brant

2017-08-07始 周记录

Add Image & Create Instance, etc

We need a Ubuntu VM in OpenStack to deploy JuJu. Below are some notes (`Select` below means you should take down the `ID` of what you select).

- Download image and add it

```
wget https://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-disk1.img
# use glance to manage image
glance image-create --name ubuntu_xenial --file ./xenial-server-cloudimg-amd64-disk1.img --disk-format qcow2 --container-format bare --progress
```

- Select flavor

```
# select one flavor
openstack flavor list
# maybe you want create a new flavor
nova flavor-create --ephemeral 0 --is-public True m1.small 1 2048 20 1
nova flavor-create --ephemeral 0 --is-public True m1.medium 2 4096 40 2
```

- Select security group

```
openstack security group list
```

- Select key-pair

```
nova keypair-list
# if no keypair exists or you want a new one
nova keypair-add xenial_test > xenial_test.pem
```

- Select network

Exactly, if vlan or local network exists you do not need to select it and by default nova may automatically select one for the new VM. But if you have no network except the `external` network, you need to create one.

```
neutron net-list
neutron net-create test_hou
neutron subnet-create test_hou 192.168.22.0/24 --name test_hou_sub
```

- Create instance

```
nova boot --flavor 1 --image 9bbd4e7f-44cc-47fe-a724-1b604f9e1f09 --key-name
xenial_test --security-groups default --description "test for ubuntu16.04
xenial" test_zhian
```

- Add floating-ip

You should specify a floating-ip for your VM so it can be accessible from outside.

```
openstack floating ip create external
nova floating-ip-associate test_zhian 192.168.37.200
```

- Route

You need to add routeing for your vlan:

```
neutron router-create router_test_hou
neutron router-gateway-set ea181883-3a3d-4d28-b528-75ae34c33c13 4fa91a9c-1708-
4bbb-b873-ebd63744f11b
neutron router-interface-add ea181883-3a3d-4d28-b528-75ae34c33c13 edc959a3-
5964-452d-b781-b92fd7cffb07
neutron port-create test_hou
```

Now you have one VM done! But `ping` or `SSH` can not succeed because of the `default` security group. So you may add some rules:

```
openstack security group list
openstack security group rule create d8f27130-7cb2-4282-bfb7-0c60d14cd643 --
protocol tcp --dst-port 22:22 --src-ip 0.0.0.0/0
openstack security group rule create --protocol icmp d8f27130-7cb2-4282-bfb7-
0c60d14cd643
# if you want more ports to be accessible (you can also use `openstack ...`)
nova secgroup-add-rule d8f27130-7cb2-4282-bfb7-0c60d14cd643 tcp 1 65535
0.0.0.0/0
```

Now `ping` and `ssh` are OK. For `ssh` you should do as:

```
ssh -i ./xenial_test.pem ubuntu@192.168.37.200
```

Attention! If the attributes of the file `xenial_test.pem` are undue `ssh` may fail. Then you need to `chmod 700 ./xenial_test.pem` to modify the attributes and try again.

- NAT

To install JuJu, our VM must be connected to the Internet. The network arthitecture is:

```
192.168.22.10   <---> VM in vlan IP
192.168.37.200  <---> VM in external network IP
192.168.122.1   <---> VM's gateway; Real server's one interface
192.168.24.2    <---> Real server's interface to Internet
```

Configuration on VM:

```
vim /etc/resolv.conf
# add
nameserver 192.168.122.1
```

Configuration on the real server:

```
# turn on forward function
sysctl net.ipv4.ip_forward=1
# add rules in iptables
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A POSTROUTING -s 192.168.37.0/24 -j SNAT --to 192.168.24.2
iptables -t nat -A POSTROUTING -s 192.168.122.0/24 -j SNAT --to 192.168.24.2
```

Now the VM can visit the Internet!

REFERENCE

http://www.cnblogs.com/CloudMan6/p/5393376.html

https://colinleefish.gitbooks.io/openstack-end-user-guide-simplified-chinese/content/openstack_command_line_clients/cli_nova_configure_access_security_for_instances.html
https://colinleefish.gitbooks.io/openstack-end-user-guide-simplified-chinese/content/openstack_command_line_clients/cli_launch_instances.html
https://colinleefish.gitbooks.io/openstack-end-user-guide-simplified-chinese/content/openstack_command_line_clients/cli_create_and_manage_networks.html

Deploy JuJu

Now we can deploy JuJu.

We came across a trouble that after JuJu managed to create a `juju controller` node it wants to install packages using command like `apt-get install xxx` but there's no effective DNS in `resolv.conf`. If you do not solve this trouble, you will fail to deploy JuJu.
Our solution is to use `openstack dashboard` (see notes before for how to use `socat` to `forward`) and add DNS `8.8.8.8` and `192.168.122.1` (one IP belongs to the real server) in the management of network.

Go ahead. Firstly, `ssh` onto the VM and execute

```
juju --debug --show-log  bootstrap mycloud mycloud --config image-metadata-
url=http://192.168.22.10/images --config network=test_hou --config use-
floating-ip=True &> my_juju_log
```

Then open a new session and see the progress:

```
tail -f ./my_juju_log
```

If lucky, JuJu will be deployed successfully. Whereas we came across another trouble: after some failures the compute-node-0 can not be `ping`ed or `ssh`ed. Our solution is turn down that node and turn on it again in the management of VM in `dashboard`.

May the force be with you :)

Congress for Plan B

In plan B we decide to let congress play the role to instruct Nova to evacuate the VMs on the compute nodes which should have been down. So we create policy like this:

```
openstack congress policy rule create --name evacuate_vms
classification  'execute[nova:servers.evacuate(vmid)] :- host_down(host),
active_instance_in_host(vmid, host)'
```

https://developer.openstack.org/api-ref/compute/#evacuate-server-evacuate-action