Amazon Managed Workflows for Apache Airflow

Agenda

Why Apache Airflow?

Overview of Apache Airflow

How Amazon MWAA does Apache Airflow?

Demo

Amazon MWAA Tips

Resources

Speaker

Sanchit Jain

FOLLOW ME

- Working as Data Analytics & Cloud Practice Lead - AWS Alliance at Quantiphi
- Recognition - AWS Hero & AWS Ambassador, MVP by Multicloud4u & AWS User Group Mumbai Lead
- Speaker at various events like AWS re:Invent, AWS Summit, AWS User Groups, AWS Community Days, and various educational institutes
- Follow me @ LinkedIn, Twitter or Github

Why Apache Airflow?

Problems with CRON & similar options

- CRON does a poor job at handling task dependencies and viewing them.
- Poor or no strategy for retrying tasks or backfills.
- Limited data about task times, execution durations and failures
- Need to ssh into server to check logs and interact.
- No easy way to scale beyond one machine.
- People mostly write jobs in BASH or XML 🤮
- Some questions that are hard to answer:
  - Do you know when your CRON jobs fail?
  - Can you spot when your tasks become 3x slower?
  - Can you visualize what's currently running? What's queued? Do you have reusable components you can use across workflows?

Overview of Apache Airflow

Apache Airflow : What is it?

Pipelines are configured as code, allowing for dynamic pipeline generation

A platform to monitor and control data pipelines

Easily define your own operators, executors and extend the library

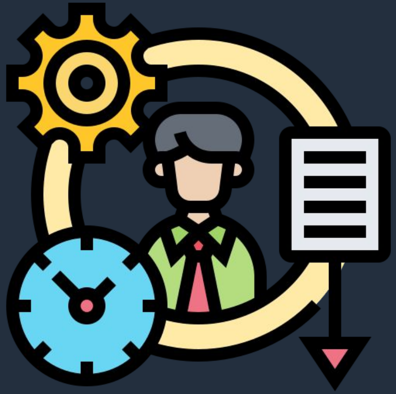
100% developed in Python

It's all about DAGs

Apache Airflow : Why do I need it?

- There are several critical processes to be maintained & monitored
- Different kinds of jobs in different tools
- Jobs require dependencies and run in a specific order
- A consistent notification method
- Action must be takes in case things go wrong

Apache Airflow : Very Flexible!

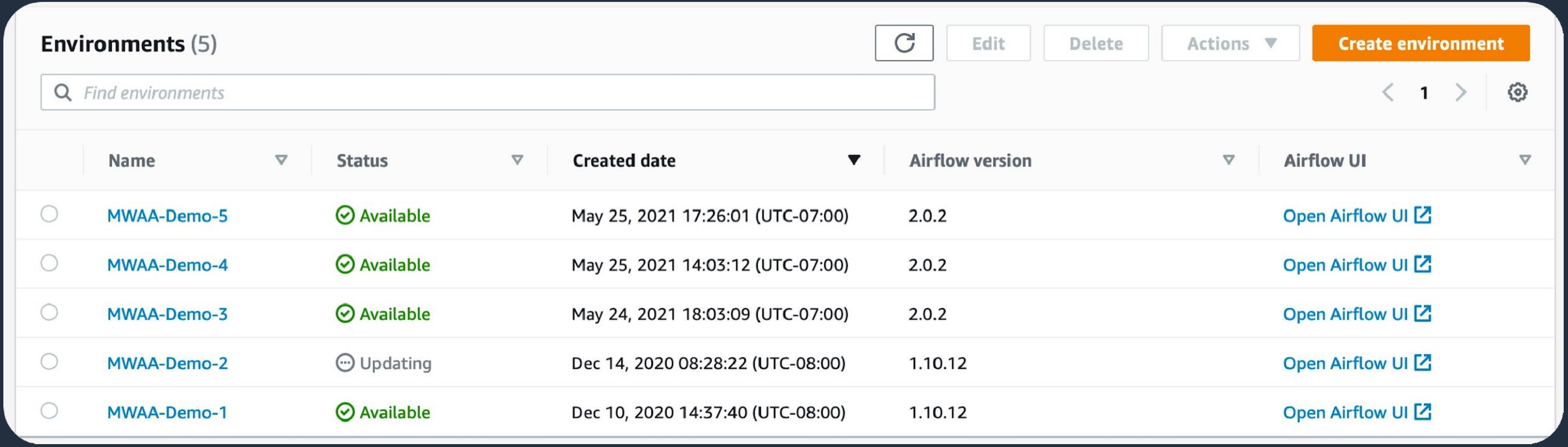


- Easily extensible
- Efficient CLI
- DAGS are made in code
- Backfill control
- Rich User Interface
- Allow communication between task

How Amazon MWAA does Apache Airflow?

Amazon Managed Workflows for Apache Airflow (MWAA)

- A managed service for Apache Airflow that makes it easy for data engineers and data scientists to execute data processing workflows on AWS
- Released November 24, 2020, added Airflow 2.0 support May 26, 2021



Environments (5)

	Name	Status	Created date	Airflow version	Airflow UI
○	MWAA-Demo-5	Available	May 25, 2021 17:26:01 (UTC-07:00)	2.0.2	Open Airflow UI
○	MWAA-Demo-4	Available	May 25, 2021 14:03:12 (UTC-07:00)	2.0.2	Open Airflow UI
○	MWAA-Demo-3	Available	May 24, 2021 18:03:09 (UTC-07:00)	2.0.2	Open Airflow UI
○	MWAA-Demo-2	Updating	Dec 14, 2020 08:28:22 (UTC-08:00)	1.10.12	Open Airflow UI
○	MWAA-Demo-1	Available	Dec 10, 2020 14:37:40 (UTC-08:00)	1.10.12	Open Airflow UI

How Amazon MWAA Helps?

- Deployments & Operations
  - Easy to setup up and maintain
- Availability and Sizing
  - Multi AZ/HA with Airflow 2.0 on Amazon ECS Fargate
- Scaling
  - Auto scaling with Celery executor
- Security
  - IAM and VPC

Setup

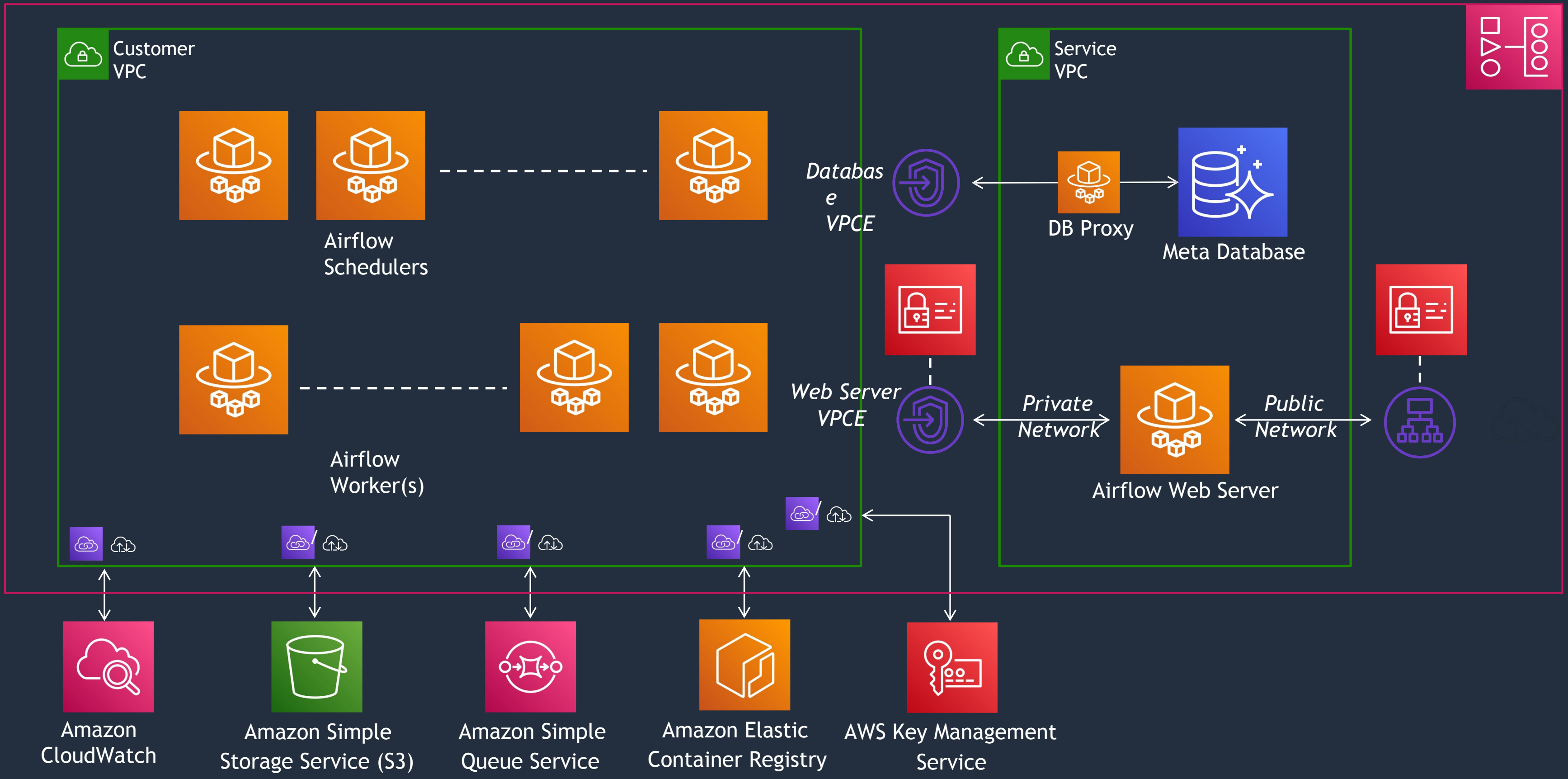
Upgrades

Scaling

Security

Maintenance

Amazon MWAA Architecture



Apache Airflow components

Scheduler

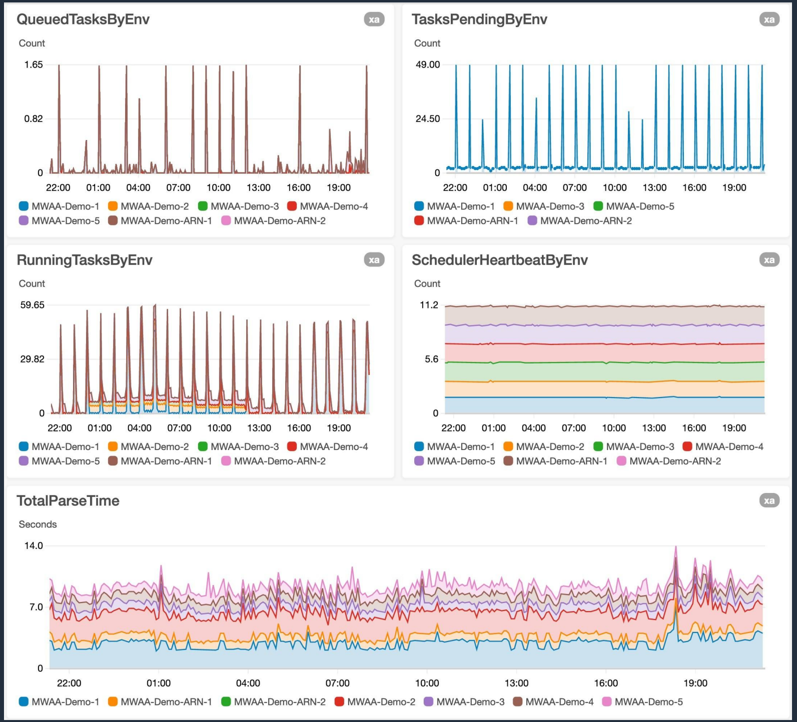
Web server

Worker

Meta database

Deployments and Operations

- Easy to set up multiple small environments
- Prod/Dev environments
- CloudWatch
    - Logging
    - Metrics/Alarms
    - Dashboards
- CloudFormation/Terraform
    - Configure User Access
    - Environment (Execution Role) access
    - Airflow Configurations without accessing UI


QueuedTasksByEnv

TasksPendingByEnv

RunningTasksByEnv

SchedulerHeartbeatByEnv

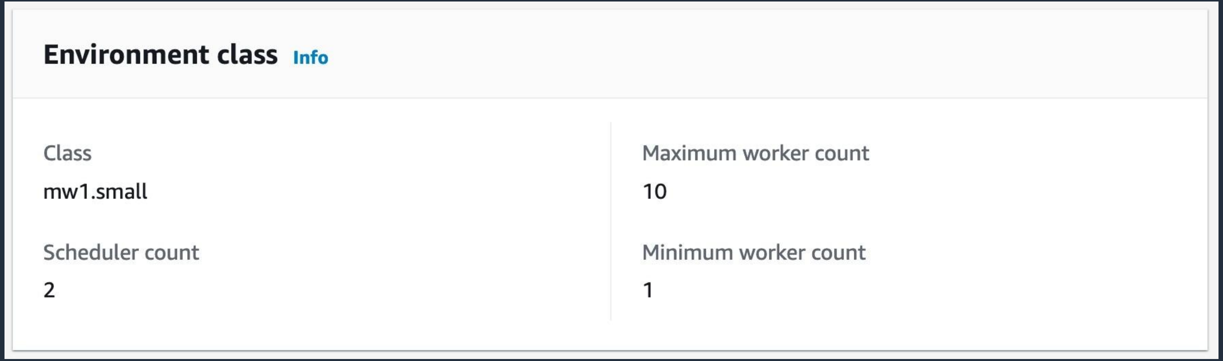
TotalParseTime

Scaling/Environment Sizes

- Using Airflow metrics

- (Tasks running + tasks queued)/tasks per container = number of containers required

- Downscaling occurs when (Tasks running+tasks queued)=0



Environment class Info

Class mw1.small	Maximum worker count 10
Scheduler count 2	Minimum worker count 1

Security with MWAA

- Auth (IAM/Federation)
- VPC & Security Group
- AWS Secrets Manager
- IAM Execution Role

CI/CD

- Versioned S3 Bucket deployment target
- Flexible integration with existing CI/CD pipelines
- DAGs automatically synchronized every 30-60s
- Custom Plugin and Python dependency changes require Environment update

Demo

How Amazon MWAA works?

Create an MWAA environment

Copy your DAGs & plugins to Amazon S3

Access the Airflow UI

Key Concepts

- DAG (Directed Acyclic Graph) are collections of tasks and describe how to run a workflow written in Python

- Task defines a unit of work within a DAG; it is represented as a node in the DAG graph

- Operators determine what gets done in that task when a DAG runs

- Workflow - DAG + Operator + Task

- Task instance represents a specific run of a task characterized by a DAG, a task, and a point in time

Resources

Resources

- Amazon MWAA docs https://docs.aws.amazon.com/mwaa
- Amazon MWAA product page https://aws.amazon.com/mwaa
- #airflow-aws Slack Channel: https://apache-airflow.slack.com
- GitHub samples: https://github.com/aws-samples/amazon-mwaa-examples
- Local Runner https://github.com/aws/aws-mwaa-local-runner

Questions?

Thank you!

Sanchit Jain @sanchitdilipjain *(LinkedIn)*