Building FAST Data Solutions with DC/OS on Azure

Rob Bagby
Sr. Software Development Engineer
rob.bagby@microsoft.com
FAST Data and the SMACK stack

- FAST Data
  - IoT-type solutions
  - Speed of response time is crucial

- SMACK Stack
  - Spark
  - Mesos
  - Akka
  - Cassandra
  - Kafka
This Session is DEMO-Driven
Application Overview
We will illustrate enabling:

What

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

How

Containers
Orchestrator – DC/OS
DC/OS
Portworx
Workflow Solution
VAMP
Challenges containers address

- Running Cassandra / Kafka for Development
- Dependency issues
- Enabling application density
Running Cassandra / Kafka for Development

- Traditional Options
  - Install locally – very difficult
  - Shared instances
    - Step on one another
    - Not portable
- Containerized

```bash
docker run -d --name localcassandra -p 9042:9042 --network=sensor-network -v C:/data:/var/lib/cassandra cassandra:3.10

docker run -d --name kafka -p 2181:2181 -p 9092:9092 --network=sensor-network -env ADVERTISED_HOST=172.30.0.1 --env ADVERTISED_PORT=9092 spotify/kafka
```
Dependency Issues

Command Prompt:

```
C:\Users\robbag>python --version
Python 3.5.2
C:\Users\robbag>
```

Cassandra Driver 3.10 documentation:

**Supported Platforms**

Python 2.6, 2.7, 3.3, and 3.4 are supported, both CPython (the standard Python implementation) and PyPy are supported and tested.

Linux, OSX, and Windows are supported.

**Installation through pip**

pip is the suggested tool for installing packages. It will handle installing all Python dependencies for the driver at the same time as the driver itself. To install the driver:

```
pip install cassandra-driver
```
Dependency encapsulation enables density

Diagram showing layers and components:
- Dependencies Operating System Virtual Machine
- Dependencies Operating System Virtual Machine
- Container Engine
- Hypervisor
- Hardware Layer
- Applications
- Hypervisor
- Hardware Layer
Demo – Developing Locally
Rob Bagby
We will illustrate enabling:

**What**

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

**How**

- Containers
- Orchestrator
  - DC/OS
  - Portworx
- Workflow Solution
  - VAMP
Challenges orchestrators address

- Treat multiple hosts as a single unit
- Determine where containers are started
- Monitor health of containers / applications
- Orchestrate application density
- Allow you to scale services
DC/OS Superpower

(vetted) Frameworks / Services
Mesos Frameworks

- Applications that run on Mesos
- Distributed applications
  - Controller – called the “Scheduler”
  - Workers – called “Executors”
- Frameworks are “Cluster Aware”
  - Specific needs / requirements of the application
  - Cluster resources
  - External triggers
  - …
## Example – Cassandra Framework

<table>
<thead>
<tr>
<th></th>
<th>Cassandra on DC/OS</th>
<th>Cassandra Bare-Metal/VM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation</td>
<td>Automated</td>
<td>Manual</td>
</tr>
<tr>
<td>Dynamic Resource Allocation &amp; Resizing</td>
<td>Yes</td>
<td>VMs Only (Complex)</td>
</tr>
<tr>
<td>Node Scaling</td>
<td>Automated</td>
<td>Manual</td>
</tr>
<tr>
<td>Multi Datacenter Replication</td>
<td>Simple</td>
<td>Complex</td>
</tr>
<tr>
<td>Readiness Checks</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Management</td>
<td>Simple &amp; Integrated</td>
<td>Difficult &amp; Isolated</td>
</tr>
<tr>
<td>HA Node Replace/Restart</td>
<td>Automated</td>
<td>Manual</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>Simple</td>
<td>Difficult</td>
</tr>
</tbody>
</table>
Custom DC/OS clusters on Azure: acs-engine
We will illustrate enabling:

<table>
<thead>
<tr>
<th>What</th>
<th>How</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Development</td>
<td>Containers</td>
</tr>
<tr>
<td>2. Running at scale</td>
<td>Orchestration</td>
</tr>
<tr>
<td>• Big Data Solutions</td>
<td>DC/OS</td>
</tr>
<tr>
<td>• Data Persistence</td>
<td>Portworx</td>
</tr>
<tr>
<td>3. Managing at scale</td>
<td>Workflow Solution</td>
</tr>
<tr>
<td>• Autoscaling</td>
<td>VAMP</td>
</tr>
<tr>
<td>• Workflows</td>
<td></td>
</tr>
</tbody>
</table>
Options for container persistence - Azure

- Azure Files
- VMs / Ephemeral Disks
- Attached / Managed Disks
- Pooled Storage
  - GlusterFS
  - Portworx
  - ...

Challenges with Attached / Managed Disks

1. Container Rescheduling
   - disk has to move
Challenges with Attached / Managed Disks

1. Container Rescheduling
   • disk has to move
   • or rescheduled on same node
Challenges with Attached / Managed Disks

1. Container Rescheduling
   - disk has to move
   - or rescheduled on same node

2. Container/Disk Challenges
   - schedule all nodes together
   - or 1:1 container:disk relationship

3. Max number of disks/VM
Pooling Disks

Node1

1

10 GB

Node2

Pool of 1024

Disk 1 – 128GB
Disk 2 – 128GB
Disk 3 – 128GB
Disk 4 – 128GB

Disk A – 128GB
Disk B – 128GB
Disk C – 128GB
Disk D – 128GB
Portworx

- Pooled software-defined storage solution
- Storage virtualization - serves virtual volumes
- Enterprise grade
  - Backup
  - Snapshots
- Container-focused
  - Docker volume driver
  - Scheduler aware
  - Per-volume encryption
  - If the scheduler moves a container, the volume moves with it
Demo – Running at Scale

Rob Bagby
We will illustrate enabling:

<table>
<thead>
<tr>
<th>What</th>
<th>How</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Development</td>
<td>Containers</td>
</tr>
<tr>
<td>2. Running at scale</td>
<td>Orchestrator</td>
</tr>
<tr>
<td>· Big Data Solutions</td>
<td>DC/OS</td>
</tr>
<tr>
<td>· Data Persistence</td>
<td>Portworx</td>
</tr>
<tr>
<td>3. Managing at scale</td>
<td>Workflow Solution</td>
</tr>
<tr>
<td>· Autoscaling</td>
<td>VAMP</td>
</tr>
</tbody>
</table>
CANARY RELEASING AND AUTOSCALING FOR MICROSERVICE SYSTEMS

VAMP.IO
VAMP Artifacts

Static
- Breeds – Describe entities
- Blueprints – Describe topologies
- Scales – Define the size of a deployed service

Runtime
- Deployments – Running Blueprints
- Workflows – NodeJS-bases workflow services
- Gateways – Stable Routing endpoints
Demo – Managing at Scale
Rob Bagby
Session resources

- https://github.com/RobBagby/dcos-kafka-cassandra
- https://github.com/RobBagby/dcos-primer
- http://www.deveducate.com