Elastic Compute for Batch Platform

Name: Muralidhar Sortur
Date: 13th Apr 2015
Content

- Background on Batch
- Infra need for Batch
- Available technology
- Mesos Eco system
- BAAS
- Architecture and Design
- Demo
- What Next?
- Questions
Background

• Common use case of batch processing
  – Statement Generation, bank postings, risk evaluation, credit score calculation, inventory management, portfolio optimization, Data backup, Data crunching, Information extraction etc.

• JSR-352 : Batch Application for Java platform

• Logging, check pointing & parallelization

• Start, Stop, Restart, Kill

• Require huge dedicated backend infrastructure

• ROI
“Batch” As Is

• **Infrastructure**: Dedicated VMs, Under utilized Resources, Time to Production is more

• **Operation**: Dedicated ops to address production job failures

• **Development**: No Standard way to accomplish task
Where does “Batch Infra” stand?

- **PaaS**
- **Control-M QUARTZ**
- **Batch Infra**
  - Spring Batch
  - Chronos
  - Docker
  - Spark
  - Hadoop
  - Mesos

**Commercial Solutions**
- Qu bole
- Heroku
- Iron.io
- TIBCO
Infra need for batch

Exec Environment: RHEL, Ubuntu, CentOS, Windows?

Resources: CPU, Memory, Network

Data: Hadoop, swift store, SQL, NoSQL
Infra need for batch

Exec Environment: freedom to use different flavor and version of container

Resources: Fail proof, bill as per usage

Data: Abstraction, consistency, availability, performance
Static Partitioning & Resource Utilization

Today: static partitioning

Mesos: dynamic sharing

Shared cluster
Mesos Eco System
Apache Mesos Architecture
What is BAAS?

1. Batch as a Hosted service
2. Self service
3. Fault tolerant
4. Better monitoring & notification
5. Better ways to interact with outside world (DBs, services, Hadoop, streams, messaging)
6. Flexibility in writing batch job.
7. Open source!!
Our Approach

• Abstracting user experience from specifics of technology.
• Reduce opex and capex.
• Proactive monitoring and alerting
• Self service capability
• Fault tolerant.
BAAS Deployment View
Interacting modules in Docker

- Deployer
  - Manifest
  - Execute
  - cleanup
  - libraries

- Docker

- PASS
  - mongoDB
  - LOGSTORE
  - MANIFEST STORE
  - MAVEN REPO
  - APP LIFE CYCLE MGMT
  - MYSQL DB
  - GITHUB
  - AUTHN

Infra
Create Batch Flow
Schedule and Run Batch Job
Self Service

• Creating & Registering Batch App.
• Provisioning
• Build & Deployment
• Scheduling
• Monitoring batch jobs during execution
• Email alert for failure, long running jobs.
• Killing a running batch job.
• Restarting a failed job.
• Report generation for batch job executions.
• Chain the execution of batch Jobs
Development

• Do I need to code all steps in batch?
• Is there a standard way of writing a step / task?
• How can I pass parameters from one step to another?
• Why should every one implement same task in different way?
• Where can I contribute a well defined task / step to be used in batch job?

Storage

• Is there a abstraction for batch storage needs?
• Is it possible to isolate storage and choice of storage from code?
• Should I write a separate batch app for taking backup?
Batch Reusable Infra component

0 to N Inputs

BRIC Tasklet

Inputs can be

1. hardcoded parameters
2. Previous Tasklet
3. Command line parameters

Output can be passed to other Tasklet
Storage as a Service

- Meta Data Repo
- Event manager
- BAAS

Storage Manager

Compute

Batch Job

Jcloud API

Development
“Batch” with this

• Infrastructure: NO Dedicated VMs, improved Resource utilization, Time to Production in less than a day

• Operation: Complete self service, Developer is responsible to take action if failure

• Development: Standard way to accomplish task
Resources

- http://www.wired.com/wiredenterprise/2013/03/google-borg-twitter-mesos/
- http://www.slideshare.net/tomasbart/introduction-to-apache-mesos
- https://www.docker.com/tryit/