OpenStack Tutorial

Shihabur R. Chowdhury
CS 856 - Fall 2015
University of Waterloo

Environment Setup

- Download the VirtualBox image from here
- Open VirtualBox and go to
  - **File > Import Appliance**
- Choose the just downloaded virtual appliance file and click **Next**
- Set at least **4096MB** of memory and **1CPU** in the Appliance Settings window and click **Import**

Environment Setup

- VM credentials
  - username: **openstack**
  - password: **openstackpass**
- OpenStack credentials
  - username: **admin**
  - password: **adminpass**

DevStack

- A collection of scripts to run OpenStack on a single machine
  - For development and demo purposes
- Download devstack from github
  - `git clone` https://git.openstack.org/openstack-dev/devstack
- Put the configuration in local.conf
- Run the **`stack.sh`** script inside devstack directory.

DevStack Configuration

- Start the file with
  - `[[local|localrc]]`
- A bunch of password configurations
  - `ADMIN_PASSWORD=adminpass`
  - `DATABASE_PASSWORD=$ADMIN_PASSWORD`
  - `RABBIT_PASSWORD=$ADMIN_PASSWORD`
  - `SERVICE_PASSWORD=$ADMIN_PASSWORD`
  - `SERVICE_TOKEN=servicetoken`

DevStack Configuration (contd…)

- Network configuration
  - **FLOATING_RANGE=10.0.3.0/27**
  - **PUBLIC_NETWORK_GATEWAY=10.0.3.1**
  - **HOST_IP=10.0.2.15**

DevStack Configuration (contd…)

- Disable nova network
  - `disable nova-net`
- Enable neutron networking
  - `enable_service q-svc`
  - `enable_service q-agt`
  - `enable_service q-dhcp`
  - `enable_service q-meta`
  - `enable_service q-l3`
  - `enable_service q-lbaas`

DevStack Configuration (contd…)

- Neutron configuration
  - `Q_USE_SECGROUP=True`
  - `ENABLE_TENANT_VLANS=True`
  - `TENANT_VLAN_RANGE=1000:1999`
  - `PHYSICAL_NETWORK=default`
  - `FLAT_INTERFACE=eth0`
  - `PUBLIC_INTERFACE=eth0`

Environment Setup

- **`stack.sh`** takes quite a while to finish. It has been already run for you. Run the **`rejoin-stack.sh`** script to finish configuring the environment
  - **`~/devstack/rejoin-stack.sh`**
- Press **`Ctrl-a`** then press **`d`** to detach the screen session

General Tips

- Every component has detailed help
  - `nova help`
- Parameters of a particular command can be found in similar way
  - `nova help boot`
- Almost every component has a *-list command to show list of *s
  - `glance image-list`
  - `neutron subnet-list`

What services are running ?

- Show the list of currently available services
  - `keystone service-list`
- List of URLs for accessing REST API of the services
  - `keystone endpoint-list`
- Show everything
  - `keystone catalog`

User Management

- View list of users
  - `keystone user-list`
- Add a new user
  - `keystone user-create --name bob --pass bobpass`
- Add ‘bob’ to tenant ‘admin’
  - `keystone user-role-add --user bob --role _member_ --tenant admin`

Quick Exercise

- How to change password of the user bob to ‘nobob’ ?

Images and Flavors

- Show available images
  - `glance image-list`
- Add a VM image to glance
  - `glance image-create --name tinycore-x86 --disk-format qcow2 --container-format bare --file ~/images/base_tc.qcow2`
- List available flavors
  - `nova flavor-list`
- Create a new flavor
  - `nova flavor-create <name> <id> <ram> <disk> <vcpu(s)>`
  - `nova flavor-create m1.verytiny 6 64 1 1`

Network Management

- Create a private network
  - **`neutron net-create  private-ipv4-net`**
  - Note the ID and export it as **`NETWORK_ID`**
- Create a subnet under ‘private-net’
  - **`neutron subnet-create --name private-ipv4-subnet $NETWORK_ID 172.16.0.0/24 --gateway 172.16.0.1 --dns-nameserver 8.8.8.8`**
- Show details of a subnet
  - **`neutron subnet-show $SUBNET_ID`**

Network Management

- Show list of routers
  - `neutron router-list`
- Create a router named ‘border’
  - `neutron router-create border`
- Add the private and public networks to one of ‘border’’s interfaces
  - `neutron router-interface-add border private-ipv4-subnet`
- Set a gateway interface for the router
  - `neutron router-gateway-set border public`

Security Groups

- Show the current tenant's security groups
  - `neutron security-group-list`
- Create a new security group
  - `neutron security-group-create ftp --description "Allow ftp traffic"`
- Add rule to a security group
  - `neutron security-group-rule-create --direction ingress --protocol tcp --port_range_min 21 --port_range_max 21 ftp`

Security Groups

- List all security rules
  - `neutron security-group-rule-list`
- Delete a security group rule
  - `neutron security-group-rule-delete $RULE_ID`
- Delete a security group
  - `neutron security-group-delete $GROUP_ID`

Security Group Exercise

- Create a security group that allows incoming UDP traffic from ports 10000 to 11000.

Virtual Machines

- Boot a virtual machine from an existing image
  - `nova boot --flavor 1 --image cirros-0.3.4-x86_64-uec --nic net-id=$PRIVATE_NET --security-groups default, ssh,icmp --poll vm-0`
- Shutdown a VM
  - `nova stop $VM_ID`
- Delete a VM
  - `nova delete $VM_ID`

Virtual Machines

- Show VM details
  - **`nova show $VM_ID`**
- Show the VM log
  - **`nova console-log $VM_ID`** (or VM name)
- Get the VNC console URL
  - **`nova get-vnc-console $VM_ID novnc`**
  - Paste the console URL to a browser to get the VM terminal.

Assign External IP to VM

- Allocate floating IP addresses from the floating range
  - `neutron floatingip-create $PUBLIC_NETWORK_ID`
- List the network port of a VM
  - `neutron port-list --device-id $VM_ID`
- Associate a floating IP with a VM nic
  - `neutron floatingip-associate $FLOATING_IP_ID $VM_PORT_ID`

Volume Management

- LVM concepts
  - https://www.howtoforge.com/linux_lvm
  - http://www.routemybrain.com/understanding-the-concept-of-logical-volume-manager-%E2%80%93-lvm/
  - http://tldp.org/HOWTO/LVM-HOWTO/anatomy.html

Volume Management

- Create a new disk volume of size 1GB
  - `cinder create 1 --display-name portable-disk`
- Create a virtual machine with this disk volume attached
  - ```
    nova boot --flavor 1 --image cirros-0.3.4-x86_64-uec
    --nic net-id=$PRIVATE_NET --block-device
    source=volume,id=$VOLUME_ID,dest=volume,
    shutdown=preserve --poll vm-1
    ```
    ```

Volume Management

- Open the vnc console of **`vm-1`** and initialize the volume:
  - **`# partition the disk`**
    `sudo fdisk /dev/vdb`
  - **`# create a file system`**
    `sudo mkfs -t ext3 /dev/vdb`
  - **`# create mount point`**
    `sudo mkdir /mnt/vdb`
  - **`# mount the disk`**
    `sudo mount /dev/vdb /mnt/vdb`

Volume Management

- Detach volume from a VM
  - `nova volume-detach vm-1 $VOLUME_ID`
- Attach volume to a running VM
  - `nova volume-attach vm-0 $VOLUME_ID`

Load Balancing with Neutron

- Create 2 virtual machines with nova
  - `nova boot --flavor 1 --image cirros-0.3.4-x86_64-uec --nic net-id=$PRIVATE_NET --poll vm-00`
  - `nova boot --flavor 1 --image cirros-0.3.4-x86_64-uec --nic net-id=$PRIVATE_NET --poll vm-01`
- Create a load balancer pool
  - `neutron lb-pool-create --lb-method ROUND_ROBIN --name balancer-pool --protocol TCP --subnet-id $PRIVATE_SUBNET`

Load Balancing with Neutron

- Add the two VMs to the load balancer pool
  - **neutron lb-member-create --address $SERVER1_IP --protocol 22 balancer-pool**
  - **neutron lb-member-create --address $SERVER2_IP --protocol 22 balancer-pool**
- Create a virtual IP (VIP)
  - **neutron lb-vip-create --name lb-vip --protocol-port 22 --protocol TCP --subnet-id $PRIVATE_SUBNET balancer-pool**

Load Balancing with Neutron

- Associate a floating IP with the VIP
  - `neutron floatingip-associate $FLOATING_IP_ID $VIP_PORT_ID`
- Port ID of a VIP can be obtained by
  - `neutron lb-vip-show`

OpenStack Python API

- OpenStack has a Python binding for it’s RESTful API
- Each component of OpenStack exposes it’s own API
- The first step is to create a Python object that acts as a client to a particular OpenStack component

OpenStack Python API - Nova

- To use nova api import the novaclient first
  - `from novaclient import client as nova_client`
- Create a nova client by providing it with proper credentials
  - `nova = nova_client.Client(<api-version>, <username>, <password>, <tenant-name>, <auth_url>)`
- Once authorized, the nova object will be used to make all API calls

OpenStack Python API - Nova

- List all flavors
  - `nova.flavors.list()`
- List all servers
  - `nova.servers.list()`
- Find a specific server
  - `nova.servers.find(name="vm-0")`
- Show the supported operations on a server
  - `dir(nova.servers.find(name="vm-0"))`

OpenStack Python API - Nova

- Show a server's console log
  - `nova.servers.find(name="vm-0").get_console_output()`
- Show status of a server
  - `nova.servers.find(name="vm-0").status`
- Show the tenant who owns this server
  - `nova.servers.find(name="vm-0").tenant_id`

OpenStack Python API - Nova

- Find a server’s Id using the API
- List the security groups a server belongs to
- Reboot a server
- Pause a server, print its status and unpause the server

OpenStack Python API - Nova

- Create a new server
  - `nova.servers.create( name='vm-2', flavor=nova.flavors.find(name='m1.very-tiny'), image=nova.images.find(name='cirros-0.3.4-x86_64-uec), nics=[{'net-id' : <NET_ID>}])`

OpenStack Python API - Neutron

- Similar for neutron. Create a client first
  - ```
    from neutronclient.v2_0 import client

    neutron = client.Client(username=<username>,
    password=<password>, tenant_name=<tenant_name>,
    auth_url=<auth_url>)
    ```
    ```

OpenStack Python API - Neutron

- List the networks
  - `neutron.list_networks()`
- List the subnets
  - `neutron list_subnets()`
- List the routers
  - `neutron list_routers()`