CONTINUOUS INTEGRATION, DELIVERY, AND DEPLOYMENT WITH DOCKER AND DC/OS

Velocity 2016, Amsterdam, Day 2
http://301.sh/velocity-2016-ams-day2
Clusters:

- Go to this [GDoc](#) and I’ll assign a cluster, every person should have one
- If there are not enough clusters (21 are provisioned), let me know!
- Download your [private SSH key](#), you’ll need it to access the cluster—hint: here’s how you create the public key from the private key: `ssh-keygen -y -f ~/.ssh/ams24 > ~/.ssh/ams24.pub`
● To ssh into the DC/OS cluster: https://dcos.io/docs/1.8/administration/access-node/sshcluster/

● To find the IP address of the public agent: https://dcos.io/docs/1.8/administration/locate-public-agent/

● To validate a Marathon app spec from the command line: https://github.com/dcos-labs/marathon-validate

● For more on how to use DC/OS: https://dcos.io/docs/1.8/usage/
Hands-on...

https://github.com/mesosphere/training/blob/master/velocity-training-11-2016/DAY2/dc0s-102.md
https://github.com/mesosphere/training/blob/master/velocity-training-11-2016/DAY2/dc0s-103.md
https://github.com/mesosphere/training/blob/master/velocity-training-11-2016/DAY2/dc0s-104.md
WHAT IS CONTAINER OPERATIONS?

APPLICATION DEVELOPMENT LIFECYCLE
Source repo, Continuous Integration, artifact repo

CONTAINER ORCHESTRATION
Scheduling, Continuous Deployments

DAY 2 OPERATIONS
Debugging, Maintenance
APPLICATION DEVELOPMENT LIFECYCLE

CONTAINER OPERATIONS

- Code
- Build, Test, Package
- Publish
- Deploy
DEPLOYMENT: NOW AND THEN

Phase 0: SFTP, rsync, git pull, powered by bash scripts

Phase 1: CMS like Chef/Puppet, fleet, etc.

Phase 2: push-to-deploy, CI/CD with container orchestrator
examples

Marathon app spec, Kubernetes RC, Docker compose

Dockerfile, runc

.source artifacts

layer

runtime

packaging

.source artifacts
CI/CD TOOLING

1st generation
- Jenkins https://jenkins.io
- Bamboo https://www.atlassian.com/software/bamboo
- TeamCity https://www.jetbrains.com/teamcity/

2nd generation
- Travis https://travis-ci.org
- CircleCI https://circleci.com/
- Concourse https://concourse.ci/
Hands-on ...

CONTAINER ORCHESTRATION
Orchestration

- Organizational primitives
- Scheduling
- Health checking
- Scaling
- Upgrades
- Service discovery
# Service Discovery

<table>
<thead>
<tr>
<th>DNS-based</th>
<th>Proxy-based</th>
<th>Application-aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ easy to integrate</td>
<td>+ no port conflicts</td>
<td>+ developer fully in control</td>
</tr>
<tr>
<td>- SRV records</td>
<td>+ fast failover</td>
<td>- implementation effort</td>
</tr>
<tr>
<td>- no health checks</td>
<td>- no UDP</td>
<td>- requires distributed state management (ZK, etcd)</td>
</tr>
<tr>
<td>- TTL</td>
<td>- management of VIPs or ports</td>
<td></td>
</tr>
</tbody>
</table>

| Mesos-DNS, Consul | Minuteman, Marathon-LB | Roll-your-own, **Finagle** |
Hands-on …

https://gist.github.com/mhausenblas/7aba37703f9669576b00e973ae6a50c8
ZERO-DOWNTIME DEPLOYMENTS

rolling deployment
bring up new version and terminate old ones until all old are gone

blue-green deployment
launch a new stack and switch traffic from old to new when new instances are healthy

canary deployment
bring up a new version, start by routing a small portion of traffic to the new app, and slowly increase

T0: [0.9] [0.9] [0.9] [0.9]
T1: deployment kicks off
T2: [0.9] [0.9] [0.9] [0.9] [1.0]
T3: [0.9] [0.9] [0.9] [1.0] [1.0]
T4: [0.9] [0.9] [1.0] [1.0] [1.0]
T5: [0.9] [1.0] [1.0] [1.0] [1.0]
T6: [1.0] [1.0] [1.0] [1.0]
T7: deployment done
Hands-on ...

https://github.com/mhausenblas/zdd-lab
CONTAINER OPERATIONS

DAY 2 OPERATIONS

DEBUGGING
- Logging & Metrics
  - Emitting
  - Aggregation
  - Search
  - Reporting
  - Alerting
  - Dashboards
- Shell Access

MAINTENANCE
- Package Management
- Upgrading
- Cluster Resizing
- Application Autoscaling
- Capacity Planning
- Software Defined Networking
- Backups
- Disaster Recovery
service (app/business)

container

host & intra-host
LOGGING & MONITORING

node

- service
- container
- host

collectd

event router

storage

dashboard

alerts
LOGGING & MONITORING

- Event router
  - Riemann
  - fluentd
  - Kafka, logstash, Flume, dcos/metrics

- Storage
  - OpenTSDB/KairosDB
  - InfluxDB
  - (Local FS, HDFS)
Multi-role tools

- Sysdig
- Prometheus
- DataDog
- Ruxit
- Sensu
Hands-on ...

TRAINING FEEDBACK

- @mhausenblas
- mhausenblas.info
- michael@dcos.io

http://301.sh/velocity-2016-ams-feedback