**Eucalyptus Education Cloud Setup**

(Thanks to Ryan Baker for edits to this document.)

There are two basic constructs for each student on the Eucalyptus Education Cloud (EEC), the desktop and the pod. The desktop gives the students a public access point with a “known good” configuration for performing lab assignments. The pod is a set of two systems used for installing and configuring a Eucalyptus frontend (CLC, Walrus, CC, and SC) and node controller (NC) with a “known good” configuration.

**Desktops**

The desktops are currently Debian Squeeze instances run by a Eucalyptus cloud. The cloud uses a range of publicly routed IPs to give students easy access. Each desktop provides the students with two methods of access: SSH and VNC. The VNC session hosts a XFCE desktop environment that the student may use to access a terminal and Firefox (for HybridFox and access to the WebUI).

Currently the image size is around 8GiB to give the student sufficient space for any storage needed during the labs.

Each desktop is set up with a user-data script during boot. The user-data script does the following:

* Set the hostname for the instance based on a reverse lookup of the current public IP address
* Update the packages on the instance
* Set the password for both the instance and the VNC session
* Start the VNC server
* Disable hashing known hosts entries for the SSH client
* Disable DNS lookups on the SSH server
* Download an already made and tested boot from EBS image from Walrus.

The current version of this script can be found at: <https://raw.github.com/EucalyptusSystems/Edu-Scripts/master/setup_desktop.sh>

We have chosen to do this with a user-data script but these instances could also use a preconfigured image.

**Pods**

**Hardware**

Currently the supported configuration for Eucalyptus currently requires installation on bare metal and at least two systems. Each student receives two HP SL170z G6 sleds (Note: This setup is in no way vendor specific). Each sled currently has the following specification:

* Intel Xeon E5504 @ 2.00GHz (Quad Core)
* 8 GB RAM
* 2 x 160GB HDD @ 5400 rpm
* 2 x 1 Gbit Ethernet ports

These systems were originally selected due to their availability upon the creation of the EEC.

Any server class system should work as long as it has at least a dual-core CPU, 4GB of RAM, and at least 100 GB of disk space available. These requirements may change depending on labs that are performed by the student in the class.

**Operating Systems**

The following OS and hypervisor combinations are currently supported by the EEC:

|  |  |
| --- | --- |
| **Operating System** | **Hypervisor** |
| CentOS 5 | Xen |
| CentOS 6 | KVM |
| Ubuntu 10.04 / 12.04 | KVM |

All systems are setup and installed using specific kickstart or preseed configurations. The [Cobbler](http://cobbler.github.com/) system management service is used as it allows for easy configuration and management of a systems. With its templating capabilities for kickstart and preseed, a single kickstart or preseed file can be used for each OS configuration for a group of systems without much work by the administrator. Cobbler has an XMLRPC API which is used to automate the deployment of these systems.

With the templating capabilities, Cobbler allows the abstraction of some configuration such as network interfaces and other settings without needing to have a different kickstart for each OS variant. Currently the node and front-end configurations for each OS variant are completed using different kickstart or preseed files.

Our current kickstart and preseed files can be found here: <https://github.com/EucalyptusSystems/Edu-Scripts/tree/master/kickstarts>. The password hashes have also been removed from these files and will need to be replaced before running. These kickstart and preseed files are rendered by the Cobbler daemon when a system requests them with any custom information being rendered for the specific host.

A couple Cobbler snippets have been customized to change the behavior around the network interface setup. In Cobbler we currently are using network interfaces in the eth0, eth1, etc style when defining an interface on a system. This convention has begun to change with RHEL/CentOS 6 based systems. Therefore, as part of the template for the network configurations, if the profile is RHEL based and at version 6 then the other values will be substituted as the kickstart is requested by the system.

Another configuration change is made but it is not done by the Cobbler system itself. Upon launching a PXE boot of a system with our automation script, if the profile for the NC is known to run KVM then a bridge is created on the NC systems in Cobbler before restarting the systems.

**Disk Layout**

The current disk layout is as follows for each OS:

|  |  |  |
| --- | --- | --- |
| **Disk Partition\*** | **Mount Point\*\*** | **Size** |
| /dev/sda1 | / | At least 20GiB |
| /dev/sda2 | Swap | 2GiB |
| /dev/sda3 | /mnt/extra | Remaining storage on /dev/sda |
| /dev/sdb1 | /instances or /buckets | Entire /dev/sdb storage space |

\*Disk partitions are not mandatory for a Eucalyptus installation. What matters most are the mount points of the devices and that they are sufficiently large for the activities to be performed by the students.

\*\*Some of these locations are not ideal for a production system but they make it easier for the students to work with the systems. Mount points such as */instances* or */buckets* on a production system would be better suited at */srv/instances* and */srv/buckets* respectively.

The configuration separates the root partition from the front-end’s bucket and volume storage (volumes could be placed on /mnt/extra) and the NC separates the root partition from the NC’s cache and work storage.

A future tweak to the disk layout could involve moving towards LVM. It would also allow for easier configuration with Debian preseed configurations that are quite broken when it comes to partitioning multiple disks.

**Networking**

There are three networks setup for the pods across two layer 2 broadcast domains. We have used subnets in all 3 private network classes to prevent confusing students new to network configuration. The different network schemes can also assist the instructors when dealing with issues in a student cloud as they can quickly notice if the subnets are used correctly.

We are currently using the following network ranges:

|  |  |
| --- | --- |
| **Subnet** | **Usage** |
| 192.168.105.0/24 | Private system network |
| 172.16.128.0/17 | Public system network |
| 10.110.128.0/17 | Private instance network (VNET\_SUBNET) |

The private system network is used for communications between the frontend and the NC. This network is put on the *em1* interface and has layer 2 separation from the public system network. The private system network does not need to be routed between the desktop cloud and the pod systems.

The public system network is used for the accessing the systems from the student desktops. Each pod has a /24 range in the 172.16.128.0/17 range. The addresses 172.16.xxx.1 and 172.16.xxx.254 are used for public access of the frontend and NC respectively. A small subset of these address are also used in the Eucalyptus configuration for VNET\_PUBLICIPS setting of each pod. These addresses are in the range 172.16.xxx.20 - 172.16.xxx.40.

The public system network must be routed between the desktop cloud and the pod systems. This allows the system to keep the number of publically routed systems to a minimum.

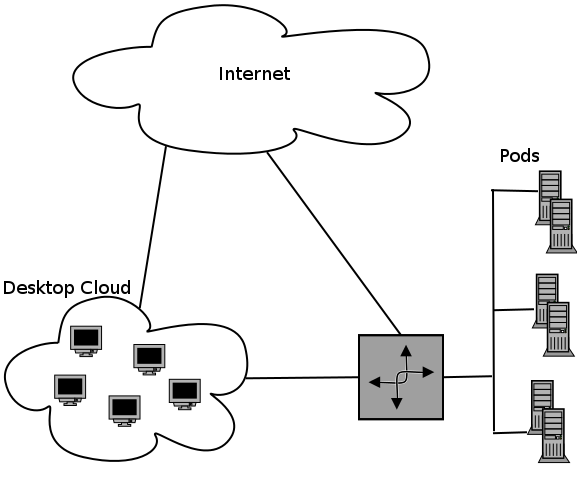
The private instance network is the private network that an instance will get an IP address from when booting. Each pod is given a range (e.g. 10.110.xxx.0/23) and this value is used for the VNET\_SUBNET value. With a VNET\_ADDRSPERNET value of 16, students can launch 5 instances in each security group. With the systems we are currently using this is close to the maximum that each NC can handle.

The last octet of the private network for the frontend system is used in the network segments of the other two networks. In the above descriptions about the public system network and the private instance network the “xxx” in each address or subnet would be replaced with the last octet of the frontend system on the 192.168.105.0/24 network. For example:

|  |  |
| --- | --- |
| Frontend Private System Network | 192.168.105.**160** |
| NC Private System Network | 192.168.105.161 |
| Frontend Public System Network | 172.16.**160**.1 |
| NC Public System Network | 172.16.**160**.254 |
| Pod VNET\_PUBLICIPS | 172.16.**160**.20 - 172.16.**160**.40 |
| Pod VNET\_SUBNET | 10.110.**160**.0 |
| Pod VNET\_SUBMASK | 255.255.254.0 |

This method for assigning IP addresses appears to allow for easier set up by the student and easier debugging by the instructors since the IP ranges can easily be determined while on the system with a simple ip addr show.

**Basic Diagram of Current Setup**



**Automation**

Currently we have a basic set of automation scripts. In the future, we intend to update these scripts to do whatever the instructor needs with minimal intervention.

Currently these scripts do the following:

* Set the profile (OS) to use on a specific set of systems.
* Restart the system and set them to netboot on the next reboot
* Change the password on the systems after they have been recently installed.

These scripts are currently a work in progress and hopefully in the future they will have many additional features.