Configuring your environment for OpenStack

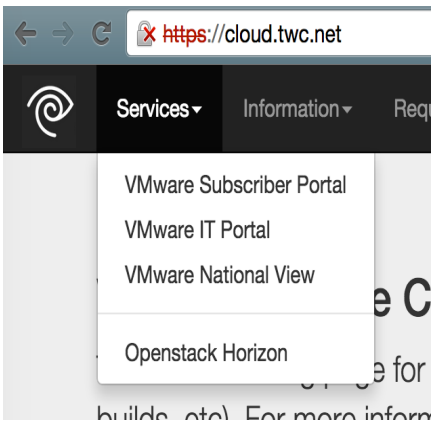
Section 1 Access via Horizon

Horizon is the graphical user interface for OpenStack that is also available for the TWC cloud. There are a number of ways to access horizon, but the easiest is to point a web browser at the web url:

https://horizon.os.cloud.twc.net/horizon

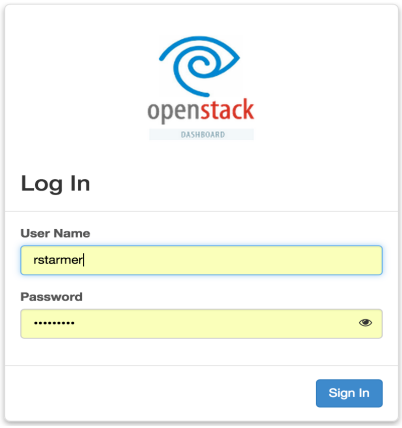
It may however be easier to remember just:

https://cloud.twc.net



And from there select the Services menu item, and then the OpenStack Horizon tab.

Once the Horizon portal loads, your first task is to log in via your EID and corporate password, unless you have been given a service ID, and associated password.



Once you have successfully logged in to Horizon, it will be possible to manipulate a number of the capabilities of OpenStack.

Section 2 Installing OpenStack CLI tools

To use the cli you will first need to install the appropriate OpenStack command line clients, and you will normally do this on your development machine (e.g. your laptop, or a development VM depending on your needs). In general, the installation is as simple as running the “pip” python installer.

For details on specific OS installations visit:
http://docs.openstack.org/user-guide/content/install_clients.html

There is also a document in TWC that lists the latest tested versions of the CLI tools, so if your version is >= to the versions on the list, you can likely skip the rest of this document. The guide is avialable here:

https://cloud.twc.net/conf/pages/viewpage.action?pageId=14222062

For Unix(alike) Clients such as Linux or OS-X

Open a Terminal window and run:

```
sudo easy_install pip
```

Note that you may have to enter your local system password in order to elevate your privileges to root.

For OS-X Specifically, you need to have Xcode's command line tools at a minimum (if you have the full Xcode installed you may still need to install the command line tools if you haven't already). The quickest way to do this is to run the following at a command prompt, which will pop up a notice asking you if you want to install.

```
xcode-select --install
```

For Linux, you will likely need to ensure you have the pip development packages as well, called something like python-dev or python-devel depending on your specific flavor of Linux.

If you have or had an older set of clients, or if you had installed clients from another service, you should likely first uninstall any/all that were installed, and then install them from scratch. To uninstall (assuming you have a pip environment already):

```
$ sudo pip install pip --upgrade
$ for n in nova cinder keystone neutron swift heat glance ceilometer \
trove sahara openstack; do sudo pip uninstall -y python-${n}client; \
done
```

Then run the install:

```
$ for n in nova cinder keystone neutron swift heat glance ceilometer \
trove sahara openstack; do sudo pip install -y python-${n}client --upgrade; \
done
```

For Windows Clients such as Windows 7

Python is available for windows, and you'll want Python 2.7, not Python 3.X at this point, which is available here:

https://www.python.org/downloads/windows/

Once you have python installed, ensure that your PATH environment has `C:\Python27\Scripts` included in it:

```
set PATH=%PATH%:C:\Python27\Scripts
```

Then install pip:

```
C:\> easy_install pip
```

And then as with the linux environments, ensure you have the latest pip, uninstall any previous installations and install the current set:

```
pip install pip --upgrade
pip uninstall -y python-novaclient
pip uninstall -y python-cinderclient
pip uninstall -y python-keystoneclient
pip uninstall -y python-neutronclient
pip uninstall -y python-swiftclient
pip uninstall -y python-heatclient
pip uninstall -y python-glanceclient
pip uninstall -y python-ceilometerclient
pip uninstall -y python-troveclient
pip uninstall -y python-saharaclient
pip uninstall -y python-openstackclient
```

And then install the clients:

```
pip install -y python-novaclient
pip install -y python-cinderclient
pip install -y python-keystoneclient
pip install -y python-neutronclient
pip install -y python-swiftclient
pip install -y python-heatclient
pip install -y python-glanceclient
pip install -y python-ceilometerclient
pip install -y python-troveclient
pip install -y python-saharaclient
pip install -y python-openstackclient
```

Verify that the clients are availalbe

Now try to run one of the commands, such as:

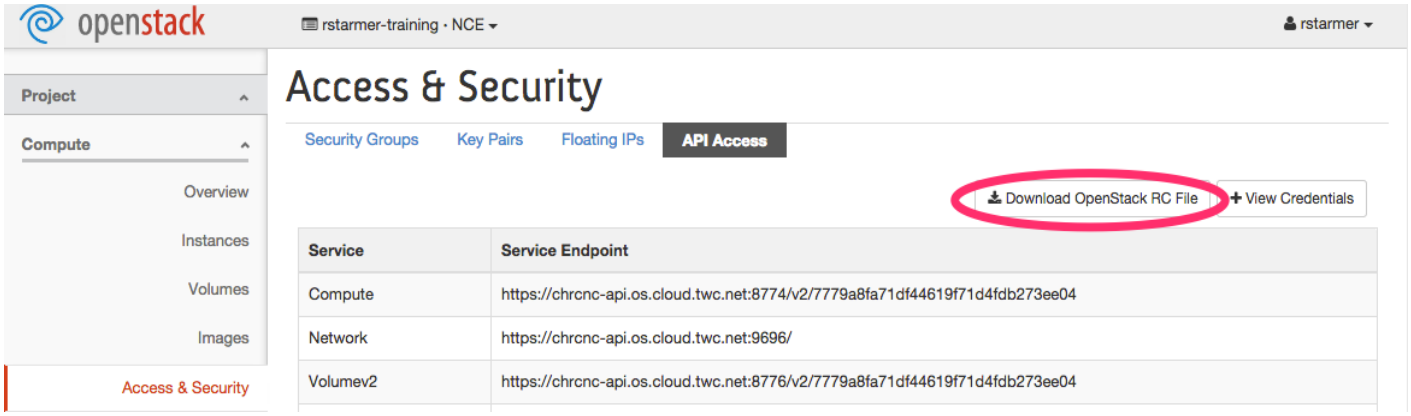
```
nova list
```

This should fail with an error like:

```
ERROR (CommandError): You must provide a username or user id via --os-username,
--os-user-id, env[OS_USERNAME] or env[OS_USER_ID]
```

Section 3 OpenStack login credentials “RC” file

In order to be able to use OpenStack cli clients you have just installed, you need to set up the necessary environment variables. While you could manually create a script named openrc.sh from scratch, we can get this file from the our Horizon Portal session. If you have logged out or not yet logged in, log into Horizon and then navigate to Project > Compute> Access and Security> API access. Select the ‘Download OpenStack rc file’ button and copy this file into your local dev directory. (You can create one now if needed.)



The file you’ve just downloaded contains the following local environmental information:

```
export OS_USERNAME=”JoeUser”
export OS_PASSWORD=”joes_password”
export OS_TENANT_NAME=”demo”
export OS_AUTH_URL=”https://chrcnc-api.os.cloud.twc.net:5000/v2.0”
export OS_REGION_NAME=”NCE”
```

In the above, OS_USERNAME/OS_PASSWORD are OpenStack user name and its password. OS_TENANT_NAME is the name of the project created. OS_AUTH_URL is the URL of the Keystone endpoint. in your deployment, and OS_REGION_NAME is the region (NCE or NCW) for your application deployment.

Now, set the environment variables by running openrc.sh as follows (note the download name will be {project}-openrc.sh).

```
$ source {project}-openrc.sh
```

If you are on a windows laptop, you will need to generate this file yourself, or at least change out the “export” to “set” and change from a .sh to a .bat file. If you edit the file to look like the following, you should be able to get the same “setup” as a Unix based environment:

```
# OpenRC.sh equivalent for Windows
```

```
# Save this content or equivalent as OpenRC.bat
ECHO OFF
SET OS_USERNAME=”username”
SET OS_TENANT_NAME=”tenant”
SET OS_TENANT_ID=”tenant_uuid”
SET OS_AUTH_URL=”auth_url”
SET OS_REGION_NAME=”NCE”
SET /P OS_PASSWORD=Enter your OpenStack Project Password (or EID password):
```

Then, you can “source the file with:

```
C:\> CALL OpenRC.bat
```

In either the Windows or Unix style model, you will be prompted to provide your TWC Portal password. Once that is accepted, you are ready to interact with OpenStack through OpenStack command line clients.

Just to check that all is working run the command:

```
$ nova list
```

This should show an empty result (as below), as you have not yet created any VMs, but this indicates that you now have cli access to OpenStack.

```
+----+------+--------+------------+-------------+----------+
| ID | Name | Status | Task State | Power State | Networks |
+----+------+--------+------------+-------------+----------+
+----+------+--------+------------+-------------+----------+
```

Security Keys

OpenStack attempts to maintain security for it’s cloud instances by limiting the access mechanisms available. Specifically, cloud deployed VMs do not have any passwords set or in other words, you can not log in with {or without} a password into the VMs normally deployed in OpenStack. To remedy this, the openstack community leverages a tool called ‘cloud-init’ to allow deployed VMs to request and download public ssh keys so that a users private key can be used to authenticate an SSH session into a cloud image.

We will need a pair of keys for later lab components, so you should create one if you don’t have one already.

Key create/install for OSX

You may already have an ssh keypair, and you can get that installed into OpenStack fairly quickly. The simplest method is to use the nova keypair-add command along with the _PUBLIC_ half of a keypair. Specifically if the file ~/.ssh/id_rsa.pub exists (and the private counterpart ~/.ssh/id_rsa), you can just upload that file with the nova client or via Horizon.

For the Nova client, after having sourced your {project}-openrc.sh script, you can run:

```
nova keypair-add default --pub-key ~/.ssh/id_rsa.pub
```

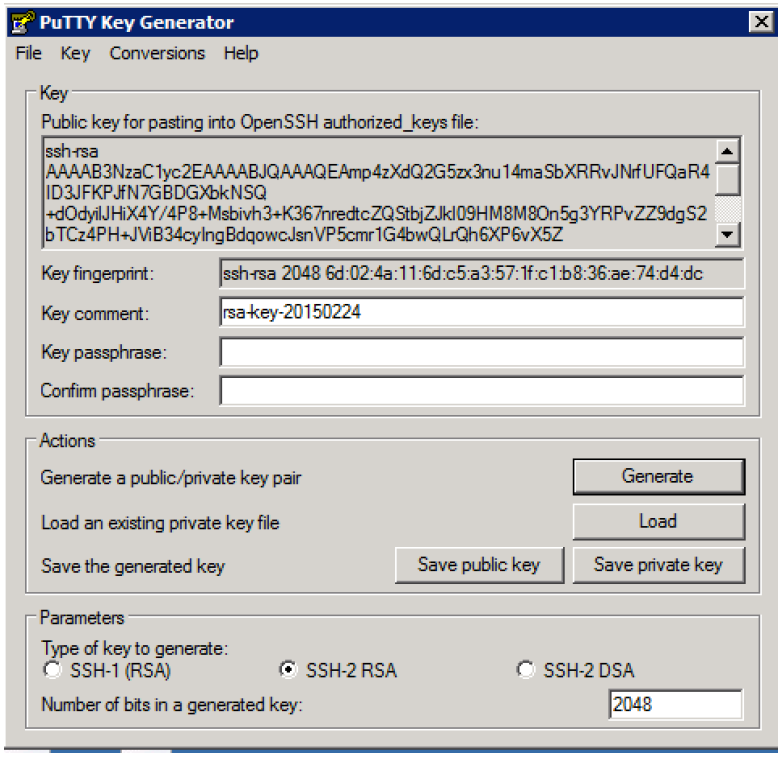
If you don't have a keypair, you can create one easily with:

```
ssh-keygen -t rsa -N '' -f ~/.ssh/id_rsa
```

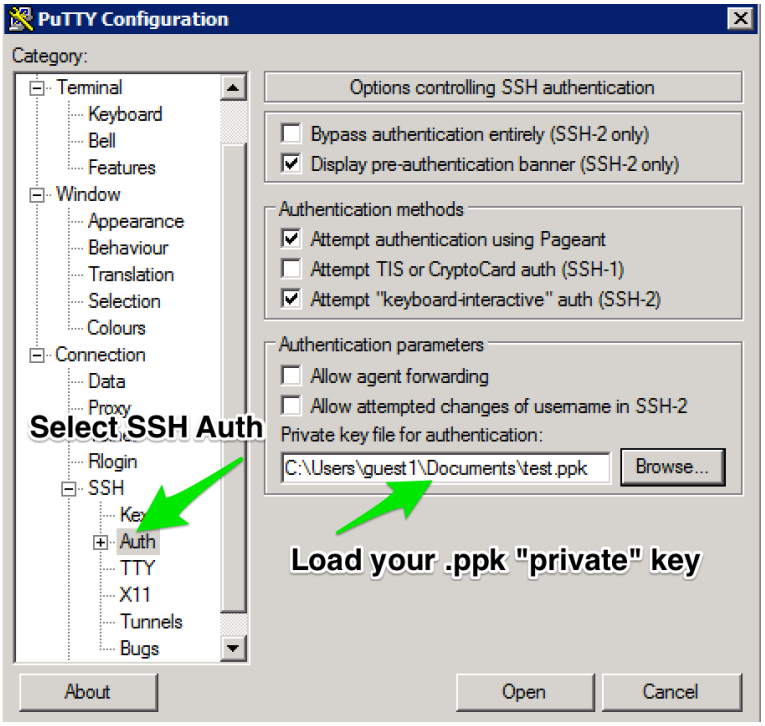
Key create/install for Windows

In Windows, you will need a utility, such as either the Cygwin tools (and the openssh package), or the putty-gen.exe application (and likely at the same time, it makes sense to get putty-gen's ssh client putty.exe)

PuTTYgen to create a new private key (if you don't already have one), and then copy the "Public Key for Pasting into OpenSSH" so that you can update the OpenStack keypair (as above in the Unix section). The ssh key (starts with ssh-rsa) can also be copied to a file and uploaded via nova as would be done via the OS-X model.

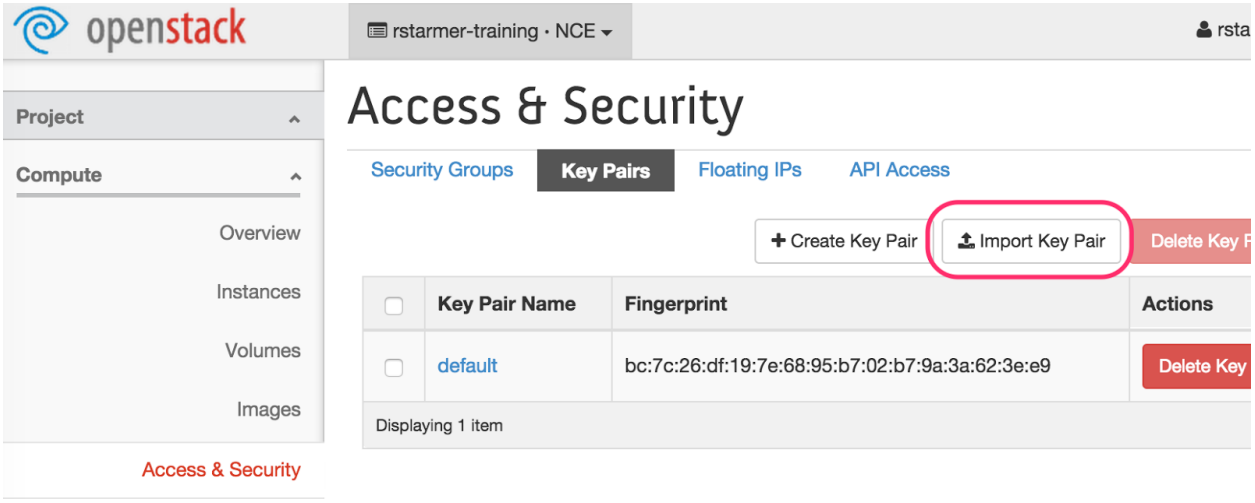


If you generate a new key, don't forget to save the private key as a .ppk file locally. You will want to install it in a tool like putty to simplify access to VMs that are created in the OpenStack environment.



For either platform

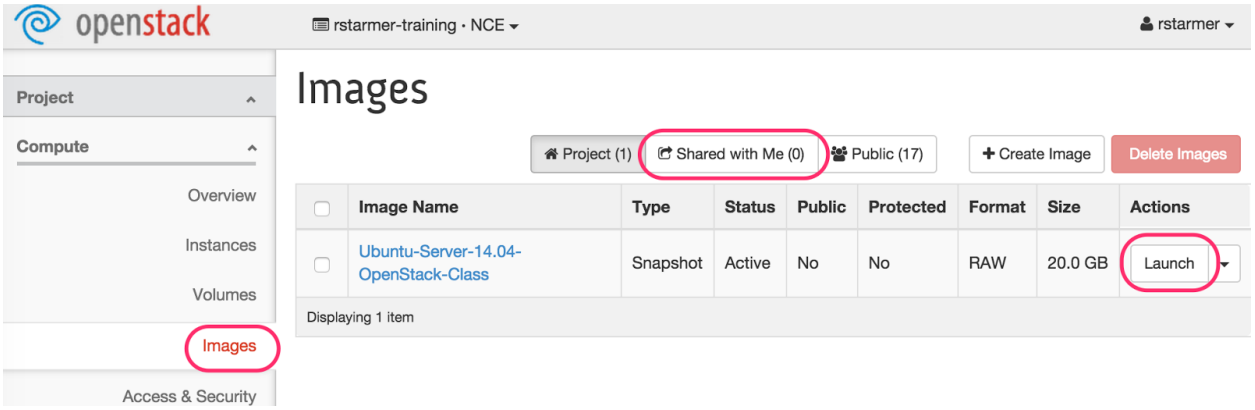
You can also upload the public key via Horizon (Project->Compute->Access & Security: Key PAirs tab, select “Import”)



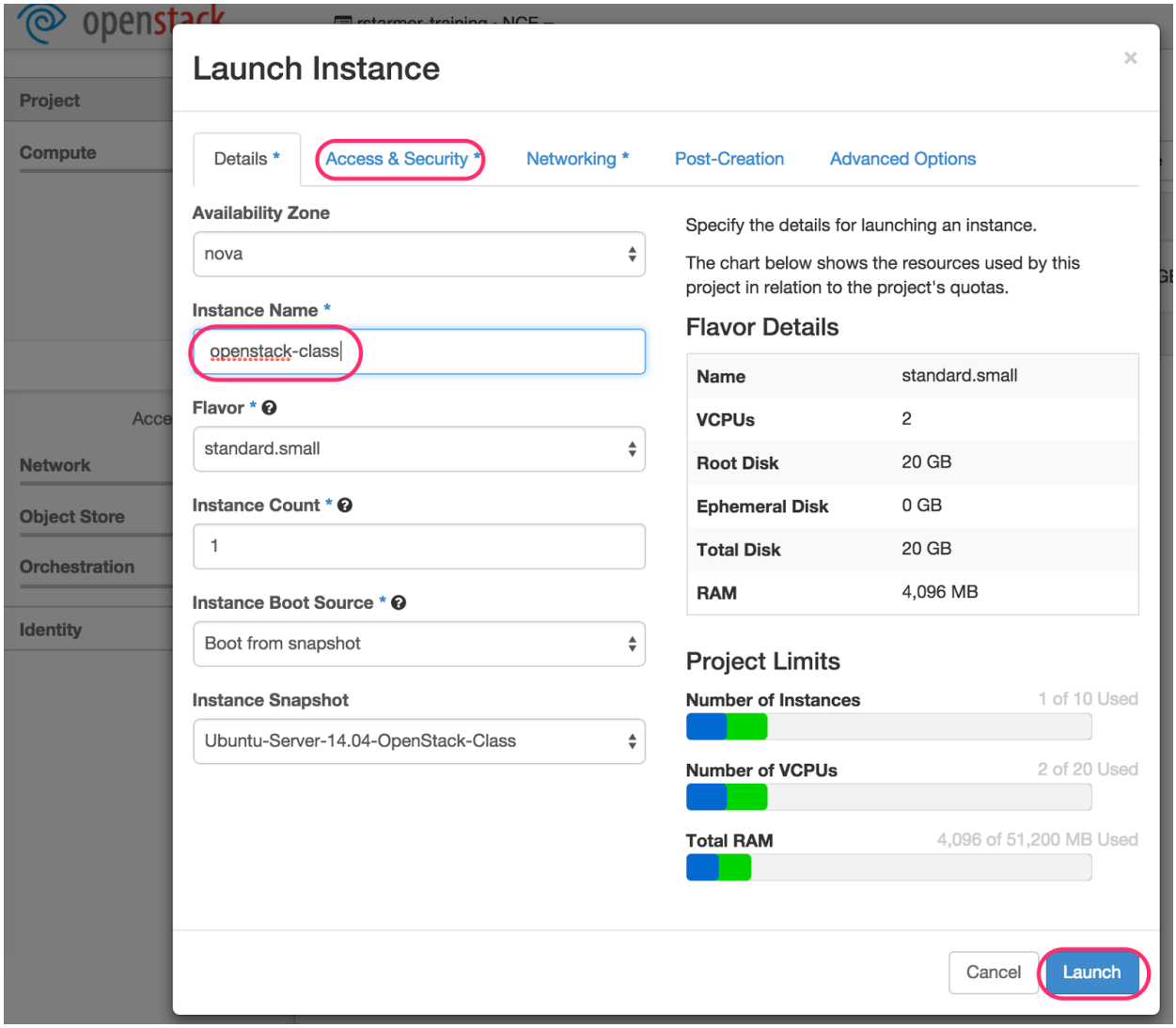
Section 4 I tried all that and I still can’t get the tools to load

If you were able to get a keypair registered with Horizon, there is one more possibility, to start a VM and use that as the source of your command line interface with OpenStack. The training projects should all have an image called Ubuntu-Server-14.04-OpenStack-Class, and

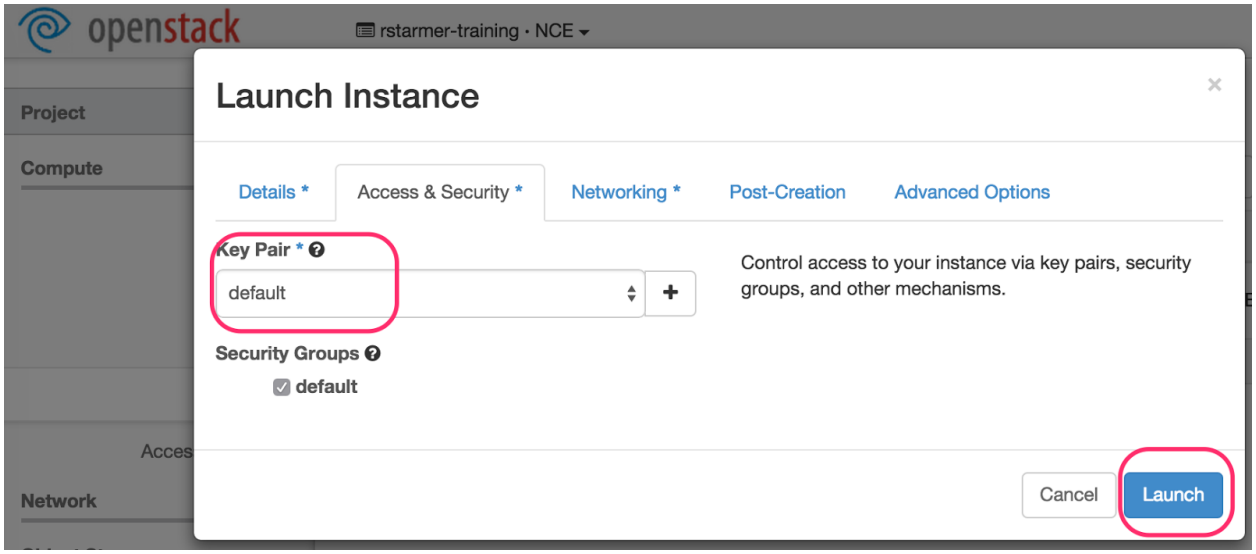
this image is pre-configured with the openstack clients. You will still have to add your {project}-openrc.sh file as in previous sections of this document, but then you will be able to use these versions of the tools to manipulate and leverage OpenStack@TWC.



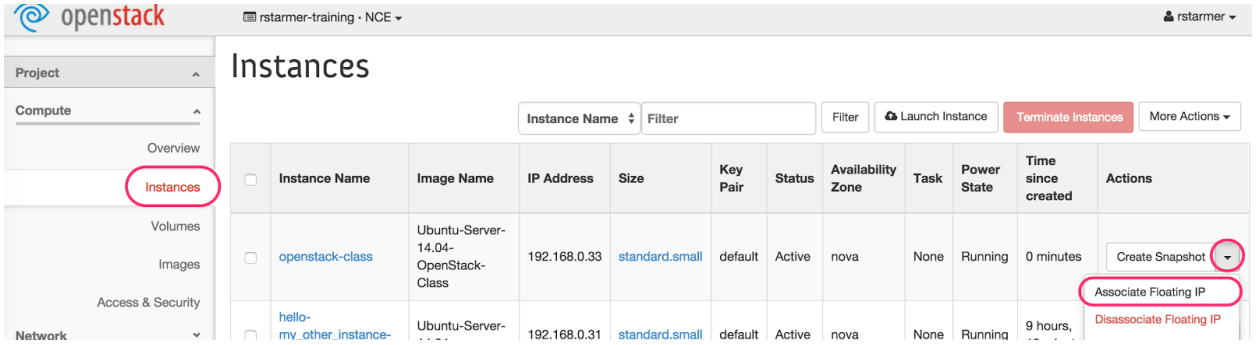
Launch the Image, and then associate a “Floating IP” with the image:

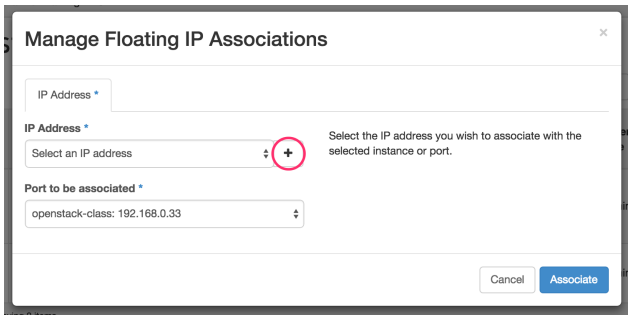


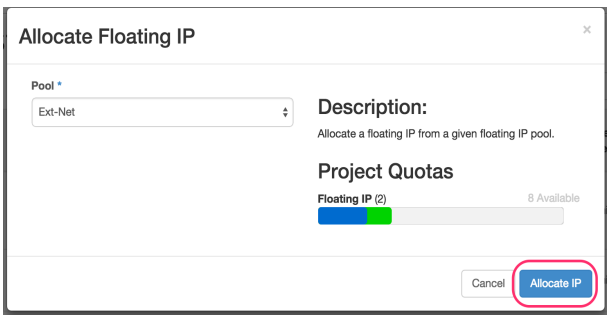
Note that if you didn't upload a security key, you will _not_ be able to get into your VM at all. This can be checked before launching the VM via the Access & Security tab on the Launch Instance Wizard.



Once my VM is in the Active state (you can check on the Instances page of Horizon), it is necessary to add a Floating IP to the system, so that we can access it via ssh remotely:

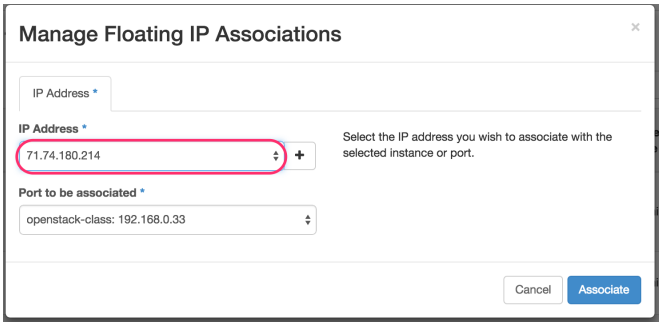






Once the response is returned, note the IP address that was assigned, as we'll use this to log in to the instance via ssh.

Finally, either open a terminal session (or Cygwin session), or launch a tool like putty (don't forget to pass putty the path



to the private key!), and log in to your new VM (note, the default user for this VM is ubuntu).

```
ssh ubuntu@71.74.180.214
```

And now incorporate your openrc.sh script as we did earlier in this document, and you should be able to get a result from running ‘nova list’!

At this point, you should be able to continue with the rest of the lab exercises.