Compute Node Setup

The compute nodes will be entirely set up utilizing the Puppet configuration management tool. The only thing that should be done by hand (preferably utilizing a parallel shell if you have many nodes or other basic tool during provisioning) is to install the Puppet package from the appropriate repository. I'd suggest something agentless like Ansible instead here.

Pull in the latest changes to the Git repository:

```
$ cd ~/build-a-cluster && git pull
```

If you had a ***freshly generated image*** from an LCI instructor, make sure to set the hostnames again:

```
$ bash ~/build-a-cluster/scripts/hostname_setup.sh

$ exec bash

$ bash ~/build-a-cluster/scripts/fix_puppet.sh

$ sudo clush -w @compute systemctl is-active puppet
```

1. Verify all compute nodes are connected to the Puppet master.
    a. `$ sudo clush -g compute puppet agent -t`
2. Configure the compute nodes to mount the NFS applications and home directories. Example Puppet manifest file is below.
    a. `$ sudo cp -R ~/build-a-cluster/etc/puppet/modules/nfs_filesystems /etc/puppet/modules/`
    b. `$ sudo vim /etc/puppet/manifests/site.pp`
        i. Add "include nfs_filesystems"
    c. `$ sudo clush -w @compute puppet agent -t`
    d. `$ sudo clush -bw @compute df -h`
3. Make sure that the cluster users are available by synchronizing the following files appropriately. This can often be carried out by a cluster manager, but we'll utilize Puppet here, see `~/build-a-cluster/etc/puppet/modules/users/manifests/init.pp`:

a. /etc/passwd

   i. `$ sudo mkdir -p /etc/puppet/modules/users/{manifests,files}`

   ii. `$ sudo ln -s /etc/passwd /etc/puppet/modules/users/files`

b. /etc/group

   i. `$ sudo ln -s /etc/group /etc/puppet/modules/users/files`

c. /etc/shadow (We also need to set a POSIX ACL so Puppet can read /etc/shadow file.)

   i. `$ sudo ln -s /etc/shadow /etc/puppet/modules/users/files`

   ii. `$ sudo setfacl -m u:puppet:r /etc/shadow`

d. Copy the necessary Puppet manifests from the Git repo

   i. `$ sudo cp ~/build-a-cluster/etc/puppet/modules/users/manifests/init.pp /etc/puppet/modules/users/manifests/`

   ii. `$ sudo vim /etc/puppet/manifests/site.pp`

      1. Add "include users"

   iii. `$ sudo clush -w @compute puppet agent -t`

4. WARNING: While this may work for a few users, it is significantly better to utilize a centralized authentication system like FreeIPA, 389 Directory Server, OpenLDAP, or other. Centralized user management is **very** important.

5. We can now see if our user created in Chapter 05 (lcistudent) can ssh to a compute node.

   a. `$ sudo su - lcistudent`

   b. `[lcistudent@master ~]$ ssh node02`

   c. `[lcistudent@node02 ~]$ exit`

   d. `$ exit`

6. We should also set up Puppet to make sure we have all our packages installed on our compute nodes. Therefore, we can create a Puppet module to install packages as necessary. For the next section on the job scheduler, we'll use a more complex Puppet

configuration to show the true power of configuration management. We'll architect The Puppet modules as follows:

a. Create a module for making sure repositories are configured for each compute node

b. Create a module to make sure all appropriate packages are installed

c. Segregate the installation, configuration, and service portions of the scheduler.

d. Create a Puppet module defining a compute node which will encompass all necessary modules.

e. There a collection of Puppet files within the git repo (`~/build-a-cluster/etc/puppet/modules/scheduler`). We've made a module to manage PBS as client software and make sure all dependency packages are in contained within the Puppet configuration.

f. `$ sudo cp -R ~/build-a-cluster/etc/puppet/modules/scheduler /etc/puppet/modules/`

g. `$ sudo cp -R ~/build-a-cluster/etc/puppet/modules/repos /etc/puppet/modules/`

h. `$ sudo vim /etc/puppet/manifests/site.pp`

    i. Add "include scheduler"

i. `$ sudo clush -w @compute puppet agent -t`

/etc/puppet/modules/nfs_filesystems/manifests/init.pp

```
class nfs_filesystems {

        file { "/apps":
                ensure => "directory",
                owner => "root",
                group => "root",
                mode => "755",
        }

        mount { "/apps":
                device => "master:/exports/apps",
                fstype => "nfs",
                ensure => "mounted",
                options => "defaults,noauto",
                atboot => true,
        }

        file { "/dhome":
                ensure => "directory",
                owner => "root",
                group => "root",
                mode => "755",
        }

        mount { "/dhome":
                device => "master:/exports/dhome",
                fstype => "nfs",
                ensure => "mounted",
                options => "defaults,noauto",
                atboot => true,
        }

}
```

/etc/puppet/modules/users/manifests/init.pp

```
class users {
      file {      "/etc/passwd":
                  ensure => "file",
                  owner => "root",
                  group => "root",
                  mode => "644",
                  links => "follow",
                  source => "puppet:///modules/users/passwd",
      }

      file {      "/etc/shadow":
                  ensure => "file",
                  owner => "root",
                  group => "root",
                  mode => "644",
                  links => "follow",
                  source => "puppet:///modules/users/shadow",
      }

      file {      "/etc/group":
                  ensure => "file",
                  owner => "root",
                  group => "root",
                  mode => "644",
                  links => "follow",
                  source => "puppet:///modules/users/group",
      }
}
```