Airflow ❤ Kubernetes
About us

Kaxil Naik
Airflow Committer & PMC member
Manager - Airflow Engineering @ Astronomer.io
Twitter: @kaxil

Jarek Potiuk
Independent Open-Source Contributor and Advisor
Airflow Committer & PMC member
Twitter: @jarekpotiuk
What the talk?

- Why Kubernetes? Why Not?
- Why Docker/Containers?
- Why Helm?
- How to make the best of it:
  - Docker/Container image
  - Helm Chart
- What’s next for Airflow & K8S?
Why Kubernetes and Containers?
Why Kubernetes and Containers?

- Kubernetes eats the world
- NoOps promise
- Isolation between components
- Standard deployment model
- Cloud and on premise
- Standard packaging/installation (Helm)
Why NOT Kubernetes?

- Complex
- Hard to debug for newcomers
- Leaky abstraction: you need to know it all
- Not easy for local development
What is Airflow’s approach?

- Airflow ♡ Kubernetes, but
- Airflow is NOT K8S native/only
  - Docker Compose/Swarm
  - Container Services
  - VMs
  - On-Prem
  - Managed services
    - (Astronomer/Composer/MWAA)
  - ...

How do you deploy Airflow?

(Airflow 2020 Survey)
Docker/Container images
Why Docker/Containers?

- Package YOUR software and dependencies together
- You can share images
- Isolation between components
- Immutable, easily deployable building blocks
- Lots of images ready-to-use
- Easy to build your own, custom images
Extending images is easy for everyone (including novice users)

Add PIP package:
FROM apache/airflow
RUN pip install --no-cache-dir vim

Add 'apt' package:
FROM apache/airflow:2.1.2
USER root
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    vim \
    && apt-get autoremove -yqq --purge \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*
USER airflow

Build:
docker build . -f Dockerfile --tag my-image:2.1.2
Customizing images (more advanced users)

```bash
git clone https://github.com/apache/airflow.git

docker build . \
  --build-arg PYTHON_BASE_IMAGE="python:3.6-slim-buster" \
  --build-arg AIRFLOW_VERSION="2.1.2" \
  --build-arg ADDITIONAL_PYTHON_DEPS="mpi4py" \
  --build-arg ADDITIONAL_DEV_APT_DEPS="libopenmpi-dev" \
  --build-arg ADDITIONAL_RUNTIME_APT_DEPS="openmpi-common" \
  --tag "my-custom-image:2.1.2"
```
Airflow Official image is mature

- Supports K8S and Quick Start Docker Compose out-of-the-box
- Enterprise ready
  - Image automatically verified
  - OpenShift-compatible
  - Customizable installation sources
  - Building in restricted environments
- Development friendly
  - Easy to inspect and debug airflow
  - Quick test features: adding admin user, upgrading DB, installing packages
Traps of convenience

● We care about our users
  ○ 1. security
  ○ 2. stability
  ○ 3. convenience

● Example: installing additional PIP packages
  ○ --env "_PIP_ADDITIONAL_REQUIREMENTS=lxml==4.6.3 charset-normalizer==1.4.1"

● NEVER, EVER use this in PRODUCTION
  ○ Slower container restarts
  ○ “leftpad” vulnerability: 3rd-party developer can bring your whole Airflow down at ANY time

● USE CUSTOM AIRFLOW IMAGES instead
Helm Chart
Why Helm?

- Package manager for Kubernetes
- Manage complex Kubernetes applications easily
  - Provides repeatable application installation
  - Serves as a single point of authority
- Easy Updates
- Simple Sharing
- Rollbacks

kube-prometheus-stack
OOG: Prometheus  REPO: prometheus-community
VERSION: 16.14.1  APP VERSION: 0.48.1
kube-prometheus-stack collects Kubernetes manifests, Grafana dashboards, and Prometheus rules combined with docum...

prometheus
OOG: Prometheus  REPO: prometheus-community
VERSION: 14.4.0  APP VERSION: 2.26.0
Prometheus is a monitoring system and time series database.

airflow
OOG: Apache Airflow  REPO: Apache Airflow
VERSION: 1.0.0  APP VERSION: 2.0.2
LICENSE: Apache-2.0
Helm chart to deploy Apache Airflow, a platform to programatically author, schedule, and monitor workflows.
What is a Helm Chart?

- Collection of YAML template files
- Files organized into a specific directory structure
- Powerful Helm template language
Airflow Helm Chart(s)!
The “Multiple Charts” problem

There were few chart options available causing confusion on which to use

1. Chart from Astronomer ([https://github.com/astronomer/airflow-chart](https://github.com/astronomer/airflow-chart))
3. **User-community** Chart ([https://github.com/airflow-helm/charts](https://github.com/airflow-helm/charts)) - previously under **Helm Stable Repo**
The “Multiple Charts” problem

- A big thanks to all the maintainers & contributors of these charts

[Link to charts] https://gph.is/g/4DL7BM2
The “Multiple Charts” problem

- Each chart had their limitations and certain features were **not good for production**
- Some of these charts had little to no testing unfortunately
- Need of an **official Apache Airflow Chart**
- An updated version of Astronomer Chart was donated to the Airflow project in 2020
- Before releasing an official version we wanted to make sure we covered:
  - Reviewed all features & decisions
  - Testing & Stability
  - Licenses & Integrity
  - Docs
- Finally the official Apache Airflow Chart released on 16 May 2021
The Official Apache Airflow Helm Chart
Official Apache Airflow Community Helm Chart

- 1.0.0 was released on 16 May 2021!
- Created by the community and for the community
- ArtifactHub: [https://artifacthub.io/packages/helm/apache-airflow/airflow](https://artifacthub.io/packages/helm/apache-airflow/airflow)
- Versioned documentation: [link](#)
Features

- All executors are supported
- Airflow version: 1.10+, 2.0+
- Database backend: PostgreSQL, MySQL
- Autoscaling for Celery Workers provided by KEDA
- PostgreSQL and PgBouncer with a battle-tested configuration
- Monitoring:
  - StatsD/Prometheus metrics for Airflow
  - Prometheus metrics for PgBouncer
  - Flower
- Automatic database migration after a new deployment
- Kerberos secure configuration
- One-command deployment for any type of executor
- DAG Deployment: git-sync, persistent volumes, baked in docker image
- and a lot more ....
Why use the official Airflow Helm Chart?

- It is the “official” Helm chart :)
- Built by the community and for the community
- Code lives with the same Airflow code
  - Tested on each merged commit to Airflow
- Uses official Airflow Docker / Container image
- Enterprise-ready & Battle-tested with Astronomer customers
- Unit tests and Integration tests
- Future-proof (including backwards compatibility)
- Use schema for validating values passed to values.yaml
Why use the official Airflow Helm Chart?

- Supports new Airflow features immediately
- Follows best-practices for Helm, Airflow and Python
  - No compromises for “convenience”
  - Focused on Production use-cases
- Versioned documentation on Airflow site: https://airflow.apache.org/docs/helm-chart/
- Stamp of Approval from the Apache Software Foundation
  - Signed releases
  - Licenses - (complies with ASF licensing policy)
  - Voting (requires at least 3 “+1” from PMC Members)
  - Helm provenance file (to verify the integrity and origin of a package)
Using the Helm Chart
Quick Start using Helm Chart

Add Airflow Helm Repo:
```
helm repo add apache-airflow https://airflow.apache.org
helm repo update
```

Create namespace and Install the chart:
```
export RELEASE_NAME=example-release
export NAMESPACE=example-namespace

kubectl create namespace $NAMESPACE
helm install $RELEASE_NAME apache-airflow/airflow \
  --namespace $NAMESPACE \
  --set 'env[0].name=AIRFLOW__CORE__LOAD_EXAMPLES,env[0].value=True'
```

Confirm Pods are up:
```
kubectl get pods --namespace $NAMESPACE
```

Port-forward Webserver:
```
kubectl port-forward svc/airflow-webserver 8080:8080 -n $NAMESPACE
```
### Every 2.0s: kubectl get pods -n example-namespace

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>example-release-flower-689d67c5fb-kg8z6</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-postgresql-0</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-redis-0</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-scheduler-6b79c98568-8qsq8</td>
<td>2/2</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-statsd-8676674f76-qh95q</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-webserver-6b4b595896-bhcw9</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
<tr>
<td>example-release-worker-0</td>
<td>2/2</td>
<td>Running</td>
<td>0</td>
<td>5m27s</td>
</tr>
</tbody>
</table>

### DAGs

<table>
<thead>
<tr>
<th>DAG</th>
<th>Owner</th>
<th>Runs</th>
<th>Schedule</th>
<th>Last Run</th>
<th>Recent Tasks</th>
<th>Actions</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>example_automate</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_branch_dag_operator.x3</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_branch_operator</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_branch_operator.s3</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_completes</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_dag_decoration</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_external_task_marker_child</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_external_task_marker_padded</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_kubernetes_executor</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_kubernetes_executor.config</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_reverted_branch_dag</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_parsing_parameters_via_test_command</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_python_operator</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_short_circuit_operator</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>example_skip_dag</td>
<td>all-fl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Links

- Source Code: https://github.com/apache/airflow/tree/main/chart
- Docs: https://airflow.apache.org/docs/helm-chart/
- ArtifactHub: https://artifacthub.io/packages/helm/apache-airflow/airflow
What’s next for Airflow & K8S