1.3 Create a first VM

Before you jump to doing this, please be sure you have an operational Image/Template to work with (see the end of section1.2).

The problem is that the centos5.5 template Cloudstack provides boots VEEERRRRRYYYY slowly in this environment. It may seem like it isn't booting at all. But, let's give it a whirl anyway.

## Just in Case

If you want to know what the management server is doing, you can go to

/var/log/cloudstack/management

and do

tail -f management-server.log

## Instances

Lots of debate goes into what to call a VM in a cloud environment.

KVM/Libvirt call them domains. Some people just call them systems or VMs.

Cloudstack uses the term Compute Instance.

So in the left Nav access "Compute" and then "Instances".

Find the "Add instance +" button and press it.

1 -- we can use all the defaults in this section, so you don't have to fill anything out.

2 -- there is only one template and it is already selected

3 -- change the compute offering to "Medium instance" in the hope we can get the system to boot if we just give it some more CPU and Memory

4 -- no additional data disk, just the boot drive -- so leave this as is

5 -- select the default security group

6 -- we don't have an ssh key pair, so leave this blank

7 -- leave advanced settings alone

8 -- provide a name and select the US keyboard

Press "Launch instance"

## What happens

After you press "Launch instance" the VM will stay in a "starting" state for a very long time.

It is waiting for a virtual network to be set up. The first system on a network causes a virtual router to start. You can look under Infrastructure and will now find that you have one virtual router.

When the router is operational, the VM you requested is given an IP address and it progresses into a "Running" state.

## Accessing your VM

The NORMAL way they give you to access your VM through the Cloudstack UI is a via a VNC utility. If you click on the name of your VM you can find an icon that represents a terminal shell environment (square box with a >\_ in it).

However, this won't work in our environment. The communication appears to go out over the external IP address and needs to return to the console proxy (system VM). I haven't been able to figure this networking out (anyone want to try it for a final project?).

So… we will have to use other means.

You know that this install uses Libvirt and KVM, just like outernetwork1. And on outernetwork1 we used the

virsh console

command to access the VM. Yes, we can do that here too.

Cloudstack doesn't refer to your VM by the name you gave it. That is just for your convenience. If you look at the Instances web page, you will see "Internal name" and it looks something like

i-2-4-VM

Look for the internal name in the output from:

virsh list --all

done from the cloudstackmgr1 command line. If it is in a running state you can connect to it with

virsh console <id or name value>

Press enter a couple of times. It may appear to be doing nothing, but -- it is, in fact, booting. Given enough time, you should see it start to go through startup messages.

But this is really unsatisfactory…. so, let's add a different template.

## Deleting a VM

Use ctrl-] to leave your attempt at virsh console.

Go back to the Cloudstack web UI -- Console > Instances.

Select the instance.

Select "Stop instance" and slide the Force option to the on position.

Select "Destroy instance" and slide the Expunge option to the on position.