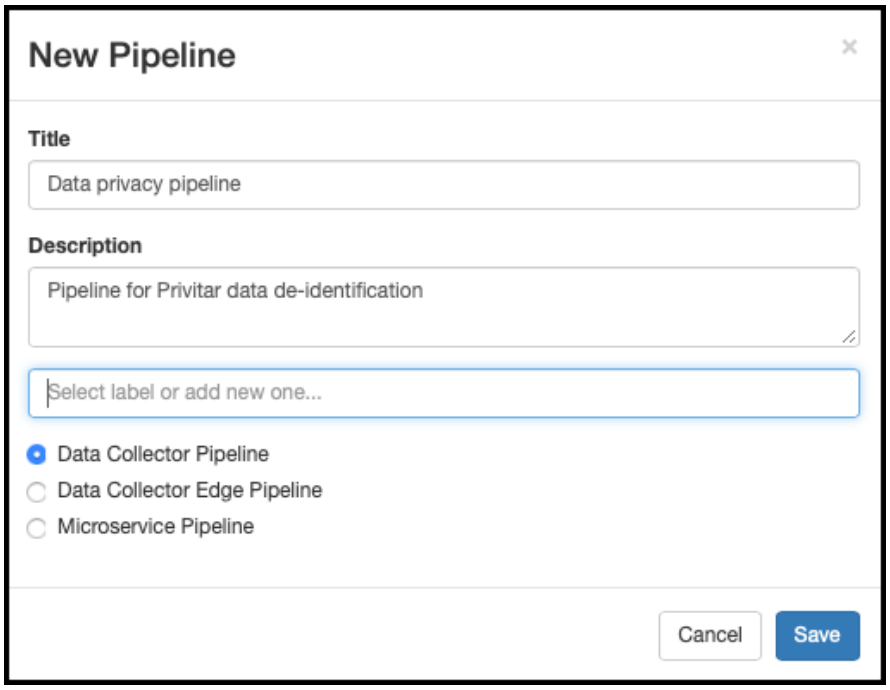
- The Job **ID** for the Data Flow job created in Privitar. See:



4.4. Create a StreamSets Data Pipeline

To create a StreamSets pipeline that uses the Privitar processor:

1. Create a a new pipeline by selecting, **Create New Pipeline** from the StreamSets **Pipelines** page.
2. Enter the details for the new pipeline in the **New Pipeline** dialog box.
3. Select **Data Collector Pipeline** as the pipeline type:



4. Select a data source (Origin) from the **Select Origin** drop-down list box. Choose, **Directory**.
5. Select a data processor to connect to from the **Select Processor to connect ...** list box. Choose, **Apply Privitar Policy** from the drop-down list box.
6. Select a data target (Destination) from the **Select Destination to connect ...** list box. Choose, **Local FS**.

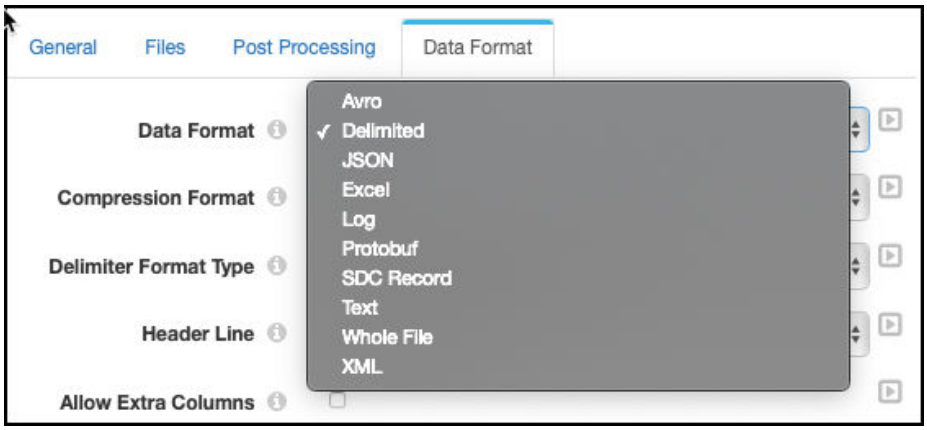
4.5. Configure the Origin and Destination Components

For information on how to configure the Origin and Destination components, refer to the StreamSets documentation.

Note that when configuring the Origin component you need to check that the data format of the files to be processed are supported:

- The Privitar StreamSets data processor has been specifically tested with **JSON**, **Avro** and **Delimited** file formats, but can process many other file formats.
- Refer to the StreamSets documentation (Data Formats Overview) for more information about supported file formats.

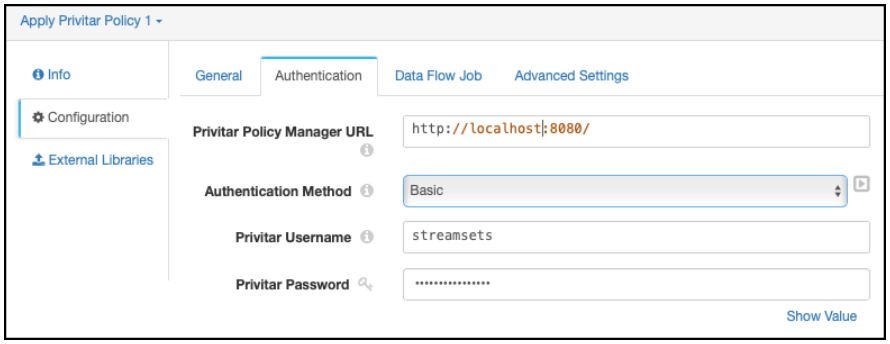
The file format used in the Origin data source is defined in the **Data Format** tab:



4.6. Configure the Privitar Processor

To configure the Privitar processor, you need to edit the **Authentication** tab and the **Data Flow** tab with details about the Data Flow job you have setup on the Privitar platform, together with connection details to use to connect from StreamSets to the Privitar platform.

1. Edit the **Authentication** tab to define how StreamSets connects to Privitar:



2. Enter the http address and port number for the Privitar platform. For example:

   `http://localhost:8080`.

   If you are using Basic Authentication, the address and port number must be of the format:

   `http://<address>:8080`

   If you are using Mutual TLS authentication, the address and port number must be of the format:

   `https://<address>:8443`

3. Select the authentication method from the **Authentication Method** drop-down list box, depending on the authentication method used to connect to the Privitar platform:

   - For **Basic Authentication** and **Mutual TLS Authentication**, enter the **Privitar username** and **Privitar password** details for an existing API user on the Privitar platform that has a Role with a **Run Data Flow** permission in the Team that the Data Flow Job is defined in.
   - Additionally, for **Mutual TLS authentication**, enter the necessary certificate and password information.

4. In the **Data Flow Job** tab, enter the **Job ID** of the Data Flow job that you want to run in this pipeline.

   It's worth noting that you can only use one Job ID per StreamSets processor. It is not possible to run multiple Job IDs in a processor. The Job ID you specify when setting up the processor is the only job that can be run by that processor. If you want the processor to run a different job, you need to enter a new Job ID in this tab, or create a new data pipeline.
5. Select **Start** to run the pipeline. If there are no errors, the activity of the data processing pipeline will be displayed in the **Summary** window showing the data records being processed.

For more information about all the configuration options that are available for the Privitar processor, refer to Configuration Options.

4.7. Usage and Setup for Data Flow UnMasking Jobs

The usage and setup of the Privitar data processor in a StreamSets pipeline for Data Flow UnMasking Jobs is very similar to the requirements for a Data Flow Masking Job. Here are some differences to be aware of:

- Need to ensure that if you have created an API User for connecting to Privitar from Streamsets that this user also has rights to **Run Data Flow** for **Unmasking jobs**. In the **default** Team in Privitar this permission is not enabled. For more information, see Configuring users [14].

5. Configuring users

To run Data Flow jobs in Privitar you need to create API users with the correct permissions to run both Masking and UnMasking Jobs in the Team that the Data Flow Job is defined in. This section describes how to create and configure API users on Privitar to run Data Flow Jobs. It is applicable for Data Flow jobs being set up on any of the following data processing platforms:

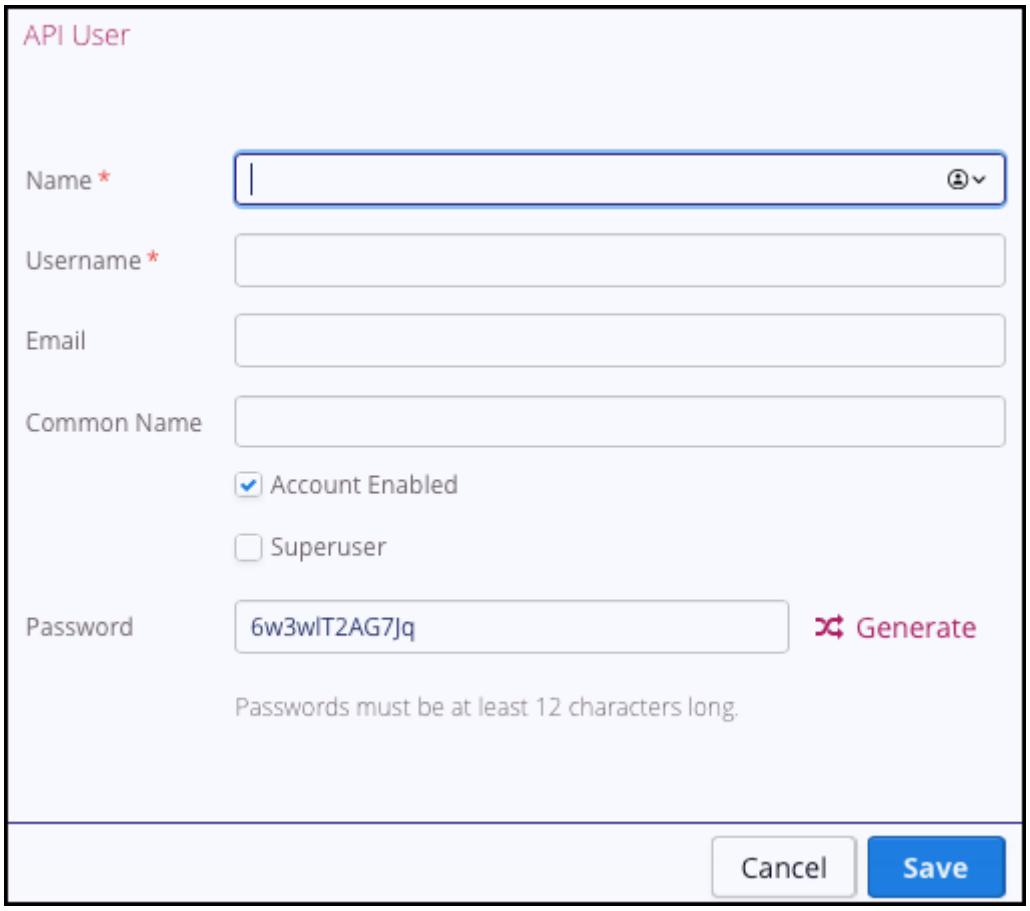
- Kafka/Confluent
- Apache Nifi
- StreamSets

For more general information about managing Users, Roles and Teams in Privitar, refer to the *Privitar Data Privacy Platform User Guide*.

5.1. Creating an API user to run Data Flow jobs

To create API users in Privitar for running Data Flow jobs:

1. Select **API Users** from the Superuser navigation panel.
2. Select **Create New API User**. The **API User** dialog box is displayed:



Enter the details for the new API user. The first two fields - **Name** and **Username** - are mandatory. All other fields are optional:

- **Name** is the display name of the API user.
- **Username** is the unique username for the API user.
- **Email** is the email address associated with the API user. This is an optional field.
- **Common Name** is used for API authentication (if your Privitar installation is configured to use Mutual TLS) or **Password** (if basic HTTP authentication is used).

  You can click on **Generate** to generate a new password.
- To make sure the User account is activated, select the **Account Enabled** check box.
- Optionally, if you want this new API User to have Superuser permissions, select the **Superuser** check box.

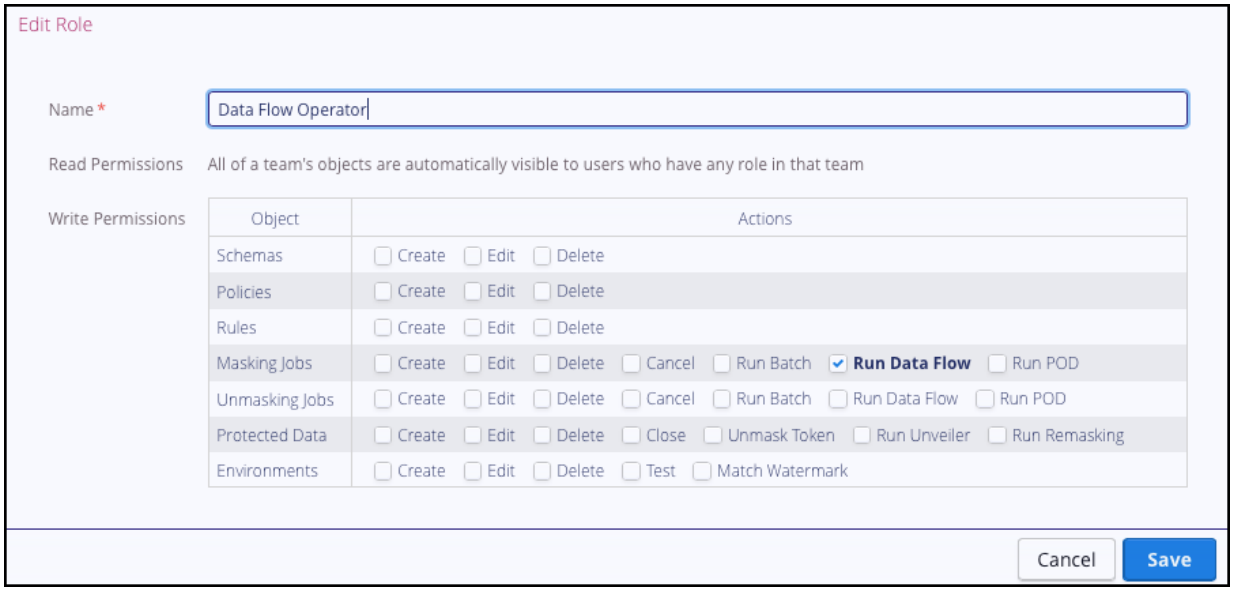
3. Click **Save** to save the details entered and to create the API user. The new API user will be added to the list of API users shown in the main window.

Typically, you would create two API users; one to run Masking jobs and the other to run UnMasking jobs. It is also possible for a single API user to run both jobs if required.

NOTE
It is also possible in Privitar to manage users externally in LDAP. If managing users in this way, then instead of assigning individual API users to a team role, you need to assign an LDAP group to the relevant team role.

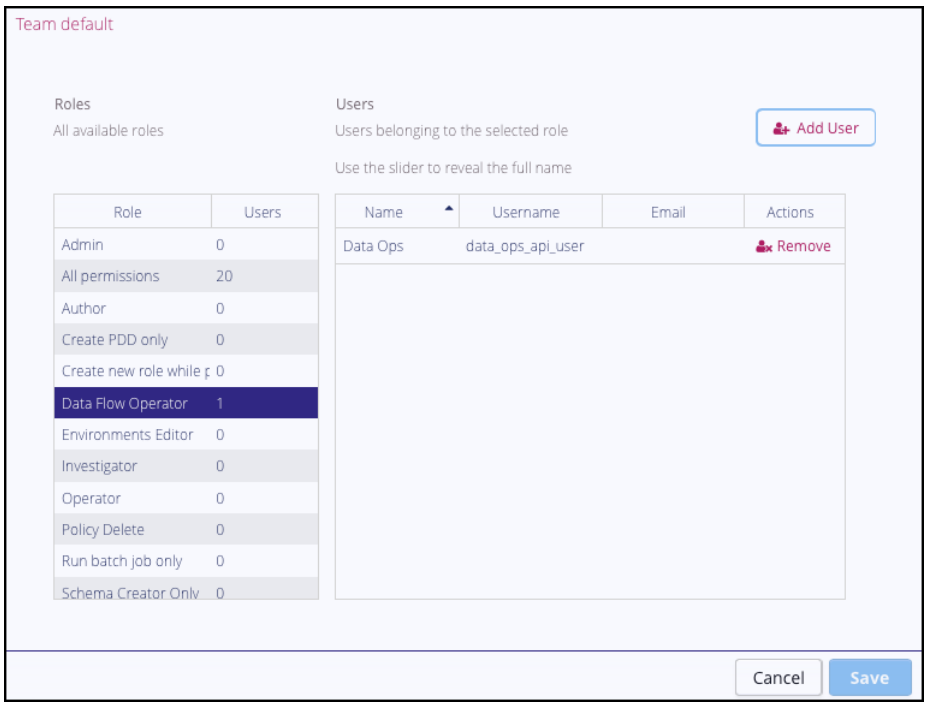
5.2. Masking Jobs

By default, the **Data Flow Operator** Role in the **default** Team in Privitar has the **Run Data Flow** permission enabled for **Masking Jobs**. See:



The API user that has been created in Privitar will need to be assigned the role of **Data Flow Operator** in the Team that the job is defined in.

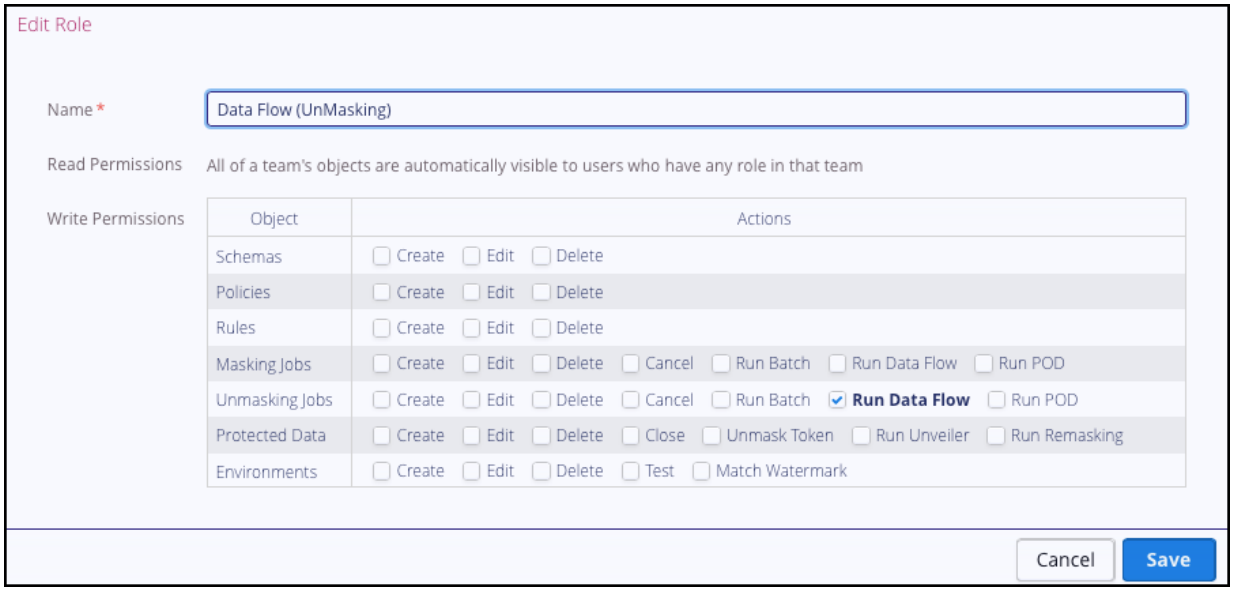
In the example below, an API user called **data_ops_api_user** has been created and assigned the role of **Data Flow Operator**:



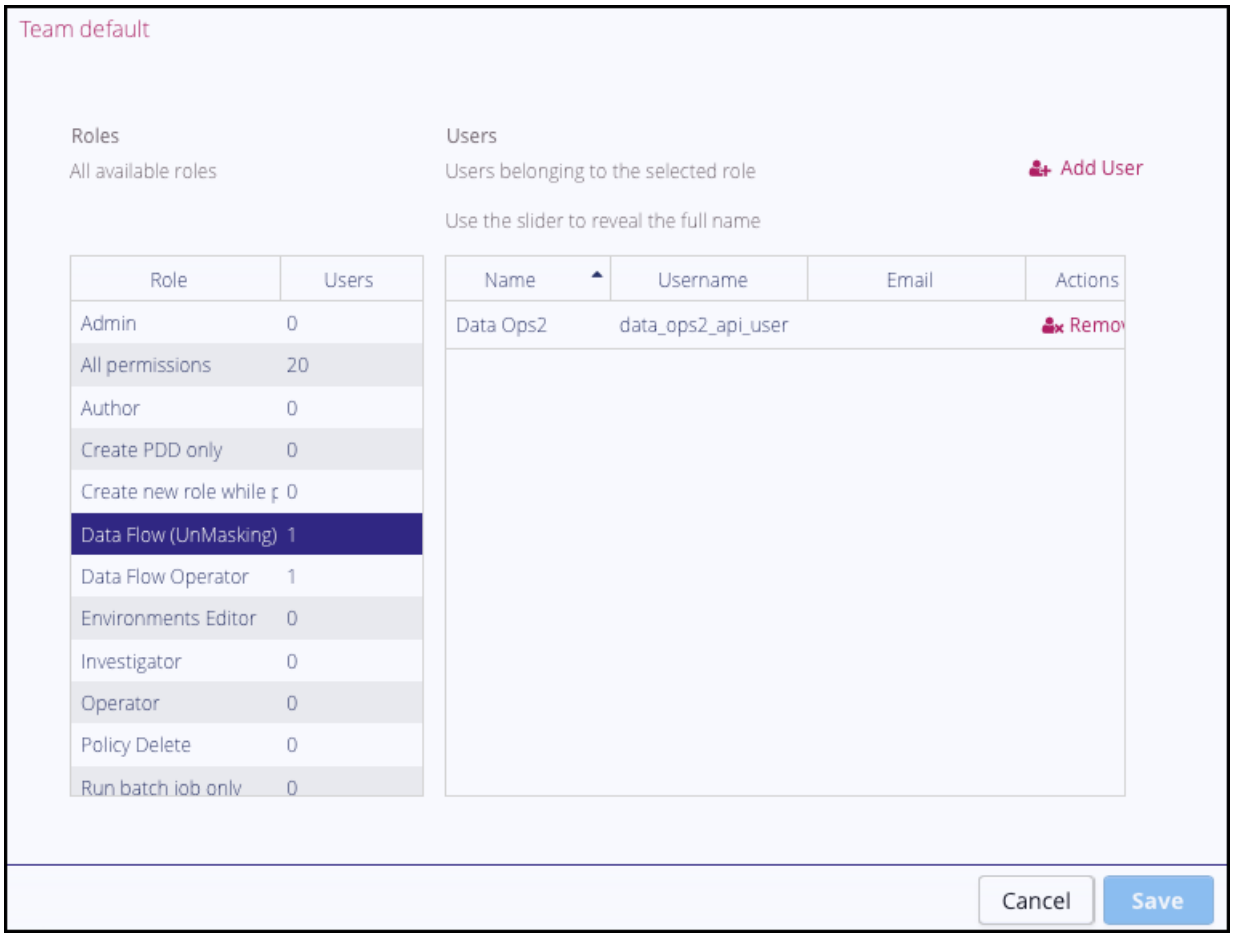
5.3. UnMasking Jobs

For UnMasking jobs, you need to assign an API user to a Role that has permission to Run Data Flow UnMasking jobs in the Team that the job is defined in.

In the example below, a new Role has been created called, **Data Flow (Unmasking)** with the **Run Data Flow** permission enabled for **Unmasking jobs**:



The new additional API user (**data_ops2_api_user**) can be assigned to the **Data Flow (Unmasking)** role:


Team default
Roles
All available roles
Users
Users belonging to the selected role
Add User
Use the slider to reveal the full name
Role
Users
Admin 0
All permissions 20
Author 0
Create PDD only 0
Create new role while p 0
Data Flow (UnMasking) 1
Data Flow Operator 1
Environments Editor 0
Investigator 0
Operator 0
Policy Delete 0
Run batch job only 0
Name
Username
Email
Actions
Data Ops2
data_ops2_api_user
Remov
Cancel
Save

6. Configuration options

This section describes the configuration options for the Privitar data processor. Many of the configuration options are set to sensible defaults, so If you are unsure about a particular setting, keep the default value.

Some things to note about configuring the processor:

- You can't change the configuration of the processor when the pipeline is running. The pipeline must be stopped to enable it to be configured.
- For convenience, some of the Configuration tabs contain an option to switch to **Bulk Edit Mode**. This enables you to enter the configuration options for that category in JSON format.

6.1. General

Attribute	Description	Default setting /Options available
Name	Name of the data processor.	Default setting (Apply Privitar Policy 1).
Description	A description of the use of the processor.	Default setting (Empty).
Required Fields	The fields in the Schema used by the Data Flow job that must contain data in order for the job to be processed.	Default setting (No fields are required.) To select fields from the Schema, choose 'Select Fields Using Preview Data' and select the fields from the list of fields that are displayed
Preconditions	Records that don't satisfy the specified preconditions are sent to error.	Default setting (No preconditions set.) If there are many preconditions to define, select 'Switch to bulk edit mode' to add multiple preconditions in a single entry. For more information on the types of preconditions that can be set, refer to the StreamSets Data Collector User Guide.
On Record Error	What action to take if a data processing error occurs.	Default setting (Send to Error.) Other options are: Discard or Stop Pipeline.

6.2. Authentication

Attribute	Description	Default setting /Options available

Privitar Policy Manager URL	The HTTP address and port number of the Policy Manager that is used to run the Data Flow job used by the data pipeline	If using basic authentication, this address would be:
		`http://<address>:8080/`
		For Mutual TLS authentication, this address would be:
		`https://<address>:8443/`
		where `<address>` is the IP address of the Policy Manager.

Authentication Method	The method used for authenticating with the Privitar Policy Manager.	
Basic Authentication		
Privitar username	Username of the API user.	Default setting (Empty) The API user must be have a Role with Run Data Flow permission for Masking Jobs or Unmasking jobs, in the Team that the job is defined in.
Privitar password	Password for the API user.	Default setting (Empty)
Mutual TLS Authentication		
TLC Client Certificate File Path (from local file system)	Specifies the location of the certificate file used for authenticating with the Privitar Policy Manager.	Default setting (Empty) The Common Name (CN) entry in the TLS certificates should resolve to an API user in Privitar. The API user must have a Role with Run Data Flow permission for Masking Jobs or Unmasking jobs, in the Team that the Job is defined in. For more information about creating API users in Privitar, see Configuring users [14]. For more information about Mutual TLS authentication, see Pipeline Configuration in the Streamsets documentation.
TLS Client Certificate Password	The password for the TLS client certificate file.	
TLS Trusted CA Certificate File Path (from local file system)	Specifies the location of the TLS CA certificate file used for authenticating with the Privitar Policy Manager.	

6.3. Data Flow Job

Attribute	Description	Default setting / Options available
Job ID	The Job ID of the Data Flow Job configured in the Policy Manager. This ID can be retrieved from the Data Flow Job details page in the Policy Manager UI).	Default setting (Empty).

6.4. Advanced settings

Attribute	Description	Default setting / Options available
Max Cache size	The maximum size (in bytes) of the local cache that is used to store tokens prior to being written to the Token Vault.	Default setting (512000000)
Max Batch size	Incoming records will be processed in batches no larger than this size.	Default setting (1000)
Concurrent Batches	The maximum number of batches that can be processed in parallel.	Default setting (20)
Job Cache Expiration (minutes)	The interval after which a Job cache entry that is not in use will be expired and closed.	Default setting (60)
Job Cache Refresh Frequency (minutes)	he frequency at which Job definitions are refreshed from the Policy Manager.	Default setting (10)
Token Vault Connection Cache Expiration (minutes)	The interval after which a Token Vault connection that is not in use will be expired and closed.	Default setting (30)
Token Vault Kerberos Keytab Path (from local file system)	Specifies the location of the Kerberos keytab used for connecting to an HBase token vault.	Default setting (Empty)
Advanced Settings	Advanced settings for debugging, tuning, monitoring, etc.	Default setting (Empty)

7. Supported Data Types

This section defines the mapping between data types supported by StreamSets and data types supported by Privitar.

For data types in your dataset that are not supported by Privitar, use the StreamSets Field Type Convertor. This component converts the data types of fields to compatible data types using a variety of different methods. For more information, see StreamSets Field Types.

The table below defines the mapping. StreamSets data types that are not supported by Privitar are passed-through as Object data types by Privitar. These specific mappings are highlighted in the table:

StreamSets Data Type	Privitar Data Type
BOOLEAN	BOOLEAN
CHAR	**OBJECT**
BYTE	BYTE
SHORT	SHORT
INTEGER	INTEGER
LONG	LONG
FLOAT	FLOAT
DOUBLE	DOUBLE
DATE	DATE
DATETIME	TIMESTAMP
TIME	**OBJECT**
DECIMAL	DECIMAL
STRING	TEXT
FILE_REF	**OBJECT**
BYTE_ARRAY	**OBJECT**
MAP	**OBJECT**
LIST	**OBJECT**
LIST_MAP	**OBJECT**
ZONED_DATETIME	**OBJECT**