0.4 Create a first VM

Now you can create a first VM.

# Create the delta disk file

The first thing we need to do is create a virtual hard drive that points to the template we just created as its "backing file." The backing file has ALL the data needed. Only as changes are made (new or modified files) will they be stored in the delta file -- so the delta file can be very small, while the backing file contains OS files that don't change.

Change directory to /var/lib/libvirt/images. This is the directory we will keep all delta files in.

Issue the command:

qemu-img create -f qcow2 -F qcow2 -b /var/lib/libvirt/boot/20centos7.qcow2 test1.qcow2

This will create a disk file called test1 in qcow2 format with a backing file (-b) of centos7.qcow2.

Note how FAST! this operation is.

Also do an ls -l to see the size of the test1.qcow2 and ../boot/centos7.qcow2.

# Create the test1 VM

Now we can create a VM that will boot that disk.

Create a bash script in the root user's home directory called mktest1 with the following content:

#! /bin/bash

virt-install \

--virt-type=kvm \

--name test1 \

--ram 2048 \

--vcpus=1 \

--cpu host \

--os-variant=centos7.0 \

--hvm \

--network network=nat1 \

--graphics vnc \

--disk /var/lib/libvirt/images/test1.qcow2 \

--boot=hd \

--noautoconsole

This will do the command virt-install. All the remaining lines are input to that command.

We provide the name of the VM and its cpus and memory. The --cpu host option tells it to pass through all characteristics of the hosting system -- the key cpu flag we are interested in is "vmx" which enables nested virtualization.

The nat1 network you created is specified and the disk you just created.

Make your script file executable and then run it. You should get back:

Starting install...

Domain creation completed.

# Accessing test1 VM

To see that you have a running VM now you can do

virsh list --all

This will show you the state (running), the name (test1) and the ID. In my case the ID is 3. The ID will change each time you start the system.

To see the system, you can simply do:

virsh console <ID# or Name>

If you press enter a couple of times, the login prompt should show up. Please note that to get OUT of this console session you will need to press ctrl and the "]" key.

Logging in. We set a password for the root user (samwiseGamgee), try to login as root user now. It should work.

Do an "ip a" and you will see that you have an interface (eth0) but it is not connected to any network (no IPv4 address).

Go to /etc/sysconfig/network-scripts and edit the ifcfg-eth0 file with this information:

change BOOTPROTO= to "static"

add the following to the bottom of the file

IPADDR=172.16.10.10

NETMASK=255.255.255.0

GATEWAY=172.16.10.1

DNS1=8.8.8.8

Save your edits and quit the editor.

From the command line issue:

ifdown eth0

ifup eth0

Then do another "ip a". You should now see the IP address you specified in the ifcfg-eth0 file on the 'inet' line of the eth0 interface.

Try a ping of the gateway:

ping 172.16.10.1

This should work. And then try a ping of www.google.com.

ping of www.google.com

this should also work.

# Accessing test1 from the internet

One more bit of configuration. We need to open a hole in the firewall Google has to your network.

In Google Cloud Platform use "the hamburger" to find VPC Network and then find Firewall.

Select Create Firewall Rule.

Name: cloudstack-allow

Targets: All instances in the network

Source IPv4 range: 0.0.0.0/0

Protocol / port: tcp / 8080

You already created a rule in the OS (using iptables) to forward packets addressed to port 8080 to a system at 172.16.10.2. We are going to "change" that rule.

**To do this you need to be logged in as root user on outernetwork1.**

Issue the command:

iptables -t nat -S

this will list out ALL the rules applied to the "NAT" table on outernetwork1.

You will see there is ONE prerouting time command using -A (append).

Grab that command minus the -A.

On the command line issue:

iptables -t nat -D <paste the command starting with PREROUTING here>

The -D (instead of -A) will delete the rule.

Now do:

iptables -t nat -A PREROUTING -i ens4 -p tcp -m tcp --dport 8080 -j DNAT --to-destination 172.16.10.10:8080

To verify it is in effect you can do iptables -t nat -S again and you should see your new rule.

And you will need to get rid of some extraneous FORWARD routing libvirt sets up by issuing the command:

iptables -F

This will Flush (delete) the forwarding rules until libvirt sets them up again.

**Return to your test1 system logged in as root user.**

On test1 issue the command:

python -m SimpleHTTPServer 8080

This starts an http web listener on port 8080 using a python module.

From the VM instances table in Google Cloud Platform get the External IP address of your outernetwork1 VM.

Start a new tab in your personal browser and put the following on the address line:

http://<external IP address>:8080

You should see an indication in test1 that an HTTP GET request was acted on.

In your browser you will see a directory listing for whatever directory root user was in when the python command was executed.

With all this working, you are now set to create VMs on which you will install cloudstack.

Move on to the deliverables for this section.