

Connections & Variables

This guide aims to inform you, the user, on how to create and manage your Airflow Connections & Variables.

Please note the following

Airflow Connections & Variables cannot be restricted and are open to all users. If created properly, the secret values won't be visible but they can still be overwritten by any user.

The Hogwarts team is currently working on a better secret management system to replace this functionality. Stay tuned.

- [What are Connections & Variables?](#)
- [Connections](#)
- [Variables](#)

What are Connections & Variables?

Connections and Variables allow users to create predefined resources which may be accessed in their DAGs at runtime. These resources are stored in Airflow's metadata DB and can only be viewed/created by users with the required permission level.

Connections are typically used by Hooks and Sensors but can also be used to generate a resource URI. Variables however, are most commonly used as a medium for storing sensitive key/values which are then injected as environment variables or parameters for the task operator.

Connections

List Airflow Connections:

- 1 - Navigate to Airflow
- 2 - Under the navigation > Admin > Connections

SPBK board udev u pb

Airflow DAGs Security Browse Admin Docs

DAGs

Variables
Configurations
Connections
Plugins
Pools
XComs

Filter DAGs by tag

All 68 Active 36 Paused 32

DAG	Owner	Runs	Schedule	Last Run	Recent Task
airflow_email_test	dcmarti	30	0 12 ***	2021-01-06 12:00:00	1
aparamtest	demo	31	None	2021-04-27 14:07:08	1
atest	airflow	25	None	2021-04-16 16:08:29	1
catchup_test	debug	32	@daily	2021-05-10 00:00:00	2
cccs_yara_feeds_test	DASI2D	51	0 */1 ***	2021-02-24 02:00:00	2
censys_ipv4_daily_downloads	DASI2A	20	55 23 ***	2021-01-05 23:55:00	2
CIRA_Dzone_DNS_Feed	INO1A	238	*/5 ***	2021-03-11 14:16:48	2

3 - You should now see the following interface

SPBK board udev u pb

Airflow DAGs Security Browse Admin Docs

List Connection

Search

+ Actions

	Conn Id	Conn Type	Description	Host
<input type="checkbox"/>	airflow_db	mysql		mysql
<input type="checkbox"/>	aws_default	aws		
<input type="checkbox"/>	azure_container_instances_default	azure_container_instances		
<input type="checkbox"/>	azure_cosmos_default	azure_cosmos		
<input type="checkbox"/>	azure_data_lake_default	azure_data_lake		
<input type="checkbox"/>	beeline_default	beeline		localhost
<input type="checkbox"/>	bigquery_default	google_cloud_platform		
<input type="checkbox"/>	cassandra_default	cassandra		cassandra
<input type="checkbox"/>	custom_azure_data_lake	wasb		
<input type="checkbox"/>	databricks_default	databricks		localhost

Create an Airflow connection:

1. Under Admin > Connections, click



2. Enter the 'Conn Id' - make sure its unique but descriptive as this will be referenced in your DAG.
3. Enter the 'Conn type' that matches the Connection you want.
4. Enter the necessary connection information for your Connection type.

Add Connection

Conn Id *	<input type="text"/>
Conn Type *	<div><div>HTTP</div><div>▼</div></div> <p>Conn Type missing? Make sure you've installed the corresponding Airflow Provider Package.</p>
Description	<input type="text"/>
Host	<input type="text"/>
Schema	<input type="text"/>
Login	<input type="text"/>
Password	<input type="password"/>
Port	<input type="text"/>
Extra	<input type="text"/>

Edit/delete an Airflow connection:

1. Under Admin > Connections, find your connection in the list
2. Click the appropriate icon to edit or delete your connection
 - a. **Note:** Inputs such as password will need to be re-entered as they are encrypted on submit and cannot be changed like other inputs.

Using Connections in your DAG

The following code will extract the full URI, the login, and the password from the stored connection and pass these values as environment variables to the BashOperator.

```
1 from airflow.hooks.base_hook import BaseHook
2 ...
3 # Connection type for this test is "http"
4 connection = BaseHook.get_connection("<connection_id>")
5 t1 = BashOperator(
6     dag=dag,
```

```

7     task_id='<task_id>',
8     bash_command='env | grep CONNECTION',
9     env={
10         "CONNECTION_URI": connection.get_uri(),
11         "CONNECTION_LOGIN": connection.login,
12         "CONNECTION_PASSWORD": connection.password
13     }
14 )

```

Variables


List Airflow Variables:

- 1 - Navigate to Airflow
- 2 - Under the navigation > Admin > Variables

The screenshot shows the Airflow web interface at <https://airflow.hogwarts.udev.azure.chimera.cyber.gc.ca/home>. The 'Admin' menu is open, showing options like Variables, Configurations, Connections, Plugins, Pools, and XComs. The 'DAGs' section is active, displaying a list of DAGs with filters for All (68), Active (36), and Paused (32). A search bar for 'Filter DAGs by tag' is present. The DAGs table lists tasks such as 'airflow_email_test', 'aparamtest', 'atest', 'catchup_test', 'cccs_yara_feeds_test', 'censys_ipv4_daily_downloads', 'CIRA_Dzone_DNS_Feed', 'complex_dependency_graph', and 'dasi2 get together', each with its owner, status, runs, and schedule.

DAG	Owner	Runs	Schedule	Last
airflow_email_test	dcmarti	30	0.12***	202
aparamtest	demo	31	None	202
atest	airflow	25	None	202
catchup_test	debug	32	@daily	202
cccs_yara_feeds_test	DASI2D	51	0.7/1***	202
censys_ipv4_daily_downloads	DASI2A	20	55.23***	202
CIRA_Dzone_DNS_Feed	INO1A	238	7/5***	202
complex_dependency_graph	<insert_owner_email>	133	@daily	202
dasi2 get together	DASI2B	169	@daily	202

- 3 - You should now see the following interface

 Airflow

[DAGs](#)

[Security](#)

[Browse](#)

[Admin](#)

[Docs](#)

Browse... No file selected.

 Import Variables

List Variable

Search

+

Actions

←

Create an Airflow Variable :

1. Under Admin > Variables, click



2. Enter the 'Key' - make sure it's unique and it contains the word 'secret'. This is how Airflow knows to encrypt the value.
3. Enter your 'Value'.

Add Variable

Key *

Key

Val

Val

Save 


Edit/delete your Airflow Variable:

1. Under Admin > Variables, find your variable in the list
2. Click the appropriate icon to edit or delete your variable
 - a. **Note:** Inputs such as password will need to be re-entered as they are encrypted on submit and cannot be changed like other inputs.

Using Variables in your DAG

The following code will extract the Variable's value, decrypt it, and pass it as an environment variable to the BashOperator.

```
1 from airflow.models import Variable
2 ...
3 t1 = BashOperator(
4     dag=dag,
5     task_id='<task_id>',
6     bash_command='env | grep SECRET',
7     env={
8         "SECRET_VARIABLE": Variable.get("<variable_secret_id>")
9     }
10 )
```

 As always, the code for these examples can be found in the [Hogwarts examples repository](#).