Airflow Summit 2020


Join us!
Same-same...

@ItaiYaffe, @RTeveth
... but different!

$30,000/month Savings

Open your eyes

Robust

@ItaiYaffe, @RTeveth
Migrating Airflow-based Spark jobs to K8s
the native way

Roi Teveth & Itai Yaffe
Nielsen
Introduction

Roi Teveth
- Big Data developer
- Kubernetes evangelist
- LinkedIn: Roi Teveth
- Twitter: @RTeveth

Itai Yaffe
- Tech Lead, Big Data group
- Dealing with Big Data challenges since 2012
- LinkedIn: Itai Yaffe
- Twitter: @ItaiYaffe

@ItaiYaffe, @RTeveth
What will you learn?

01. ETLs with Airflow & Spark

02. ETLs with Airflow & Spark

03. Apache Spark - on - Kubernetes

04. Airflow Integration & Contribution

@ItaiYaffe, @RTeveth
What will you learn?

How to leverage that in your organization

@ItaiYaffe, @RTeveth
Nielsen Identity Engine

- eXelate was acquired by Nielsen on March 2015
- A Data company
- Machine learning models for insights
- Targeting
- Business decisions
Nielsen Identity Engine in numbers

- ~3000 nodes/day
- ~30TB ingested/day
- $100K’s/month
- 10’s of TB ingested/day

@ItaiYaffe, @RTeveth
The challenges

Scalability

Cost Efficiency

Fault-tolerance

@ItaiYaffe, @RTeveth
Why did we need Airflow?

- Dozens of ETL workflows running around the clock
- Originally used AWS Data Pipeline for workflow management
- But we also wanted:
  - Better **visibility** of configuration and workflow
  - Better **monitoring** and **statistics**
  - Share **common configuration/code** between workflows
Why do we ❤️ Airflow?

- Met all requirements & more
- ~2 years in production
- ~40 users across 4 groups
- ~1000 DAG Runs/day
- ~20 automatic DAG deployments/day
- 6 contributions to open-source

@ItaiYaffe, @RTeveth
Remember this slide?

@ItaiYaffe, @RTeveth
Let’s take a closer look...

@ItaiYaffe, @RTeveth
Common data pipeline pattern - high-level architecture

Data Lake  Data Processing  Intermediate Storage  OLAP

1. Read input files  Spark  2. Write output files  Ingest to DB  druid

@ItaiYaffe, @RTeveth
Common data pipeline pattern - high-level architecture

1. Read input files
2. Write output files
3. Ingest to DB

@ItaiYaffe, @RTeveth
What is Spark?

- An analytics engine for large-scale data processing
- Distributed and highly scalable
- A unified framework for batch, streaming, machine learning, etc.
Basic Spark terminology

- Driver
- Executor
- Cluster manager
  - Mesos, YARN or Standalone
- Managed Spark on public clouds
  - AWS EMR, GCP Dataproc, etc.
What is EMR?

EMR is an AWS managed service to run Hadoop & Spark clusters.

Allows you to reduce costs by using Spot instances.

Charges management cost for each instance in a cluster.
Running Airflow-based Spark jobs on EMR

- EMR has official Airflow support
  - `emr_create_job_flow_operator` creates a new EMR cluster
  - `emr_add_steps_operator` adds a Spark step to the cluster
  - `emr_step_sensor` checks if the step succeeded

- Open-source, remember?
  - Allows us to fix existing components
    - EmrStepSensor fixes ([AIRFLOW-3297](AIRFLOW-3297))
  - ... As well as add new components
    - AWS Athena Sensor ([AIRFLOW-3403](AIRFLOW-3403))
    - OpenFaaS hook ([AIRFLOW-3411](AIRFLOW-3411))
Running Airflow-based Spark jobs on EMR

- EMR has official Airflow support
  - emr_create_job_flow_operator
  - emr_add_steps_operator
  - emr_step_sensor

  Creates new emr cluster
  Adds Spark step to the cluster
  Checks if the step succeeded

- Open source, remember?
  - Allows us to fix existing components
    - EmrStepSensor fixes [AIRFLOW-3297](https://issues.apache.org/jira/browse/AIRFLOW-3297)
  - ... As well as add new components
    - AWS Athena Sensor [AIRFLOW-3403](https://issues.apache.org/jira/browse/AIRFLOW-3403)
    - OpenFaaS hook [AIRFLOW-3411](https://issues.apache.org/jira/browse/AIRFLOW-3411)

@ItaiYaffe, @RTeveth
But we wanted MORE!

$\downarrow$

Visibility

Robustness
Introducing - Spark-on-Kubernetes

@ItaiYaffe, @RTeveth
Let’s explain what is Kubernetes (a.k.a K8s)

- Open source platform for running and managing containerized workloads

- Includes
  - Built-in controllers to support various workloads (e.g. micro-services)
  - Additional extensions (called “operators”) to support custom workloads

- Highly scalable
Basic Kubernetes terminology

- Control plane
- Cluster
- Worker nodes (EC2 in our case)
- Pods: group of one or more containers (such as Docker containers)
Basic Kubernetes terminology

- **kubectl** - K8s CLI

```
(base) ~ kubectl get po
NAME               READY STATUS   RESTARTS AGE
app1-67d75d7f9-bj8fh 1/1       Running 0 11s
```

```
(base) ~ kubectl run ubuntu --image ubuntu:18.06 --generator=run-pod/v1
pod/ubuntu created
```
Basic Kubernetes terminology

- The term “operator” exists both in Airflow and in Kubernetes
Basic Kubernetes terminology

- The term “operator” exists both in Airflow and in Kubernetes
- Operator
  - Represents a **single task**
  - Operators determine what is actually executed when your DAG runs
  - Example:
    - **bash-operator** - executes a bash command
Basic Kubernetes terminology

- The term “operator” exists both in Airflow and in Kubernetes

- **operator**
  - Additional extensions to Kubernetes
  - Holds the knowledge of how to manage a *specific application*
  - Example:
    - `postgres-operator` - defines and manages a PostgreSQL cluster

@ItaiYaffe, @RTeveth
Basic Kubernetes terminology

- The term "operator" exists both in Airflow and in Kubernetes

  - Operator
    - A non-core Kubernetes controller
    - Holds the knowledge of how to manage a specific application
    - Example:
      - postgres-operator - defines and manages a PostgreSQL cluster

@ItaiYaffe, @RTeveth
Kubernetes auto-scale

Phase 1: no applications are running on the cluster

```
(base) → ~ kubectl get pods
No resources found in default namespace.
```
Kubernetes auto-scale

Phase 2: application #1 starts running

@ItaiYaffe, @RTeveth
# Kubernetes auto-scale

**Phase 3: the cluster scales-up as needed**

<table>
<thead>
<tr>
<th>Control plane</th>
<th>Cluster</th>
</tr>
</thead>
</table>

- **aws**

```
(base) ~ kubectl get po
NAME                  READY STATUS  RESTARTS AGE
app1-67d75d7f9-bj8fh  1/1 Running 0 11s
```

@ItaiYaffe, @RTeveth
Kubernetes auto-scale

Phase 4: application #2 starts running

```
(base) -> ~ kubectl get po
NAME      READY STATUS    RESTARTS AGE
app1-67d75d7f9-bj8fh 1/1   Running  0     22h
app2-65d4c7f97f-n819d 0/1   Pending 0   4s
```
Kubernetes auto-scale

Phase 5: the cluster scales-up as needed

```
(base) → ~ kubectl get po
NAME       READY STATUS   RESTARTS AGE
app1-6589f85558-w8pdm  1/1 Running   0   12s
app2-5499955bb4-pplgl  1/1 Running   0   7s
```
Kubernetes auto-scale

Phase 6: applications finished running, cluster scales down

@ItaiYaffe, @RTeveth

AWS

Control plane
Cluster

```
(base) → ~ kubectl get pods
No resources found in default namespace.
```
Kubernetes in a nutshell

- A platform for running and managing containerized workloads
- Each cluster has
  - 1 control plane
  - 0..X worker nodes
  - 0..Y pods
  - 0..Z applications running concurrently
- Kubernetes operator != Airflow operator
- Automatically scales out and in

@ItaiYaffe, @RTeveth
Cool, so… Back to Spark-on-Kubernetes?
Spark-On-Kubernetes overview

- From Spark 2.3.0, K8s is supported as a **cluster manager**

- This is still **experimental**, and some features are missing
  - Better auto-scaling is expected in future Spark releases (e.g. through dynamic allocation)
Submitting a Spark application to Kubernetes - alternatives

1. Using `spark-submit` script

2. Using Spark-On-Kubernetes **Operator**
Spark-submit example - SparkPi

```bash
./bin/spark-submit
  --master k8s://https://<k8s-apiserver-host>:<k8s-apiserver-port>
  --deploy-mode cluster
  --name spark-pi
  --class org.apache.spark.examples.SparkPi
  --conf spark.executor.instances = 3
  --conf spark.kubernetes.container.image = <spark-image>
  local:///path/to/examples.jar
```

Kubernetes control plane

Kubernetes cluster

SparkPi driver

Executor 1

Executor 2

Executor 3

@ItaiYaffe, @RTeveth
Spark-On-Kubernetes operator

- A Kubernetes operator
- Extends Kubernetes API to support Spark applications
- Built by GCP as an open-source project

https://github.com/GoogleCloudPlatform/spark-on-k8s-operator

@ItaiYaffe, @RTeveth
Spark-On-Kubernetes operator example - SparkPi

Spark-pi.yaml

apiVersion: 
"sparkoperator.k8s.io/v1beta2"
kind: SparkApplication
metadata:
  name: "spark-pi"
  namespace: default
spec:
  ...
  driver:
  ...
  executor:
  ...

@ItaiYaffe, @RTeveth
# Submitting a Spark application to K8s

<table>
<thead>
<tr>
<th>Topic</th>
<th>Spark-submit</th>
<th>Spark-On-K8s operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airflow built-in integration</td>
<td>V</td>
<td>X</td>
</tr>
<tr>
<td>Customize Spark-pods</td>
<td>X</td>
<td>V</td>
</tr>
<tr>
<td>Easy access to Spark UI</td>
<td>X</td>
<td>V</td>
</tr>
<tr>
<td>Submit and view application from kubectl</td>
<td>X</td>
<td>V</td>
</tr>
</tbody>
</table>

@ItaiYaffe, @RTeveth
Integrate it with Airflow

@ItaiYaffe, @RTeveth
So... we decided to take the road less traveled
So... we decided to take the road less traveled

[AIRFLOW-6542] add spark-on-k8s operator/hook/sensor #7163

https://github.com/apache/airflow/pull/7163

@ItaiYaffe, @RTeveth
A special thanks to Airflow committers

@CzerwonyElmo (Kamil Breguła)
@kaxil (Kaxil Naik)
@AshBerlin (Ash Berlin-Taylor)
@higrys (Jarek Potiuk)

@ltaiYaffe, @RTeveth
Airflow SparkKubernetes integration

SparkKubernetes operator

SparkKubernetes sensor

KubernetesHook

@ItaiYaffe, @RTeveth
KubernetesHook

- Gets the **connection parameters** from Airflow’s Kubernetes connection
  - E.g kubeconfig, namespace or in-cluster configuration
- Creates an **API client** and allows to create and get a Kubernetes custom_resource_object
  - `SparkApplication` is of type `customResource`
SparkKubernetes operator

- Gets the SparkApplication resource definition
- Templates it using Jinja
- Sends it to the Kubernetes cluster

```yaml
apiVersion: "sparkoperator.k8s.io/v1beta2"
kind: SparkApplication
metadata:
  name: "spark-pi-{{ ds }}-{{ task_instance.try_number }}"
  namespace: default
spec:
  ...
  driver:
  ...
  executor:
  ...
```
SparkKubernetes sensor

- Pokes the SparkApplication as long as the applicationState is SUBMITTED or RUNNING
- Completes successfully if the applicationState is SUCCESS, or raises an exception if the applicationState is FAILED
What have we gained by building this integration?

- Official built-in Airflow support
- Security
  - Save Kubernetes credentials inside Airflow connection mechanism
- Portability
  - Use templated Kubernetes object so the same app can be migrated easily to Airflow and also be run manually
- Kubernetes native
  - Communicate directly with the Kubernetes API
Remember this slide?

@ItaiYaffe, @RTeveth
Let’s take a closer look again...
Common data pipeline pattern - revised

1. Read input files
2. Write output files
3. Ingest to DB

@ItaiYaffe, @RTeveth
Common data pipeline pattern - revised

1. Read input files
2. Write output files
3. Ingest to DB

Data Lake

Data Processing

Intermediate Storage

OLAP

@ItaiYaffe, @RTeveth
Benefits from migrating to Kubernetes

- **~30% cost reduction**
  - No additional cost per instance

- **Better visibility**
  - E.g automatic Prometheus configuration using Spark-on-K8s operator

- **Robustness**
  - E.g running a Spark job on multiple AZs
    (can be beneficial when using Spot instances)

@ItaiYaffe, @RTeveth
Current status and future plans

- Airflow integration merged to master
  - Will be available on Airflow 2.0
  - Planned to be included as a backport package on Airflow 1.10.11
- Extend the new KubernetesHook and possibly migrate kubernetes_pod_operator to work with it
- We’re gradually migrating our DAGs
Airflow “pro” tips

- Use Jinja templating
  - Provides the DAG developer with a set of built-in parameters and macros
- Make Airflow more secured using K8s’ role-based access control (RBAC)
Airflow “pro” tips

- Monitoring and alerting
  - Use the DAG’s `on_success_callback` and `on_failure_callback` (in `default_args`) to emit metrics to your monitoring system and trigger alerts based on those metrics.

- Leverage Airflow’s built-in components (Operator/Hook/Sensor)
  - Couldn’t find one? Build & contribute one 😊
Same-same...
... but different - for you!

- Reduce costs
- Open your eyes
- Robust

- Migrate your Spark jobs to Kubernetes
- Leverage the easy-to-use native Airflow integration we contributed

@ItaiYaffe, @RTeveth
Want to know more?

- **Women in Big Data**
  - A world-wide program that aims:
    - To inspire, connect, grow, and champion success of women in the Big Data & analytics field
  - 30 chapters and 17,000+ members world-wide
  - Everyone can join (regardless of gender), so find a chapter near you - [https://www.womeninbigdata.org/wibd-structure/](https://www.womeninbigdata.org/wibd-structure/)

- **Our Tech Blog** - [medium.com/ncm-techblog](https://medium.com/ncm-techblog)
  - Spark Dynamic Partition Inserts part 1 - [https://tinyurl.com/yd94ztz5](https://tinyurl.com/yd94ztz5)
  - Spark Dynamic Partition Inserts Part 2 - [https://tinyurl.com/y8uembml](https://tinyurl.com/y8uembml)
QUESTIONS
THANK YOU

Roi Teveth
Itai Yaffe