QuickStart: Mirantis Container Cloud on OpenStack

version latest

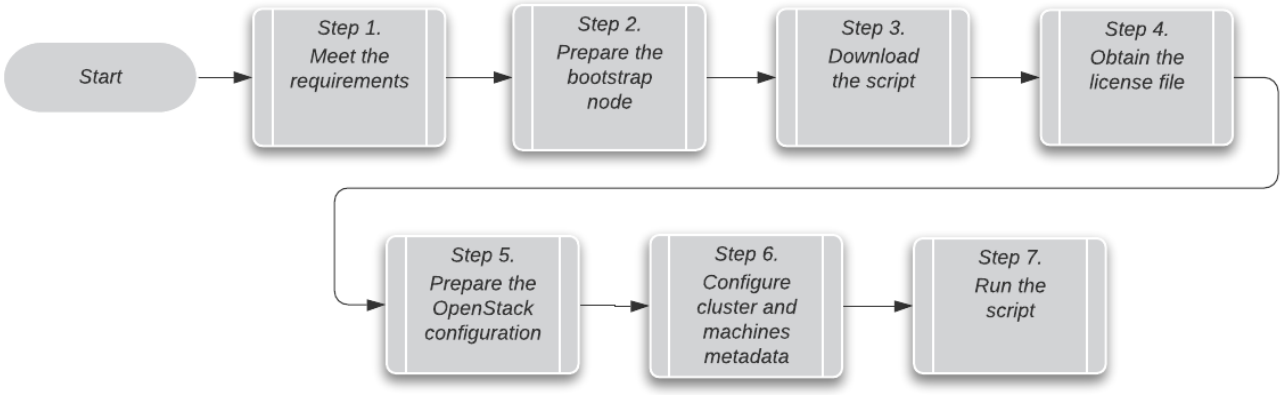
Contents

Copyright notice

Introduction

Using this QuickStart tutorial, you can deploy a Mirantis Container Cloud OpenStack-based management cluster containing 3 control plane nodes. This cluster will run the public API and the web UI. Using the Container Cloud web UI, you can deploy managed clusters that run Mirantis Kubernetes Engine.

The following diagram illustrates the deployment overview of a Container Cloud OpenStack-based management cluster:



Before you begin

Before you start the cluster deployment, verify that your system meets the following minimum hardware and software requirements for an OpenStack-based management cluster.

> Note
>
> For the bootstrap node, you can use any local machine running Ubuntu 18.04 with the following resources:
>
> - 2 vCPUs
> - 4 GB of RAM
> - 5 GB of available storage

Minimum hardware requirements for a management cluster

Resource	Requirement
# of nodes	4 (3 for HA + 1 for Bastion)
# of vCPUs	25 (8 per node + 1 for Bastion)
RAM in GB	49 (16 per node + 1 for Bastion)
Storage in GB	360 (120 per node)
OpenStack version	Queens
Obligatory OpenStack components	Octavia, Cinder, OVS
# of Cinder volumes	7 (total 110 GB)
# of load balancers	10
# of floating IPs	11

Minimum software requirements for a management cluster

Software	Version
Operating system distribution	Ubuntu 18.04
Docker	Current version available for Ubuntu 18.04

Prepare the bootstrap node

1. Log in to any personal computer or VM running Ubuntu 18.04 that you will be using as the bootstrap node.
2. If you use a newly created VM, run:

   ```
   sudo apt-get update
   ```

3. Install the current Docker version available for Ubuntu 18.04:

   ```
   sudo apt install docker.io
   ```

4. Grant your USER access to the Docker daemon:

   ```
   sudo usermod -aG docker $USER
   ```

5. Log off and log in again to the bootstrap node to apply the changes.
6. Verify that Docker is configured correctly and has access to the Container Cloud CDN. For example:

   ```
   docker run --rm alpine sh -c "apk add --no-cache curl; \
   curl https://binary.mirantis.com"
   ```

   The system output must not contain error records.

Download the bootstrap script

1. On the bootstrap node, download and run the Container Cloud bootstrap script:

```
wget https://binary.mirantis.com/releases/get_container_cloud.sh
chmod 0755 get_container_cloud.sh
./get_container_cloud.sh
```

2. Change the directory to the kaas-bootstrap folder created by the script.

Obtain the Mirantis license

1. Create a user account at mirantis.com.
2. Log in to your account and download the mirantis.lic license file.
3. Save the license file as mirantis.lic under the kaas-bootstrap directory on the bootstrap node.

Prepare the OpenStack configuration

1. Log in to the OpenStack Horizon.
2. In the Project section, select API Access.
3. In the right-side drop-down menu Download OpenStack RC File, select OpenStack clouds.yaml File.
4. Save the downloaded clouds.yaml file in the kaas-bootstrap folder created by the get_container_cloud.sh script.
5. In clouds.yaml, add the password field with your OpenStack password under the clouds/openstack/auth section.

   Example:

   ```
   clouds:
     openstack:
       auth:
         auth_url: https://auth.openstack.example.com:5000/v3
         username: your_username
         password: your_secret_password
         project_id: your_project_id
         user_domain_name: your_user_domain_name
       region_name: RegionOne
       interface: public
       identity_api_version: 3
   ```

6. Verify access to the target cloud endpoint from Docker. For example:

   ```
   docker run --rm alpine sh -c "apk add --no-cache curl; \
   curl https://auth.openstack.example.com:5000/v3"
   ```

   The system output must contain no error records.

Configure the cluster and machines metadata

1. In templates/machines.yaml.template, modify the spec:providerSpec:value section for 3 control plane nodes marked with the cluster.sigs.k8s.io/control-plane label by substituting the flavor and image parameters with the corresponding values of the control plane nodes in the related OpenStack cluster. For example:

```
spec: &cp_spec
  providerSpec:
    value:
      apiVersion: "openstackproviderconfig.k8s.io/v1alpha1"
      kind: "OpenstackMachineProviderSpec"
      flavor: kaas.minimal
      image: bionic-server-cloudimg-amd64-20190612
```

> Note
>
> The flavor parameter value provided in the example above is cloud-specific and must meet the Container Cloud requirements.

   Also, modify other parameters as required.

2. Modify the templates/cluster.yaml.template parameters to fit your deployment. For example, add the corresponding values for cidrBlocks in the spec::clusterNetwork::services section.

Finalize the bootstrap

1. Run the bootstrap script:

```
./bootstrap.sh all
```

2. When the bootstrap is complete, collect and save the following management cluster details in a secure location:

   - The kubeconfig file located in the same directory as the bootstrap script. This file contains the admin credentials for the management cluster.
   - The private ssh_key for access to the management cluster nodes that is located in the same directory as the bootstrap script.
   - The URL and credentials for the Container Cloud web UI. The system outputs these details when the bootstrap completes.
   - The StackLight endpoints. For details, see Operations Guide: Access StackLight web UIs.
   - The Keycloak URL that the system outputs when the bootstrap completes. The admin password for Keycloak is located in kaas-bootstrap/passwords.yml along with other IAM passwords.

> Note
>
> When the bootstrap is complete, the bootstrap cluster resources are freed up.

What’s next

Using your newly deployed management cluster, you can:

- Deploy an additional OpenStack-based regional cluster to operate managed clusters of several configurations within a single Container Cloud deployment in parallel.
- Create and operate OpenStack-based managed clusters. Before that, verify that your planned cluster meets the requirements for managed clusters.
- Configure an external identity provider for IAM.
- Attach an existing Mirantis Kubernetes Engine cluster.

For details about all Container Cloud features, refer to the full set of Container Cloud documentation.

