Linux Academy

The Data Dossier

Return to Table of Contents

Choose a Lesson

Cloud Composer Overview

Hands On - Cloud Composer

Cloud Composer Overview

Next

What is Cloud Composer?

- Fully managed **Apache Airflow** implementation:
    - Infrastructure/OS handled for you

open source

What is Apache Airflow?

- Programatically create, schedule, and monitor data workflows

Why is this important?

- Automation and monitoring
- Big data pipelines are often a multi-step, complex process:
    - Create resources in multiple services
    - Process and move data from one service to another
    - Remove resources when they complete a task (e.g. Dataproc cluster)
- Collaborate workflow process with other team members

How Airflow/Composer helps

- Automates the above steps, including scheduling
- Built on open source, using Python as common language
- Easy to work with, and share workflow with others
- Works with non-GCP providers (on-premises, other clouds)

Return to Table of Contents

Cloud Composer Overview

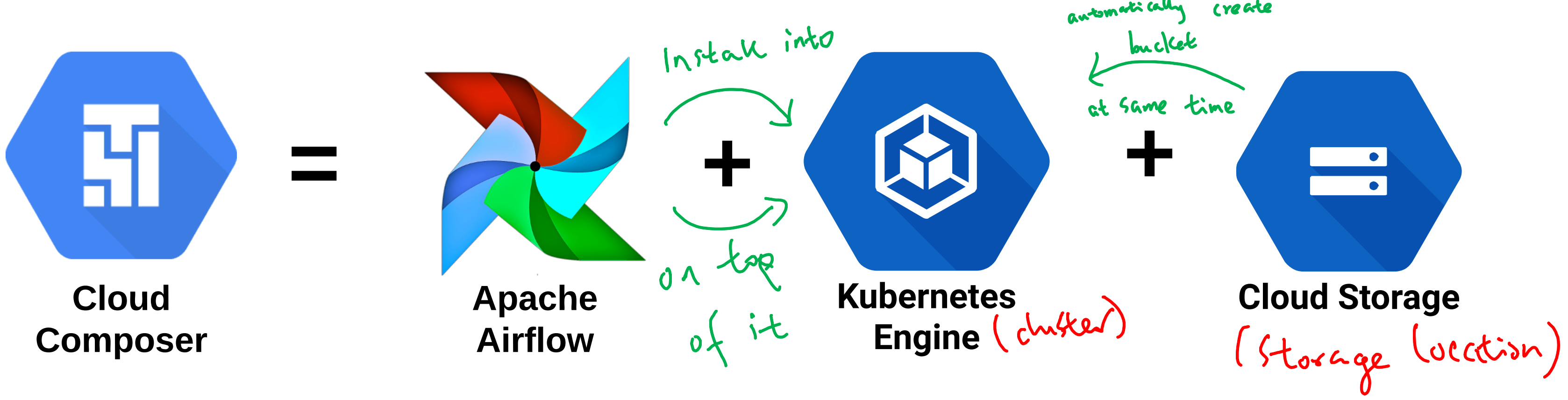
Previous

Next

How It Works

Behind the scenes:

- GKE cluster with Airflow implemented
- Cloud Storage bucket for workflow files (and other application files)


Cloud Composer
=
Apache Airflow
Install into
+
on top of it
Kubernetes Engine (cluster)
automatically create bucket at same time
+
Cloud Storage (Storage location)

Workflows?

- Orchestrate data pipelines:
  - Like a walkthrough of tasks to run
- Format = Direct Acyclic Graph (DAG): ← important terminology
  - Written in Python
  - Collection of organized tasks that you want to schedule and run
- **Cloud Composer** creates **workflows** using **DAG** files

python application

Summary!

The Process

- Create Composer Environment (kubernete instance cluster)
- Set Composer variables (i.e. project ID, GCS bucket, region)
- Add Workflows (DAG files), which Composer will execute

different from Gcloud variables

Return to Table of Contents

Cloud Composer Overview

Previous

to cloud storage bucket
(to save cost)

Examples and Exam Perspective

- Create a Dataproc cluster, submit a job, and then delete the cluster.
- Execute a Cloud Dataflow pipeline from data in GCS, and write output to BigQuery.
- Ingest third party data into Cloud Dataflow, process, then upload to GCS.
- **Exam perspective:** Know what DAGs are, and why you'd want to use workflows.

Exam topics!!!

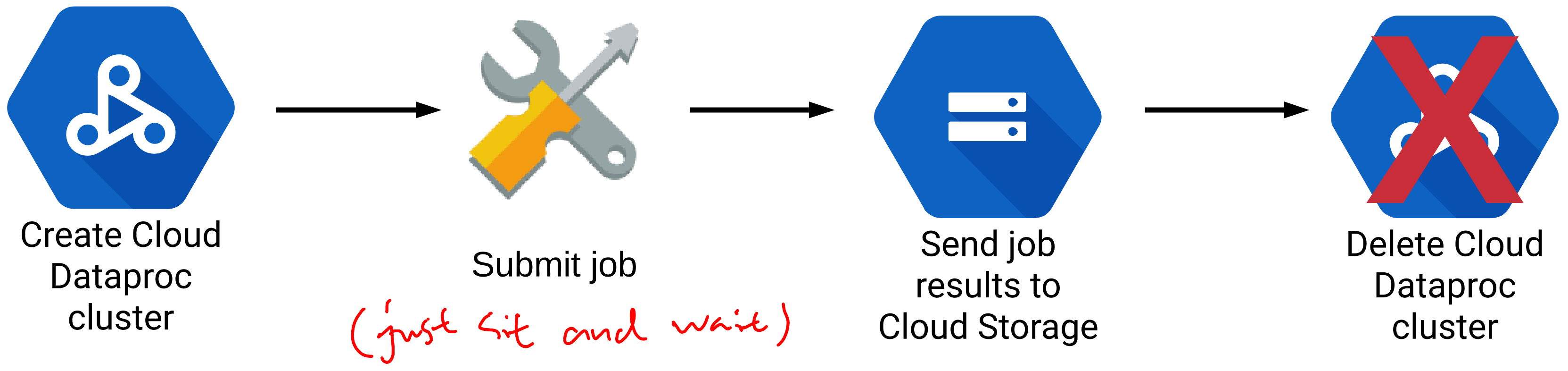
Hands On - Cloud Composer

Next

The Process:

- Create the Composer environment. cluster
- Then create the GCS bucket for Dataproc output.
- Assign Cloud Composer variables.
- Upload the workflow file to DAG folder.
- View the results.

Automatic processes -- Workflow


Create Cloud Dataproc cluster
Submit job
(just sit and wait)
Send job results to Cloud Storage
Delete Cloud Dataproc cluster

Create Composer Environment

- Enable Composer/Dataproc API
- Create environment in closest region:
  - What's happening?
  - Creating GKE cluster + GCS bucket

Create GCS bucket to output Dataproc results

- `gsutil mb -l us-central1 gs://output-$DEVSHELL_PROJECT_ID`

Return to Table of Contents

Choose a Lesson

Cloud Composer Overview

Hands On - Cloud Composer

Hands On - Cloud Composer

Previous

Configure Cloud Composer Variables

- Format
    - **`gcloud composer environments run (ENVIRONMENT_NAME) --location (LOCATION) variables -- --set (KEY VALUE)`**

```
gcloud composer environments run my-environment --location us-central1 variables -- --set gcp_project (PROJECT-ID)
gcloud composer environments run my-environment --location us-central1 variables -- --set gcs_bucket gs://output-(PROJECT-ID)
gcloud composer environments run my-environment --location us-central1 variables -- --set gce_zone us-central1-c
```

Add workflow file (Python) to Composer DAG folder:

- github link

Next step? There is none! Cloud Composer will take it from here...