Continuous Delivery for DC/OS with Spinnaker

Will Gorman @willgorman
Deploying software is challenging
Why continuous delivery?

- Decrease risks of deployment
- Decrease cost of deployment
- Decrease delay between feature development and availability
Deployments should be:

• Safe
• Automated
• Flexible
Safety features

- Unit tests
- Sandbox deployments
- Smoke tests
- Zero downtime deployments
- Rolling/canary deployments
Automation

• Automation is an enabler of safety
• Define and test the parts of deployments that aren’t often performed
• Deploy smaller changes more frequently
Deployment Strategies

Source: https://www.spinnaker.io/concepts/#deployment-strategies
Flexible

- Many teams with similar needs, but some variation
- Engineers naturally want to automate
- Make it easy to share and reuse
How?
Introducing Spinnaker

The open-source, multi-cloud, continuous delivery tool
Continuous Delivery Features

- Pipelines from commit to production
- CI builds
- Image baking
- Deployment strategies
- Validation
- Promotion across environments
Multi-cloud

DC/OS

Microsoft Azure

Google Cloud Platform

kubernetes

openstack

App Engine
Multi-cloud

• Spinnaker is multi-cloud but not a pure abstraction
Spinnaker Concepts

Applications, Clusters, Server Groups, and Instances
Applications and Clusters
Server Groups and Instances

![Image of server groups and instances](image-url)
Marathon applications and tasks
Create New Server Group

### Create New Server Group

<table>
<thead>
<tr>
<th>Basic Settings</th>
<th>Container Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Account</strong></td>
<td>demo-account</td>
</tr>
<tr>
<td><strong>Region</strong></td>
<td>us-west-1</td>
</tr>
<tr>
<td><strong>Group</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Stack</strong></td>
<td>dev</td>
</tr>
<tr>
<td><strong>Detail</strong></td>
<td>example</td>
</tr>
<tr>
<td><strong>CPUs</strong></td>
<td>0.5</td>
</tr>
<tr>
<td><strong>Memory (MiB)</strong></td>
<td>512</td>
</tr>
<tr>
<td><strong>Instances</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td></td>
</tr>
</tbody>
</table>

Your server group will be in the cluster: mesoscon-dev-example (new cluster)

**Container Settings**

**Container Image**: Please type your search...
Create New Server Group

- **Account**: demo-account
- **Region**: us-west-1
- **Group**: 
- **Stack**: dev
- **Detail**: example
- **CPUs**: 0.5
- **Memory (MiB)**: 512
- **Instances**: 1
- **Command**: 

Container Settings

- **Container Image**: Please type your search...
Multi-Region Deployments

<table>
<thead>
<tr>
<th>DEMO-ACCOUNT</th>
<th>mesoscon-prod</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>US-EAST-1</strong></td>
<td></td>
</tr>
<tr>
<td>V001: library/nginx:1.13.3</td>
<td>5 ▲ : 100%</td>
</tr>
<tr>
<td><strong>US-WEST-1</strong></td>
<td></td>
</tr>
<tr>
<td>V001: library/nginx:1.13.3</td>
<td>5 ▲ : 100%</td>
</tr>
</tbody>
</table>
Pipelines

Deploy to Dev

Configuration | Jenkins Build | Deploy New Version | Smoke Test | Destroy New Version | Destroy Previous Version | Fail | Prod Pipeline

Add stage

Copy an existing stage
Docker Registry Trigger

Automated Triggers

- Type: Docker Registry
- Registry Name: my-docker-registry
- Organization: willgorman
- Image: willgorman/mesoscon-spinnaker-demo
- Tag: master-\d+
- Run As User: Select Run As User
- Trigger Enabled

Executes the pipeline on an image update
DC/OS Stages Supported

- **Pipeline Stages**
  - Check preconditions
  - Deploy
  - Destroy Server Group
  - Disable Cluster
  - Find Image from Cluster
  - Jenkins, Manual Judgement, Pipeline, Resize Server Group, Run Job, Scale Down Cluster, Script, Shrink Cluster, Wait

- **Server Groups**
  - Create
  - Resize
  - Clone
  - Destroy
## Deploy Configuration

<table>
<thead>
<tr>
<th>Account</th>
<th>Cluster</th>
<th>Region</th>
<th>Strategy</th>
<th>Capacity</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO-ACCOUNT</td>
<td>mesoscon-prod</td>
<td>us-east-1</td>
<td>[none]</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>DEMO-ACCOUNT</td>
<td>mesoscon-prod</td>
<td>us-west-1</td>
<td>[none]</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

## Execution Options

- [ ] **If stage fails**
  - halt the entire pipeline
  - halt this branch of the pipeline
  - halt this branch and fail the pipeline once other branches complete
  - ignore the failure
Destroy Server Group

Stage Name: Destroy New Version

Depends On: Smoke Test

Destroy Server Group Configuration

- Account: demo-account
- Regions: us-east-1, us-west-1
- Cluster: us-east-1
- Target: Newest Server Group
Run Job

Execute a Metronome job as a step in a pipeline.

Write JSON or property file output to the Mesos sandbox to create context for later steps
Pipeline Expressions

- Spring Expression Language
- Works almost anywhere
- Generate attributes of pipelines at run-time
- Evaluate to test for pipeline branching conditions
Pipeline Expressions

Configure Deployment Cluster

Parameters
- Name: custom_label
- Description: The value of our custom label
- Default Value: 
- Show Options: 

Environment Variables
- Labels
  - Key: CUSTOM_LABEL
  - Value: ${parameters.custom_label}

Health Checks
- Perform health checks on running tasks to determine if they are operating as expected.
Deployment Safeguards
Execution Windows

Restrict execution to specific time windows

Days of the Week (No days selected implies execution on any day if triggered)

<table>
<thead>
<tr>
<th>Sun</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
<th>All</th>
<th>None</th>
<th>Weekdays</th>
<th>Weekend</th>
</tr>
</thead>
</table>

Time of Day

| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

This stage will only run within the following windows (all times in PDT):

From 07:00 to 10:00

Add an execution window

Source: https://www.spinnaker.io/guides/tutorials/codelabs/safe-deployments/
Execution Windows

Source: https://www.spinnaker.io/guides/tutorials/codelabs/safe-deployments/
Traffic Guards

Traffic Guards allow you to specify critical clusters that should always have active instances. If a user or process tries to delete, disable, or resize the server group, Spinnaker will verify the operation will not leave the cluster with no active instances, and fail the operation if it would.

<table>
<thead>
<tr>
<th>Account</th>
<th>Region</th>
<th>Stack</th>
<th>Detail</th>
<th>Matched clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>dcos-account</td>
<td>us-west-2</td>
<td>dev</td>
<td></td>
<td>myapp-dev in us-west-2-eng</td>
</tr>
</tbody>
</table>
Chaos Monkey

Source: https://github.com/Netflix/Chaos Monkey Configuring behaviour via Spinnaker
Deployment timeouts


```json
"docker": {
  "image": "my-docker-registry",
  "imageName": "index.docker.io/library/nginx:0.8",
  "registry": "index.docker.io",
  "repository": "library/nginx",
}
```
Destroy Server Group Configuration

- **Account**: demo-account
- **Regions**: us-west-1
- **Cluster**: mesoscon
- **Target**: Newest Server Group

Execution Options

- **If stage fails**
  - Halt the entire pipeline
  - Halt this branch of the pipeline
  - Halt this branch and fail the pipeline once other branches complete
  - Ignore the failure

- **Restrict execution to specific time windows**

- **Conditional on Expression**
  ```java
  ${#stage('Deploy in us-west-1')['context']['exception']['exceptionType'].toString() == 'TimeoutException'}
  ```
STAGE DETAILS: CLEANUP AFTER TIMEOUT

Duration: 00:21

<table>
<thead>
<tr>
<th>Step</th>
<th>Started</th>
<th>Duration</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destroy Server Group: mesoscon-v009 (us-west-1)</td>
<td>2017-08-16 23:39:32 EDT</td>
<td>00:21</td>
<td>SUCCEEDED</td>
</tr>
</tbody>
</table>
Incremental Automation

- Problem: Not everything in our release process is completely automated yet.
Manual Judgment to Rollback

<table>
<thead>
<tr>
<th>Account</th>
<th>Action</th>
<th>Trigger Status</th>
<th>Configure</th>
<th>Rollback Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO-ACCOUNT</td>
<td>Deploy to Dev</td>
<td>enabled</td>
<td>Configure</td>
<td>Start Manual Execution</td>
</tr>
</tbody>
</table>

No executions found matching the selected filters.

<table>
<thead>
<tr>
<th>Account</th>
<th>Action</th>
<th>Trigger Status</th>
<th>Configure</th>
<th>Rollback Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual Judgment</td>
<td></td>
<td>enabled</td>
<td>Configure</td>
<td>Starting Manual Execution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Account</th>
<th>Action</th>
<th>Trigger Status</th>
<th>Configure</th>
<th>Rollback Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO-ACCOUNT</td>
<td>Demo</td>
<td>enabled</td>
<td>Configure</td>
<td>Start Manual Execution</td>
</tr>
</tbody>
</table>
Configuration Change Triggers

- Problem: Spinnaker pipeline configuration isn’t the primary source for our application configuration
- Prefer configuration to be stored in version control or generated to reduce duplication across applications
Configuration Change Triggers

Edit Pipeline JSON

The JSON below represents the pipeline configuration in its persisted state. You can change it here or use this to quickly copy:

```json
{
  "desiredCapacity": 5,
  "disk": 0,
  "docker": {
    "forceFullImage": false,
    "image": [
      "imageId": "index.docker.io/willgorman/mesos-son-spinnaker-demo:latest",
      "registry": "index.docker.io",
      "repository": "willgorman/mesos-son-spinnaker-demo",
      "tag": "latest"
    ],
    "network": "BRIDGE",
    "parameters": [],
    "privileged": false
  },
  "dockerVolumes": [],
  "env": ["${#jsonPrefix}:'https://raw.githubusercontent.com/foo/bar/master/config.json'}",
  "events": [],
  "externalVolumes": [],
  "fetch": [],
  "forceDeployment": false,
  "global": {
    "dockerVolumeMounts": [],
    "globalVolumes": [],
    "globalVmes": [],
    "service": {
      "name": "spinnaker",
      "namespace": "default",
      "parameters": [],
      "privileged": false
    }
  },
  "imagePullPolicy": "IfNotPresent",
  "labels": [],
  "network": "BRIDGE",
  "networkMode": "bridge",
  "parameters": [],
  "privileged": false
}
```
Load Balancer

- Problem: Marathon-LB doesn’t support Spinnaker load balancer operations
Traefik

Source: https://traefik.io/
Traefik

HAPROXY_0_VHOST: foo.example.com
traefik.backend: foo

HAPROXY_0_VHOST: foo.example.com
traefik.backend: foo
Problem: Traefik still discovers routing rules through labels, which can’t change after an instance is started

Solution – DC/OS 1.10 programmable Edge-LB package (beta)
Enterprise and Open Source DC/OS

- Only supports DC/OS Enterprise authentication methods currently
- DC/OS Open Source works with authentication disabled
Thanks!

- https://www.spinnaker.io/
- Slack: https://join.spinnaker.io
- http://careers.cerner.com
- http://engineering.cerner.com