Lab 5 - Launch a Marathon Service

Lab Completion Time: 10 -15 minutes.

Step 1

SSH in to your bootstrap node with the **centos**/**me$o$ph3r3_** credentials:

```
$ ssh centos@<bootstrap_node_public_IP>
```

Step 2

Create a JSON file in the `~/apps` directory using your favorite text editor called `basic.json` with the following contents:

basic.json

```
{
  "id": "/hello-world",
  "cmd": "while [ true ]; do echo -n 'Hello Marathon: '; date; sleep 5; done",
  "cpus": 0.1,
  "mem": 10.0,
  "instances": 2
}
```

Step 3

Use this file to spin up a Marathon job:

```
$ dcos marathon app add ~/apps/basic.json
```

Step 4

List currently active Marathon services:

```
$ dcos marathon app list
ID             MEM  CPUS  TASKS  HEALTH  DEPLOYMENT  WAITING  CONTAINER  CMD
/hello-world    10  0.1    2/2    N/A       ---      False       N/A     while [ true ];
```

In the output above, the column labeled `TASKS` displays `2/2` once both requested container instances arrive to a running state.

Step 5

List all currently active containers (tasks in Mesos terminology):

```
$ dcos task
NAME         HOST        USER  STATE  ID                      MESOS ID     REGION
hello-world  10.0.0.105  root    R    hello-world.a8d8570f  690b7178-S0  aws/us-west-2
hello-world  10.0.0.224  root    R    hello-world.a8d808ee  690b7178-S1  aws/us-west-2
```

Copy one of the `ID` columns and use it in the command below:

```
$ dcos task log --follow <TASK_ID>
```

You should now be following the output of what is being written to `stdout` in the container. Press `Ctrl-C` on your keyboard to get back to your command prompt.

To follow the output of `stderr`, simply append the filename to the end of the command:

```
$ dcos task log --follow <TASK_ID> stderr
```

Step 7

Run this command to scale the service down to 0 instances:

```
$ dcos marathon app stop hello-world
Created deployment 9f326ea1-fd7e-4028-9bca-64ac271ef95a
```

Step 8

Run the list command to see what is running on your cluster:

```
$ dcos marathon app list
ID             MEM  CPUS  TASKS  HEALTH  DEPLOYMENT  WAITING  CONTAINER  CMD
/hello-world   10  0.1    0/0    ---       ---      False       N/A     while [ true ];
```

Notice the values under the TASKS column are now 0/0, indicating we have 0 running instances and 0 requested instances.

Step 9

Start up ten instances of our stopped service:

```
$ dcos marathon app start hello-world 10
```

Run the list command to see we now have ten instances running:

```
$ dcos marathon app list
ID             MEM  CPUS  TASKS  HEALTH  DEPLOYMENT  WAITING  CONTAINER  CMD
/hello-world   10  0.1   10/10   ---       ---      False       N/A     while [ true ];
```

Step 10

Finally run this command to delete this service entirely:

```
$ dcos marathon app remove hello-world
```

You can then verify that `/hello-world` is no longer defined on the cluster:

```
$ dcos marathon app list
```

Back to top