Airflow FAQs

How is access provisioned?

Access is provisioned in two places, within Okta and the Hieroglyph repo.

Okta

Okta manages login access to the Airflow web app. All users able to log into the web app have the `Viewer` role.

1. Full-time employees (FTEs) in the `tech` group have login access.
2. Contractors in the `contractors` group have login access.

Hieroglyph

Hieroglyph manages access at the DAG level, i.e. who can enable and run DAGs within the web app.

After you are able to login, to interact with specific DAGs, access is provisioned in a few ways. For context, subdirectories are created in the Hieroglyph repo to represent workspaces.

1. If a subdirectory is the name of an existing product group, then all members of that product group have access by default.
2. An OWNERS.yaml file can be created to specify additional groups and LDAPs that should have access.

    a. OWNERS.yaml files can also be created in workspace subdirectories to produce tiered access. Users listed in the root will have access to all DAGs. Users listed in a nested subdirectory will only have access to DAGs contained in it.
        i. Please note these subdirectories eventually create roles used in the web server, and a role has a character limit of 64. So, if a nested path is greater than this limit, tiered access breaks and will fall back to a level that does not exceed the limit.

I attempted to log into Airflow, but the Okta authentication flow loops and is unsuccessful. How can I fix this?

The root cause of this issue and its solution has varied between users that reported it.

1. On a non-Linux machine, the issue was transient. Some users were able to cancel the Okta authentication process, then retry it successfully.
2. On a Linux machine, the issue was persistent. A user reported security keys configured in Okta are Okta domain-specific in a Linux environment. The Okta domain used by Airflow is `indeed.okta.com`. Another Okta domain used by other applications at Indeed is `id.indeed.tech`. Since the keys are domain-specific, the solution was to create a separate key for each domain.
    a. For Airflow, navigate to `indeed.okta.com`, then set up a security key in the security methods panel.
3. If the above solutions do not work, please reach out on #help-itso, then notify us on #help-airflow so we are aware of the issue.

Why doesn't my DAG appear in Airflow?

There are generally two reasons why a DAG doesn't appear in Airflow.

1. You created a TOML DAG, but the file extension is incorrect. For TOML DAGs, ensure the following format is used `<filename>.dag.toml`.
2. The DAG hasn't been processed yet. The Airflow processor is responsible for periodically checking DAGs in the repository. It runs every five minutes. Coupled with the CI pipeline and the time it takes to parse every DAG, the entire process can take over 10 minutes. So, wait until some time has passed before raising the issue.

I created a DAG, but why can't I unpause it?

If you can't unpause a DAG, the issue is access related.

> If your issue is with a **brand new DAG** or you were recently added to the `OWNERS.yaml` file in the DAG directory, **there is a 15 minute delay** for syncing and updating roles and permissions. This is due to the baked in delay in the Airflow Scheduler. After your changes have been merged in and your master pipeline succeeds, it will take at least 10-15 minutes before permissions are updated on permissions have been propagated for the newly added users in the `OWNERS.yaml` file.

1. Review the section "How is access provisioned?", and ensure you're included in a product group with access, or in the list of users in the OWNERS.yaml file.
2. If you have access to the workspace, then the issue is likely the role hasn't been created or updated yet. After a DAG is created and the Hieroglyph pipeline finishes, the Airflow processor needs to run, in which DAG files are parsed and DAG objects are created. The web server eventually creates and or updates roles with DAG permissions, including the permission to enable a DAG. So, wait until some time has passed and periodically check your DAG before raising the issue.
    a. The web server creates and updates roles based on the DAGs serialized by the processor. If the web server runs before the processor is finished, then roles might not get created or updated to grant permission to your DAG. If a significant amount of time has passed since the Hieroglyph pipeline finished, e.g. over 20 to 30 minutes, please reach out on #help-airflow, and we will take further action.
3. If this is related to a DAG in a new workspace, try logging out of Airflow, then logging back in. This will trigger a process to update your roles within Airflow.

I enabled my DAG. Why isn't it running?

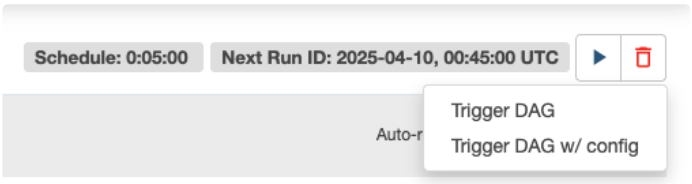
This is likely due to the specified scheduling configurations, i.e. the start and end dates, schedule interval, or catchup setting.

1. A full interval needs to have passed before the scheduled run can begin. If your interval is daily at 1000, the 2025-04-01 run will be scheduled for 2025-04-02 at 1000. This will ensure the full interval is accounted for prior to executing a DAG run.
2. Check your start date. If your start date is a date in the future, your DAG will not run.
3. Check if you have an end date. If you have one and it's a date in the past or conflicts with your next scheduled run, i.e. it's before the data interval end date, your DAG will not run.

4. If you were expecting a DAG run for each interval from your start date to the current date, ensure you set catchup to true.

How can I pass custom configuration during a manual run?

After navigating to your DAG, you can click the play button in the top right corner, then select the option `Trigger DAG w/ config`. You'll be able to enter configuration in JSON format. It will be accessible as a Python dictionary within your DAG. You can use the dag_run template variable to reference it, e.g. `{{ dag_run.conf.get("my_key", "my_default_value") }}`. Please be sure to use the get method when referencing a key, instead of square bracket notation (dictionary lookup).



What template date variables do I have access to?

Airflow provides a comprehensive set of date variables. Several are highlighted below, but refer to the official documentation for the complete list.

1. `{{ data_interval_start }}`
   a. This is a Pendulum datetime object. It represents the start of the data interval. If your DAG runs daily at 1000 UTC, then its value for a DAG run on 2025-04-01, would be a datetime object equal to 2025-04-01, 1000 UTC.
2. `{{ data_interval_end }}`
   a. This is a Pendulum datetime object. It represents the end of the data interval. If your DAG runs daily at 1000 UTC, then its value for a DAG run on 2025-04-01, would be a datetime object equal to 2025-04-02, 1000 UTC.
3. `{{ logical_date }}`
   a. This is a Pendulum datetime object. It will generally be equal to the data interval start, but not always. It's particularly used to identify a DAG. So, if you trigger a manual run, its value will likely be equal to the datetime in which you triggered the run.
4. `{{ ds }}`
   a. This is a datetime string. It's equal to the logical date in YYYY-MM-DD format.
   b. This is also a filter, though, and it can be used in conjunction with Pendulum datetime objects. So, if you wanted the data interval start date in the date format, you could do so

with `{{ data_interval_start | ds }}`.

5. `{{ ts }}`
    a. This is a datetime string. It's equal to the logical date in ISO format, e.g. 2025-04-01T00:00:00+00:00.
    b. Similar to ds, this is also a filter, so it can be used to format Pendulum datetime objects accordingly, `{{ data_interval_start | ts }}`.

Is it okay to change the start date or schedule interval for running backfills?

Although it's not explicitly mentioned in official Airflow documentation, members of the community have some general consensus that it's best to create a new DAG if the start date or schedule interval needs to be changed. The scheduler will reference past intervals to determine what to schedule in the future. If there are conflicts, it could result in unpredictable behavior. So, our recommendation is to copy the DAG, change the DAG ID, and update the start date and or schedule interval.

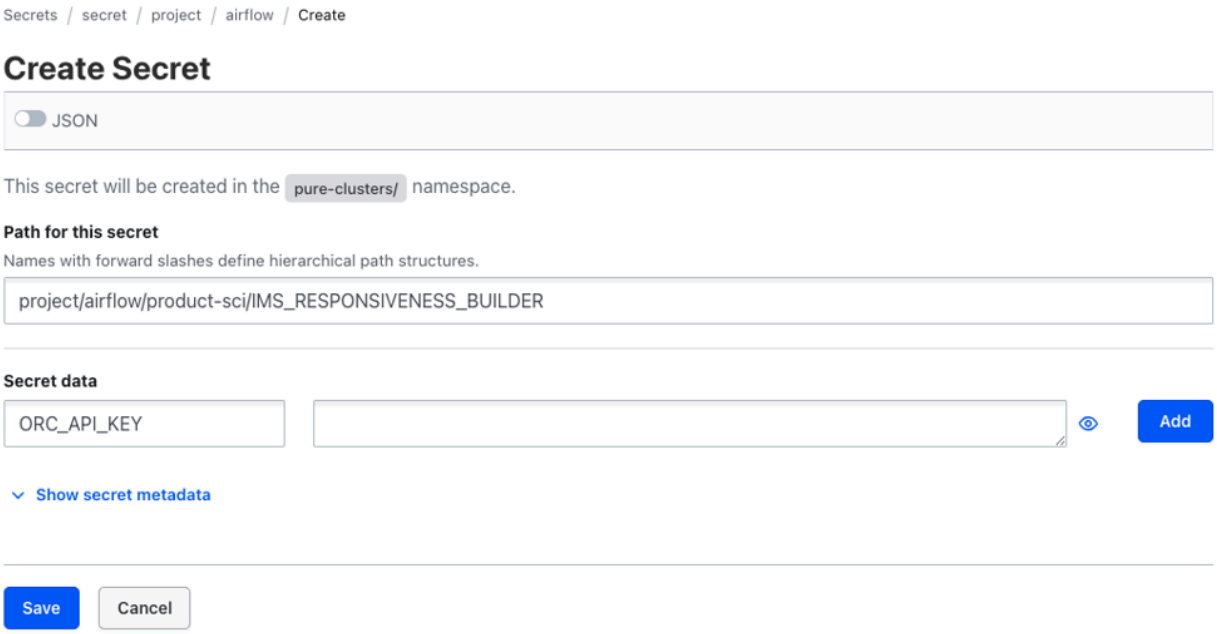
Why is my Orc task returning a 403 error?

There are likely two reasons.

1. Separate API keys are provisioned for QA and production. Check your key, and refer to the wiki accordingly.
2. Add your Orc API keys to VAULT pure-clusters. https://vault.indeed.tech/ui/vault/secrets/secret/kv/list/project/airflow/?namespace=pure-clusters. Make sure the path is the same as the one defined in your DAG. i.e.

```
        [dags.tasks.env_vars]
             ORC_USER = "messaging-airflow-prod"
             ORC_API_KEY =
"vault:secret/data/project/airflow/your_product_group/YOUR_BUILDER#ORC_API_KEY"
```

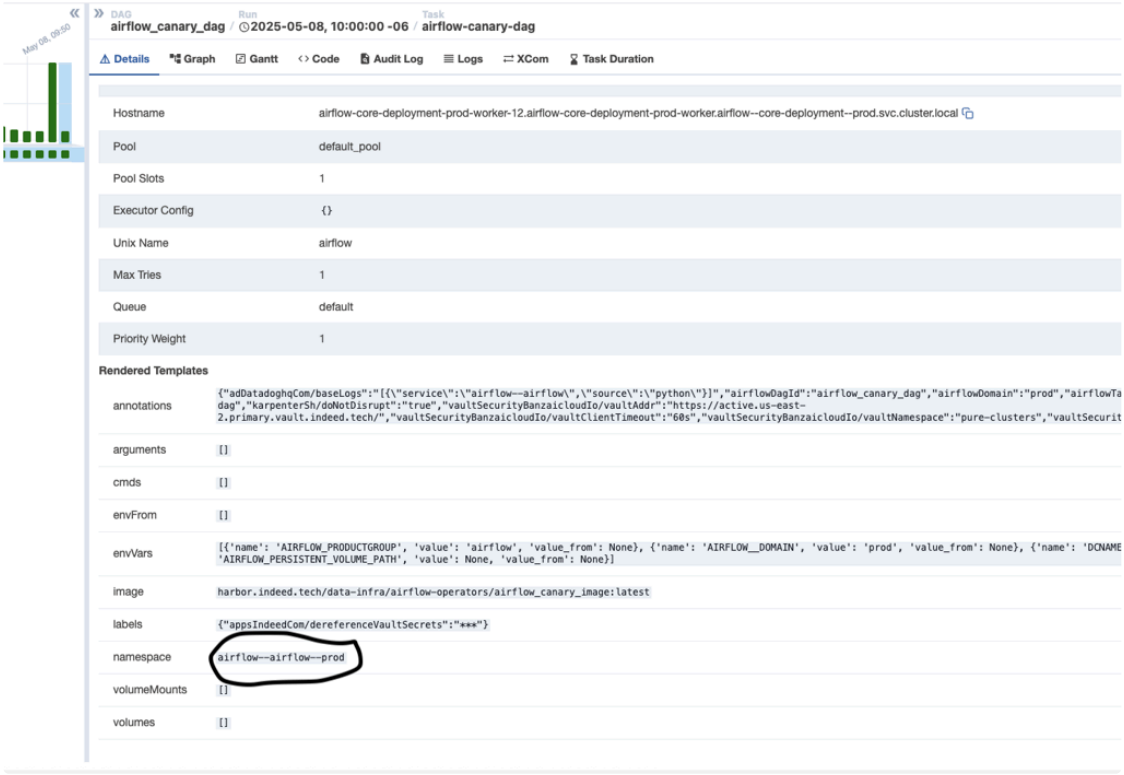
Example screenshot below for adding the API key for IMS_RESPONSIVENESS_BUILDER.



3. The Orc runner image appends a username to the API key in the headers, so ensure the username is not included. Refer to the implementation in the image repo.

How can I view the namespace that my DAG Tasks are running in?

Find your Task's k8s namespace by going to the Task detail page and finding the corresponding k8s namespace



How can I see logs that are being populated by a currently running task within my DAG?

If you have a task that is currently running, there should be an active pod running that corresponds to your task. You can view the logs for all active pods under your Task's k8 namespace.

First, find your task's k8 namespace.
View the pods that are currently available within your specified namespace:

```
kubectl --context awscmhinfra3 -n [your-namespace] get pods
```

View the real-time logs for a specified pod contained in the list that you obtained above:

```
kubectl --context awscmhinfra3 -n [your-namespace] logs -f [some-pod-name]
```

Note, you can provide search parameters for the kubectl logs to filter by a specified time window, log count, timestamps, etc.

My tasks are stuck in `WARNING - Pod not yet started: [some-pod-name]`

If your tasks are stuck with the message `Pod not yet started` it generally means there is an issue starting your main container in kubernetes. You can investigate it by:

First, find your task's k8 namespace.

Run the following command to check the pod status:

```
kubectl --context awscmhinfra3 -n [your-namespace] get [some-pod-name]
```

If the pod is in `initializing` or `containerCreating` state, it likely means that the cluster is taking longer than usual to allocate your pod, you should wait a bit longer in that case and if this persisted, reach out to the #help-kubernetes channel for more information.
If the pod is in any `error` state for instance, `CreateContainerConfigError` you will have to dive into the pod details and investigate the reason.

```
kubectl --context awscmhinfra3 -n [your-namespace] get [some-pod-name] -o yaml
```

for example,

Address the issue if the message is clear and helpful to you or reach out to #help-airflow with the information you find to get assistance from the airflow team.