

Moving to Managed Airflow

Chris Munns

Senior Manager/Principal Developer Advocate - Serverless

Amazon Web Services

About me

Chris Munns - munns@amazon.com, [@chrismunns](https://twitter.com/chrismunns)

- Sr Manager/ Principal Developer Advocate – Serverless
- New Yorker (ehhh...ish.. kids/ burbs/ ya know?)

Previously:

- AWS Business Development Manager – DevOps, July '15 - Feb '17
- AWS Solutions Architect Nov '11- Dec '14
- Formerly on operations teams @Etsy and @Meetup
- Little time at a hedge fund, Xerox and a few other startups
- Rochester Institute of Technology: Applied Networking and Systems Administration '05
- Internet infrastructure geek



Agenda

Amazon Managed Workflows for Apache Airflow (MWAA)

- What it is
- How it works
- Setting it up
- Working with it

We will not get deep into DAG design or Airflow use-cases today.

Apache Airflow

The screenshot displays the Apache Airflow web interface for a DAG named "movie_data_pipeline". The interface includes a top navigation bar with "Airflow" logo, "DAGs", "Security", "Browse", "Admin", "Docs", and "About" menus. The current date and time are "2020-10-03, 16:56:07 UTC".

The DAG is currently "On" and has a "schedule: @once" configuration. The interface offers various view options: "Graph View" (selected), "Tree View", "Task Duration", "Task Tries", "Landing Times", "Gantt", "Details", "Code", "Trigger DAG", "Refresh", and "Delete".

Control elements include a "running" status indicator, a "Base date" field set to "2020-10-03T16:55:38Z", a "Number of runs" dropdown set to "25", a "Run" dropdown set to "manual_2020-10-03T16:55:37.728665+00:00", a "Layout" dropdown set to "Left->Right", and a "Go" button. A search bar is also present.

A horizontal menu lists various operators: AWSAthenaOperator, AWSBatchOperator, ECSOperator, EmrCreateJobFlowOperator (highlighted), EmrStepSensor, HttpSensor, PythonOperator, S3DeleteObjectsOperator, S3KeySensor, S3PrefixSensor, SageMakerTrainingOperator, and SimpleHttpOperator. A legend below lists task statuses: success, running, failed, skipped, upstream_failed, up_for_reschedule, up_for_retry, queued, and no_status.

The main area shows a DAG graph with the following tasks and dependencies:

- Is_Rental_API_Available → Load_Rentals
- Is_Digital_Download_API_Available → Load_Digital_Downloads
- Is_Online_Store_Sales_API_Available → Load_Online_Store_Sales
- Is_FB_Impression_Object_Store_Available → Load_FB_Impressions
- Is_TW_Likes_Object_Store_Available → Load_TW_Likes
- Load_Rentals → Process_Sales_EMR
- Load_Digital_Downloads → Process_Sales_EMR
- Load_Online_Store_Sales → Process_Sales_EMR
- Load_FB_Impressions → Process_Social_ECS
- Load_TW_Likes → Process_Social_ECS
- Process_Sales_EMR → Wait_For_Process_Sales
- Process_Social_ECS → Wait_For_Process_Social
- Wait_For_Process_Sales → Query_Sales_Glue
- Wait_For_Process_Social → Query_Social_Athena
- Wait_For_Process_Social → Analyze_Social_Sagemaker
- Query_Sales_Glue → Clean_Data_S3
- Query_Social_Athena → Clean_Data_S3
- Analyze_Social_Sagemaker → Clean_Data_S3
- Clean_Data_S3 → Archive_Data_DynamoDB
- Clean_Data_S3 → Store_Data_Redshift

Apache Airflow use cases



ETL



AI/ML



DevOps

Apache Airflow components



Scheduler



Worker



Web server

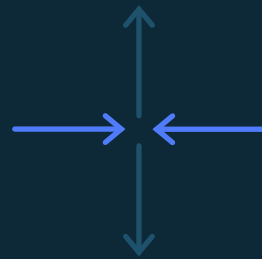


Meta database

Apache Airflow challenges



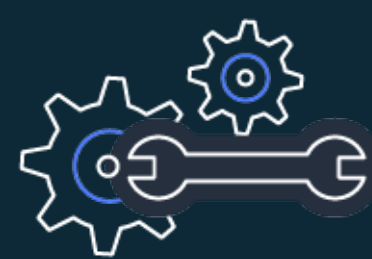
Setup



Scaling



Security



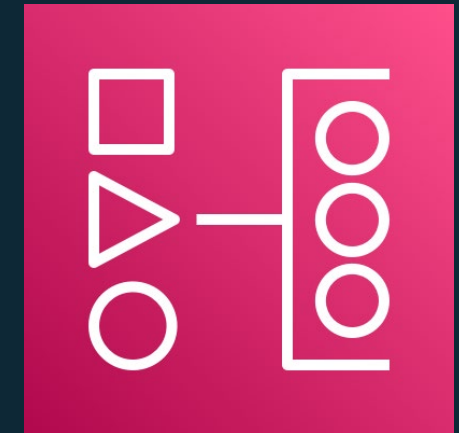
Upgrades



Maintenance

Amazon Managed Workflows for Apache Airflow (MWAA)

A new managed service for Apache Airflow that makes it easy for data engineers and data scientists to execute data processing workflows on AWS



Airflow environments

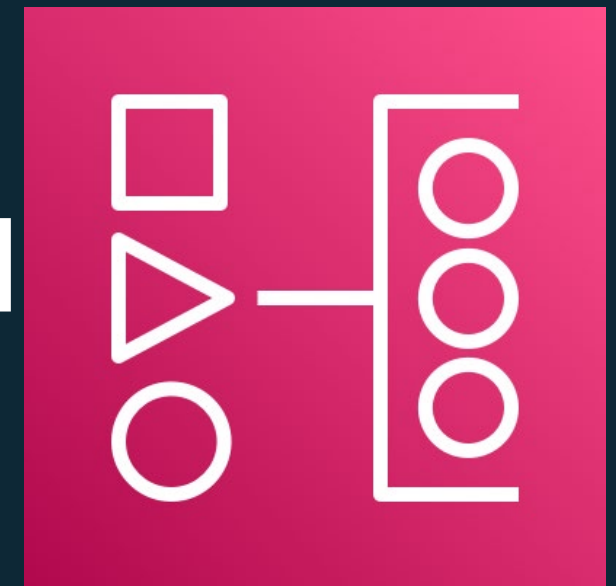
Environments (2) Refresh Edit Delete Actions Create environment

Find environments

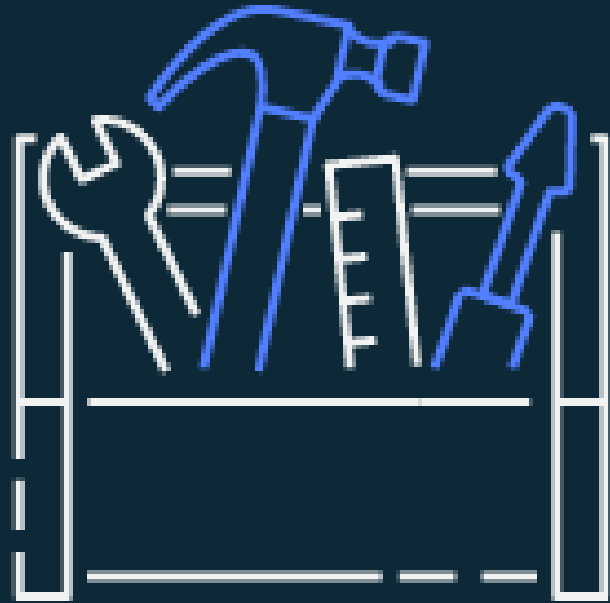
Name	Status	Created date	Airflow version	Airflow UI
<input type="radio"/> AirflowGamma1	Available	Nov 05, 2020 09:17:16 (UTC-08:00)	1.10.10	Open Airflow UI
<input type="radio"/> AirflowBeta2	Available	Oct 05, 2020 12:24:54 (UTC-07:00)	1.10.10	Open Airflow UI

Announced in November 2020!

What does MWAA do for you



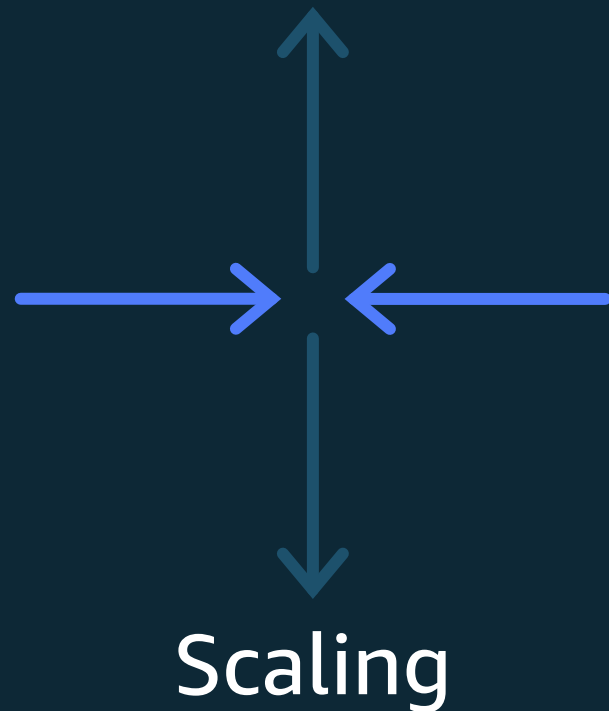
What Amazon MWAA does for you



Setup

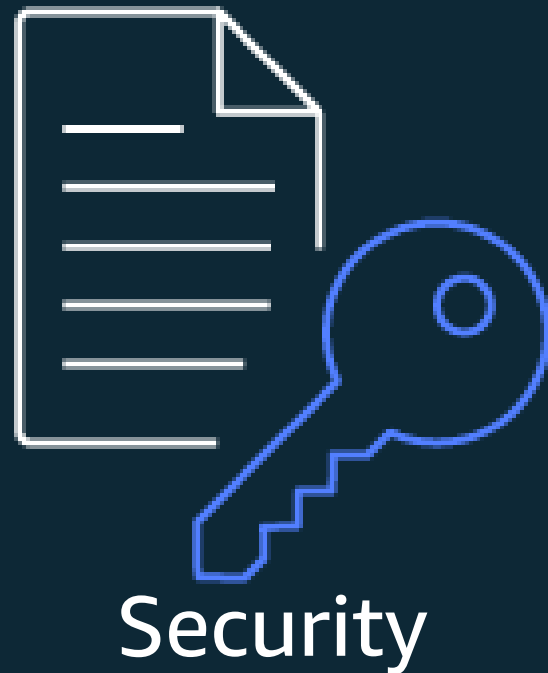
- Deploys Apache Airflow rapidly
- Same open-source Airflow
- Same Airflow UI
- AWS Management Console, AWS CLI, API, or AWS CloudFormation

What Amazon MWAA does for you



- Worker scaling, no configurations required
- Uses Celery Executor
- Amazon ECS on AWS Fargate

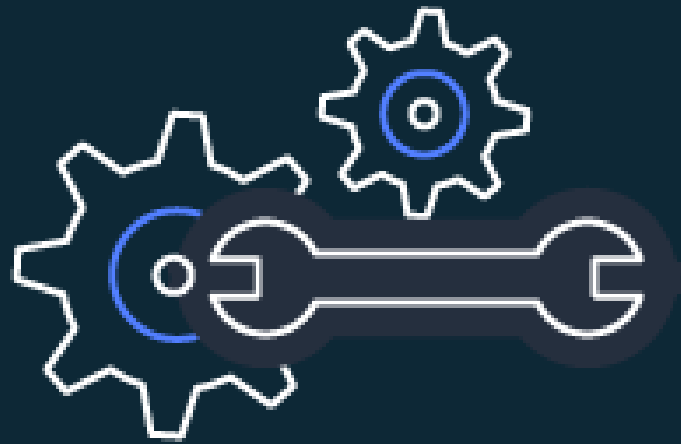
What Amazon MWAA does for you



Security

- Integrated AuthN and AuthZ with AWS Identity and Access Management (IAM)
- Airflow UI with either VPC or public, secured endpoint
- Workers assume IAM execution roles for simplified access to AWS services
- Workers and Scheduler run in customer VPC for access to customer resources

What Amazon MWAA does for you



Upgrades

- Maintenance windows for upgrades
- Snapshot and rollback in case of failure

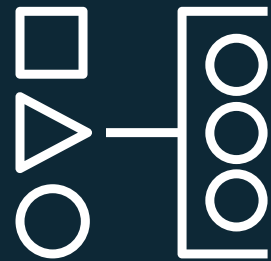
What Amazon MWAA does for you



Maintenance

- Integrated monitoring with Amazon CloudWatch
- Multi-AZ
- Automatically restarts on failure

How Amazon MWAA works



Create an
MWAA
environment

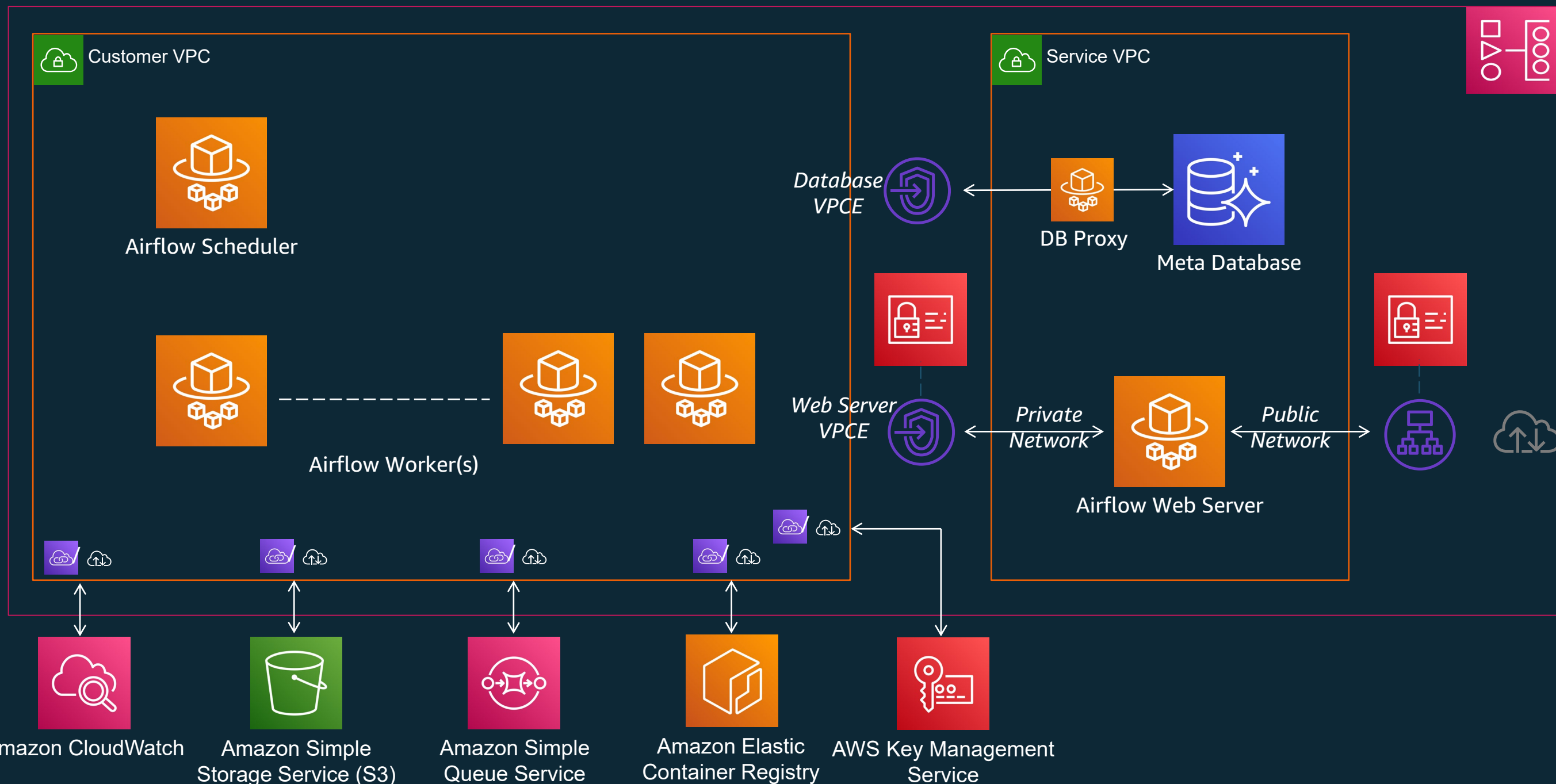


Copy your DAGs
and plugins to
Amazon S3

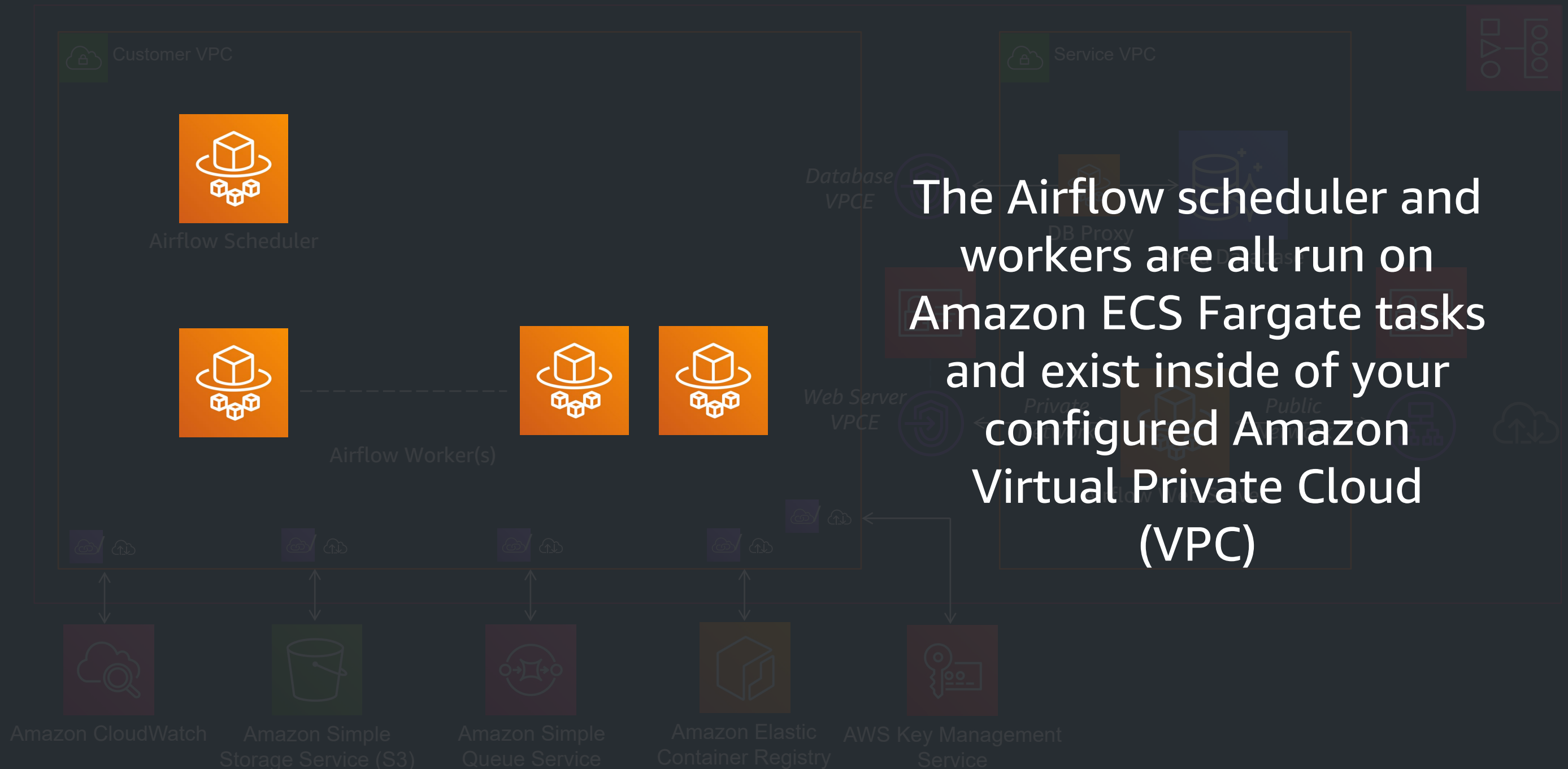


Access the
Airflow UI

Amazon MWAA Architecture



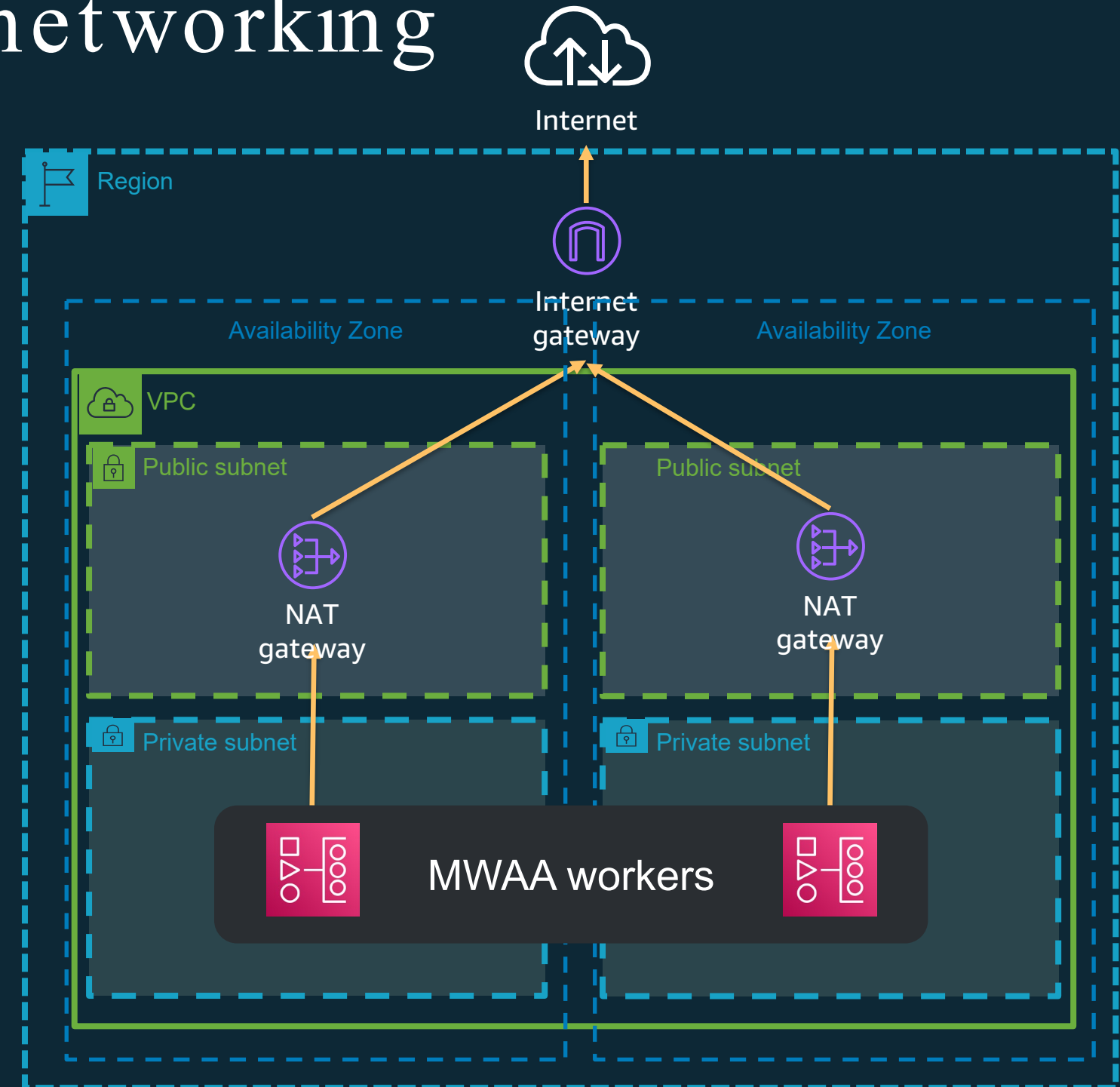
Amazon MWAA Architecture



MWAA and Amazon VPC networking

You do need an Amazon Virtual Private Network (VPC) configured to run your MWAA scheduler and workers:

- We recommend multiple Availability Zones for high availability configurations
- Security Groups can be configured for fine grained access
- MWAA scheduler and workers need access to core services either via NAT-Gateways or via AWS PrivateLink endpoints

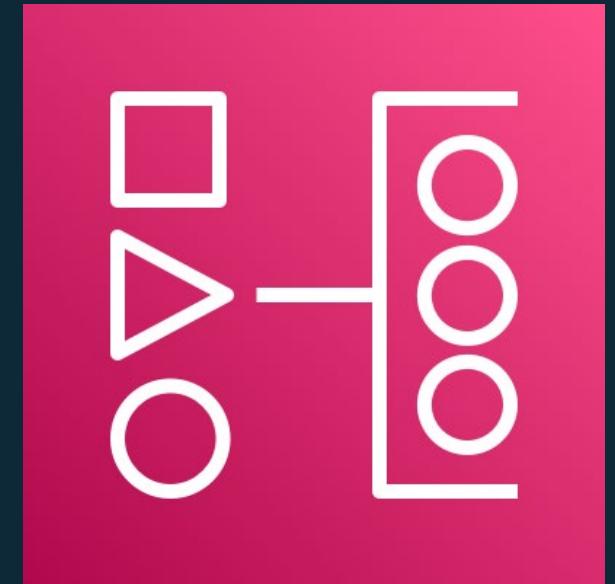


Working with Amazon MWAA

Working with MWAA and Airflow DAGs

Directed Acyclic Graphs (DAGs) are the unit of work and the structure for jobs run in Airflow. They have several common components:

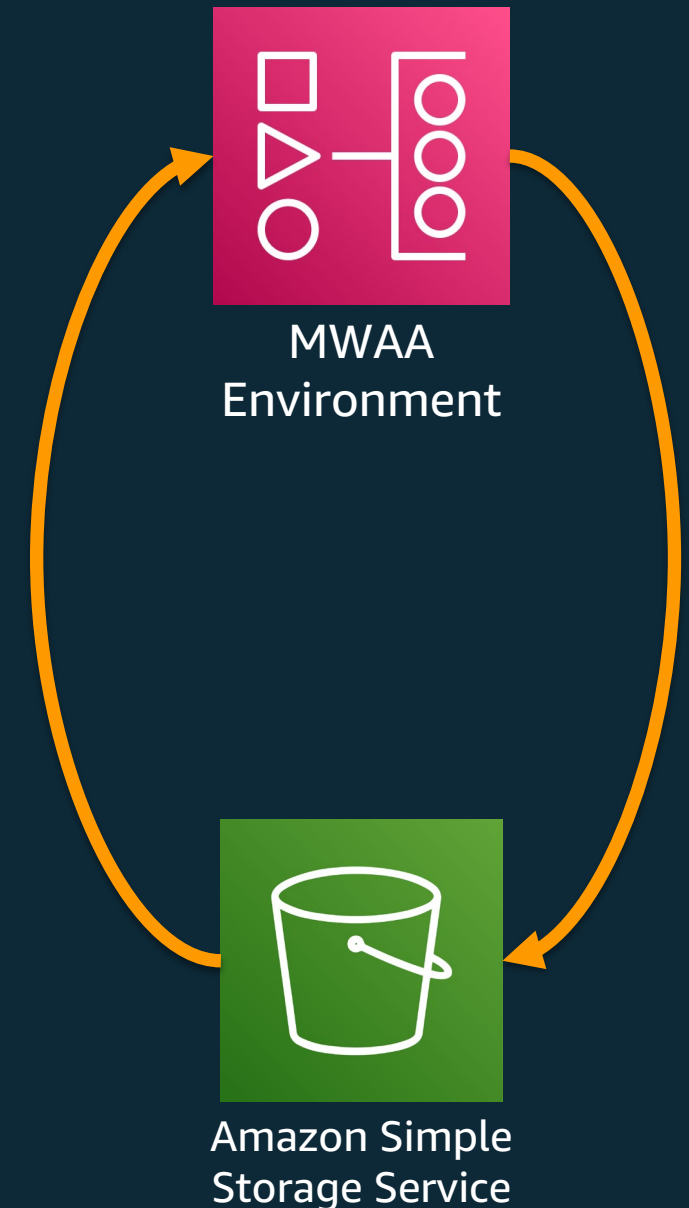
- DAG definition
- Plugins
- Requirements
- Configurations



Working with MWAA and Airflow DAGs

DAG definitions:

1. Upload directly to Amazon S3
2. Configure environment for the path of the bucket namespace DAGs will be stored in
3. Your environment checks S3 every 30 seconds for new DAGs and related files
4. After import from S3, DAGs show available in the Airflow web interface



Working with MWAA and Airflow DAGs

MWAA supports the built-in plugin manager in Airflow

1. Zip up plugins directory
2. Upload directly to Amazon S3
3. Configure environment for the object in the bucket namespace plugins will be stored in
4. Your environment checks S3 every 30 seconds for new plugins
5. After import from S3, plugins show installed in the Airflow web interface
6. Configure DAGs to use plugins

```
__init__.py
my_airflow_plugin.py
hooks/
  |-- __init__.py
  |-- my_airflow_hook.py
operators/
  |-- __init__.py
  |-- my_airflow_operator.py
sensors/
  |-- __init__.py
  |-- my_airflow_sensor.py
```

Working with MWAA and Airflow DAGs

MWAA supports Python dependencies for your DAG code

1. Create a Python requirements.txt with required dependencies
2. Upload directly to Amazon S3
3. Configure environment for the version of the requirements.txt file in S3
4. Your environment will pull down that version of the file
5. Logs for the scheduler will show processing of the requirements.txt file
6. Configure DAG code to use dependencies

```
apache-airflow-providers-apache-  
hive[amazon]>=1.0.1  
hmsclient>=0.1.0  
pyhive[hive]>=0.6.0  
thrift>=0.9.2  
apache-airflow-providers-  
presto>=1.0.1  
presto-python-client>=0.7.0
```

(example requirements.txt file)

Working with MWAA and Airflow DAGs

MWAA supports Airflow configuration settings typically seen in `airflow.cfg`

- Settings are configured in the Amazon MWAA console
- Some settings appear in the dropdown, but the full list can be found in the Airflow documentation:

<https://airflow.apache.org/docs/stable/configurations-ref.html>

Airflow configuration options - optional [Info](#)

Modify the default settings for Airflow configuration options. You can select an option from the suggestion list or type one manually.

Configuration option	Custom value	
<input type="text" value="smtp.smtp_host"/>	<input type="text" value="smtp.gmail.com"/>	<input type="button" value="Remove"/>
<input type="text" value="smtp.smtp_mail_from"/>	<input type="text" value="<your email>@gmail.com"/>	<input type="button" value="Remove"/>
<input type="text" value="smtp.smtp_password"/>	<input type="text" value="<your 16 digit app password>"/>	<input type="button" value="Remove"/>

Integrations

Amazon MWAA has integration with several other AWS services via plugins:

- Amazon EKS
- Amazon EMR
- Amazon RDS
- AWS Glue
- AWS Batch
- Amazon DynamoDB

more!



DEMO

MWAA environments and pricing

Environment Instance Pricing

Region:


Size	Scheduler vCPU	Worker vCPU	Web Server vCPU	Recommended Workflow Capacity	Price Per Hour
Small	1	1	0.5	Up to 50 DAGs	\$0.49
Medium	2	2	1	Up to 250 DAGs	\$0.74
Large	4	4	2	Up to 1000 DAGs	\$0.99

Additional Worker Instance Pricing

Size	Price Per Hour
Small	\$0.055
Medium	\$0.11
Large	\$0.22

Meta Database Storage

\$0.10 per GB Month

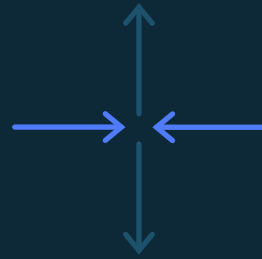


<https://aws.amazon.com/managed-workflows-for-apache-airflow/pricing/>

Amazon Managed Workflows for Apache Airflow benefits



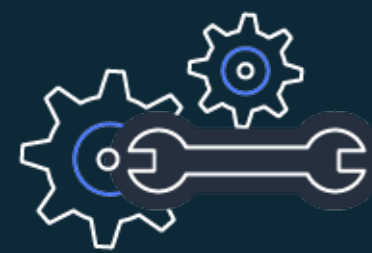
Setup



Scaling



Security



Upgrades



Maintenance

Resources

Stay “In The Know”

- Amazon MWAA docs <https://docs.aws.amazon.com/mwaa>
- Amazon MWAA landing page <https://aws.amazon.com/mwaa>
- #airflow-aws Slack Channel: <https://apache-airflow.slack.com>



Thank you!

Chris Munns

munns@amazon.com

[@chrismunns](#)