(../..)

In this lab you will deploy Redis to BOSH-lite using the CloudFoundry community's Redis deployment.

Setup

Make sure your BOSH cli is pointed at the `articulate.yml` deployment manifest. You will modify and redeploy this manifest to add Redis to that deployment.

Install spiff (https://github.com/cloudfoundry-incubator/spiff/releases)on your computer in your `PATH`.

Upload Necessary Releases

We will be using the CloudFoundry Community Redis BOSH Release (https://github.com/cloudfoundry-community/redis-boshrelease).

Upload the release to your BOSH director, then clone the release repository. For the specific commands, see the usage instructions from the Redis Release Repo README (https://github.com/cloudfoundry-community/redis-boshrelease#usage).

The repository comes with scripts to help us build a manifest: `templates/make_manifest`. Run the script, specifying that it should build a manifest that uses the `warden` CPI (the CPI BOSH-lite uses).

```
templates/make_manifest warden
bosh deployment tmp/redis-warden-manifest.yml
bosh deploy
```

What was deployed? Use `bosh vms` to find out.

```
bosh vms
```

```
+----------------------------------------------------------------+---------+-----+----------+------------+
| VM                                                             | State   | AZ  | VM Type  | IPs        |
+----------------------------------------------------------------+---------+-----+----------+------------+
| redis_leader_z1/0 (21604ece-1222-47b6-8858-ddf3f6e7bf75)       | running | n/a | small_z1 | 10.244.2.2 |
| redis_test_slave_z1/0 (78f07ebb-7305-419e-8fa2-a55438865794) | running | n/a | small_z1 | 10.244.2.6 |
| redis_z1/0 (5f2b37d2-353f-48fd-b3a9-5acad78f0576)              | running | n/a | small_z1 | 10.244.1.2 |
| redis_z1/1 (e105e60c-05b2-4280-9b01-8ce029b2d354)              | running | n/a | small_z1 | 10.244.1.6 |
+----------------------------------------------------------------+---------+-----+----------+------------+
```

4 VMs were deployed: a single leader, two slave nodes, and a slave node for test.

Run the acceptance tests.

```
bosh run errand acceptance-tests
```

Check that your deployment worked by interacting with redis.

If you do not have the `redis-cli` installed, install it with Homebrew.

```
[ ! `which redis-cli` ] && brew install redis
```

Run `redis-cli` to launch an interactive prompt.

```
redis-cli -h 10.244.2.2
```

`AUTH` with the server. Use password `red!s`.

```
AUTH red!s
```

Set the string `bar` at the key `foo`. Check that the key is set.

```
set foo "bar"
```

```
get foo
```

Exit the session using `exit`.

Clean Up the Manifest and Redeploy

Look at the generated redis manifest.

```
cat tmp/redis-warden-manifest.yml
```

Note the `networks` configuration block. The complex networking configuration is an artifact of older versions of BOSH-lite. Newer versions of BOSH-lite define a consecutive IP range.

The generated manifest also uses the older, `v1` manifest syntax that you saw in the earlier `articulate.yml`.

Below is a simplified version of the Redis deployment manifest, migrated to a syntax that more closely resembles the v2 `articulate` manifest.

```
name: redis-warden
director_uuid: <%= `bosh status --uuid` %>

releases:
- name: redis
  version: latest

instance_groups:
  - name: redis_leader_z1
    instances: 1
    resource_pool: small_z1
    persistent_disk: 4096
    networks:
      - name: redis1
        static_ips:
        - 10.244.2.2
    jobs:
      - release: redis
        name: redis
        properties:
          redis:
            password: red!s

  - name: redis_z1
    instances: 2
    resource_pool: small_z1
    persistent_disk: 4096
    networks:
      - name: redis1
    update:
      canaries: 10
    jobs:
      - release: redis
        name: redis
        properties:
          redis:
            master: 10.244.2.2
            password: red!s

  - name: redis_test_slave_z1
    instances: 1
    resource_pool: small_z1
    persistent_disk: 4096
    networks:
      - name: redis1
        static_ips:
        - 10.244.2.6
    jobs:
      - release: redis
        name: redis
        properties:
          redis:
            master: 10.244.2.2
            password: red!s

  - name: acceptance-tests
    instances: 1
    lifecycle: errand
    resource_pool: small_z1
    networks:
      - name: redis1
    jobs:
      - release: redis
        name: acceptance-tests
        properties:
          redis:
            master: 10.244.2.2
            password: red!s
            slave: 10.244.2.6

networks:
  - name: redis1
    type: manual
    subnets:
    - range: 10.244.1.0/24
      gateway: 10.244.1.1
      static: []
    - range: 10.244.2.0/24
      gateway: 10.244.2.1
      static: [10.244.2.2, 10.244.2.6]


resource_pools:
- cloud_properties:
    name: random
  name: small_z1
  network: redis1
  size: 5
  stemcell:
    name: bosh-warden-boshlite-ubuntu-trusty-go_agent
    version: latest

compilation:
  cloud_properties:
    name: random
  network: redis1
  reuse_compilation_vms: true
  workers: 6

update:
  canaries: 1
  canary_watch_time: 1000-100000
  max_in_flight: 50
  update_watch_time: 1000-100000
```

Deploy redis from the new manifest. Copy the above YAML and paste it into a file called `redis-warden-v2.yml`. Delete your previous `redis-warden` deployment. Deploy the `v2` manifest.

```
bosh delete deployment redis-warden
bosh deployment redis-warden-v2.yml
bosh deploy
```

Verify the deployment worked by running the Redis Bosh release acceptance tests.

```
bosh run errand acceptance-tests
```

Integrating Redis into the Articulate Deployment

Assume that your application (articulate) depends on Redis. In that case, it would be more efficient to combine the deployments.

As a final challenge, move the relevant pieces from `redis-warden-v2.yml` into `articulate.yml` and `cloud-config.yml` so you can deploy both redis and articulate at once.

Reminder: the Cloud Configuration will take the *networks*, *compilation*, and *resource_pools* sections, while the `articulate.yml` deployment manifest should take the *releases*, *instance_groups*, and *update* sections.

Delete the `redis-warden` deployment before trying to deploy your articulate + redis deployment.

```
bosh delete deployment redis-warden
```

Assessment

When you have finished merging the two manifests, rerun the articulate deployment. If the deployment succeeded, when you run `bosh vms`, it should look like the following.

```
bosh vms
```

```
Acting as user 'admin' on 'Bosh Lite Director'
Deployment 'articulate'

Director task 20

Task 20 done

+------------------------+---------+------------+------------+
| VM                     | State   | VM Type    | IPs        |
+------------------------+---------+------------+------------+
| articulate/0           | running | articulate | 10.244.9.5 |
| redis_leader_z1/0      | running | redis      | 10.244.2.2 |
| redis_test_slave_z1/0  | running | redis      | 10.244.2.6 |
| redis_z1/0             | running | redis      | 10.244.1.2 |
| redis_z1/1             | running | redis      | 10.244.1.3 |
+------------------------+---------+------------+------------+

VMs total: 5
```

When you are ready for assessment, show your instructor. They will verify that:

- you can curl the Articulate app
- the Redis acceptance tests run and pass
- you can log in to the interactive redis client and put and get keys