0.3 Install an OS to use with Guests

The outernetwork1 system is just there to provide a libvirt/qemu/kvm network and environment in which we can create our cloudstack systems.

The cloudstack systems themselves will run the CentOS 7 operating system.

We are going to use an image that has already been made cloud ready. This way, we don't have to go through a full centos7 install.

We can also use this one OS image as a template for the VMs we create. Instead of copying the OS image for every VM, we will use the same template for every VM and store just the changes to the VM in a "delta" (or changes) file.

# Getting the CentOS 7 template

Change your directory to /var/lib/libvirt/boot.

You can search for this yourself with the search term "centos 7 qcow2 image download".

But I will just give you the URL you will find.

Issue the command:

curl https://cloud.centos.org/centos/7/images/CentOS-7-x86\_64-GenericCloud.qcow2 -o centos7.qcow2

This will create the file centos7.qcow2 in the "boot" directory (-o for output in the command above).

# Give the template a password

This template runs a tool called cloud-init which is used to perform system "personalization" when the system boots. We will cover this in a later class.

One thing the template doesn't have is a password for any user. So, we will set one we can use with the root user in this learning environment. In the real world, you wouldn't set a password for root.

Install a tool you will need:

apt install libguestfs-tools

If you are still in the "boot" directory you can now issue the command:

virt-customize -a centos7.qcow2 --root-password password:samwiseGamgee --uninstall cloud-init

This will set root's password to samwiseGamgee and remove the cloud-init toolset we don't need.

Yes, with the libguestfs tools you can EDIT a VM virtual disk file -- pretty cool stuff.

# Enable nested virtualization and ip forwarding

You will recall that we want every VM we create to be able to do nested virtualization. We will make sure those features are turned on in the OS by making changes to the template.

Still in the "boot" directory issue the following command (the tool used is from libguestfs):

guestfish -i -a centos7.qcow2 write /etc/modprobe.d/kvm\_intel.conf "options kvm\_intel nested=1"

This will create (write) the string starting with "options…" to the kvm\_intel.conf file.

If you want to check that it has done it you can:

guestfish -a centos7.qcow2 -i

and at the ><fs> prompt issue

><fs> cat /etc/modprobe.d/kvm\_intel.conf

You can stay in guestfish for the next command.

To modify the /etc/sysctl.conf file to enable IP forwarding. Be in guestfish as above and then:

><fs> vi /etc/sysctl.conf

add a line that says

net.ipv4.ip\_forward=1

After you have saved your edits, issue the command "exit" from the ><fs> guestfish environment.

Your changes have been written to the template.

# Resizing the template partition

This template disk is "only" 8GB large. We can make do with that, but it might prove kind of tight.

If we run out of disk space we could add another virtual partition -- but that's a hassle we can save for the Linux class! Let's just make the template partition larger.

First take a look at the existing filesystem layout:

virt-filesystems --long -h --all -a centos7.qcow2

You will see that there is only one partition /dev/sda1. That is the one we will expand.

Create a new virtual disk that is 20GB, but has no data on it.

qemu-img create -f qcow2 -o preallocation=metadata 20centos7.qcow2 20G

Now resize from centos7.qcow2 to this new disk 20centos7.qcow2:

virt-resize --expand /dev/sda1 centos7.qcow2 20centos7.qcow2

You can now choose a backing disk that is either 8GB (centos7.qcow2) or 20GB (20centos7.qcow2).