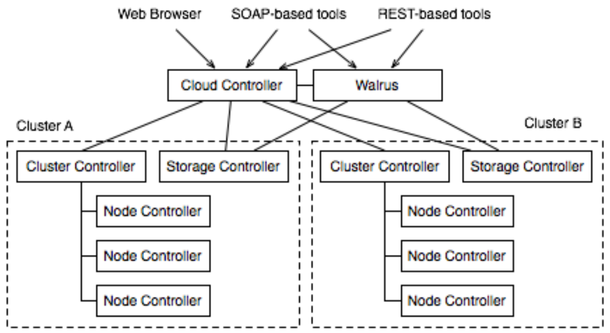
Eucalyptus Administrator's Guide (1.6)

This guide is meant for people interested in installing Eucalyptus on their resources: anything from a laptop to a set of clusters. (If you are trying to use an existing Eucalyptus installation, you may be more interested in the User's Guide.

Installing Eucalyptus (1.6)



A Eucalyptus cloud setup consists of five types of components. The *cloud controller* (CLC) and "Walrus" are top-level components, with one of each in a cloud installation. The cloud controller is a Java program that offers EC2-compatible SOAP and "Query" interfaces, as well as a Web interface to the outside world. In addition to handling incoming requests, the cloud controller performs high-level resource scheduling and system accounting. Walrus, also written in Java, implements bucket-based storage, which is available outside and inside a cloud through S3-compatible SOAP and REST interfaces.

Top-level components can aggregate resources from multiple clusters (i.e., collections of nodes sharing a LAN segment, possibly residing behind a firewall). Each cluster needs a *cluster controller* (CC) for cluster-level scheduling and network control and a "storage controller" (SC) for EBS-style block-based storage. The two cluster-level components would typically be deployed on the head-node of a cluster. Finally, every node with a hypervisor will need a *node controller* (NC) for controlling the hypervisor. CC and NC are written in C and deployed as Web services inside Apache; the SC is written in Java. Communication among these components takes place over SOAP with WS-security.

Many instructions in this guide refer to a *single-cluster installation*, in which all components except NC are co-located on one machine, which we refer to as **front-end**. All other machines, running only NCs, will be referred to as **nodes**. In more advanced configurations, such as those with multiple CCs or with Walrus deployed separately, the *front-end* will refer to just the machine running the CLC.

Eucalyptus can be installed from source or using a set of packages (RPM and DEB). The former method is more general and should work on practically any Linux system, the latter is easier but will only work on the distributions that we support. As of 1.6 they are:

- CentOS 5.4
- Debian squeeze
- OpenSUSE 11
- Ubuntu 9.04 "Jaunty" and 9.10 "Karmic"

If you are upgrading from a previous version of Eucalyptus, please follow the instructions in the Upgrade Document.

If you run into any problems, be sure to check the troubleshooting guide for solutions to commonly encountered problems.

Installing Eucalyptus from source (1.6)

1. Prerequisites

What follows is a comprehensive list of dependencies that must be satisfied before building Eucalyptus or running it. While we provide distribution-specific installation instructions that help satisfy these dependencies, this list should be useful if you are installing or building Eucalyptus on an unsupported distribution. **NOTE** - If you are upgrading from a Eucalyptus 1.6.1 or older installation, please consult the Upgrade Documentation for instructions that will explain how to preserve user account information, images, volumes and snapshots.

Prerequisites for compiling from source

- C compilers
- Java Developer Kit (SDK) version 1.6 or above

- Apache ant 1.6.5 or above
- libc development files
- pthreads development files
- libvirt development files
- Axis2C and rampart development files (included with Eucalyptus)
- Curl development files
- openssl development files
- Optional: zlib development files

Prerequisites for running Eucalyptus

There are a few different Eucalyptus components that run on either the 'front-end or 'node'. There are different run-time dependencies for 'front-end' and 'node' components. One physical machine can play the role of the front-end and the node.

Front-end run-time dependencies

- **Java 6** is needed by the Eucalyptus components running on the front end. Note that GNU Compiler for Java (gcj), included by default with some Linux distributions, is **not** sufficient. Make sure that your JAVA_HOME environment variable is set to the location of your JDK.
- **Perl** is used by helper scripts
- The head node must run a **server on port 25** that can deliver or relay email messages to cloud users' email addresses. This can be Sendmail, Exim, or postfix, or even something simpler, given that this server does not have to be able to receive incoming mail. Many Linux distributions satisfy this requirement out of the box. To test whether you have a properly functioning mail relay for localhost, try to send email to yourself from the terminal using "mail".
- Dependencies for network support differ depending on the mode used (see Eucalyptus Networking for details). For full functionality satisfy all of them:
  - For all modes:
    - `iproute` and `iptables` packages (`ip` and `iptables` commands must work)
  - For all modes except SYSTEM:
    - DHCP Server compatible with ISC DHCP Daemon version 3.0.X (dhcp3-server)
  - For MANAGED and MANAGED-NOVLAN modes:
    - `bridge-utils` package (`brctl` command must work)
    - `vtun` package, for multi-cluster configurations
  - Additionally, for MANAGED mode:
    - `vlan` package (`vconfig` command must work)
- For persistent dynamic block storage (aka EBS) to work, the front end will need to have the following software packages installed:
  - `lvm2` package (e.g., command `lvm` should work)
  - `aoetools` package. The `aoe` module needs to be loaded on the front end as well as all nodes (`modprobe aoe`). If your kernel does not have ATA-over-Ethernet support, you will have to add that.
  - `vblade` package

Node run-time dependencies

- **Perl** scripts are invoked by the Node Controller
- Two hypervisors are supported:
  1. **Xen** (version >= 3.0.x)
     - Furthermore, `xen-utils` package is needed (`xm` command must work)
  2. **KVM**
- Dependencies for network support differ depending on the mode used (see Eucalyptus Networking for details). For full functionality satisfy all of them:
  - For all modes:
    - `iproute` and `iptables` packages (`ip` and `iptables` commands must work)
  - For MANAGED and MANAGED-NOVLAN modes:
    - `bridge-utils` package (`brctl` command must work)
  - Additionally, for MANAGED mode:
    - `vlan` package (`vconfig` command must work)
- `libvirt` package (potentially with `libvirtd`, depending on hypervisor configuration)

All Eucalyptus components

- You *must* be **root** to install and start Eucalyptus components (by default they will run under a different user after start). This document assumes that all commands will be executed as root.

Attention CentOS users: The version of OpenJDK that is bundled with CentOS-5 cannot compile the version of GWT that comes with Eucalyptus as a dependency. You will need to install JDK 1.6.0 "manually". We use Sun's JDK, which can be found at http://java.sun.com/javase/downloads/index.jsp. Be sure to set your JAVA_HOME and PATH properly before running the Eucalyptus 'configure' script.

Distribution-specific examples

What follows is a superset of all packages necessary for building and running Eucalyptus on each supported distribution:

- For **Opensuse 11.1**, download and install RPMs the appropriate OpenSUSE RPM dependency package from the Eucalyptus website, then run the following command to install all required dependency packages:

```
yast2 -i bzr python-paramiko make gcc ant apache2 apache2-devel\
   java-1_6_0-openjdk java-1_6_0-openjdk-devel libvirt-devel libcurl-devel\
   vlan dhcp-server bridge-utils ant-contrib ant-nodeps curl libvirt
```

- For **Ubuntu 9.04 and 9.10**, run the following command to install all required dependency packages:

```
apt-get install bzr gcc make apache2-threaded-dev ant openjdk-6-jdk\
   libvirt-dev libcurl4-openssl-dev dhcp3-server vblade apache2 unzip curl vlan\
   bridge-utils libvirt-bin kvm vtun
```

- For **CentOS 5**, download and install RPMs the appropriate CentOS RPM dependency package from the Eucalyptus website, then run the following command to install all required dependency packages:

```
yum install -y java-1.6.0-openjdk-devel ant ant-nodeps libvirt-devel\
   curl-devel httpd httpd-devel apr-devel openssl-devel dhcp
```

- For **Debian**, run the following command to install all required dependency packages:

```
apt-get install gcc make apache2-threaded-dev ant openjdk-6-jdk\
   libvirt-dev libcurl4-dev dhcp3-server vblade apache2 unzip curl vlan\
   bridge-utils libvirt-bin kvm sudo vtun
```

Please, consult the distribution-specific pages for detailed installation instructions.

Tools for interacting with Eucalyptus

To interact with Eucalyptus, you need to install EC2-compatible command-line tools. The instructions in Eucalyptus documentation rely on the euca2ools command-line tools distributed by the Eucalyptus Team. Many other third-party tools can also be used for some of the tasks, as described on the ecosystem page.

2. Download Eucalyptus and supplied dependencies

In what follows substitute the desired version (e.g., 1.6.2) for $VERSION either manually or by setting a shell variable.

Download either

- eucalyptus-$VERSION-src.tar.gz (Eucalyptus source with included java libraries)

or

- eucalyptus-$VERSION-src-online.tar.gz (Eucalyptus source that will download java libraries at build-time)

and for both

- eucalyptus-$VERSION-src-deps.tar.gz (Eucalyptus C library dependency packages)

All packages can be found on the Eucalyptus Web site:

- http://www.eucalyptus.com/download/eucalyptus

Unpack the Eucalyptus source:

```
tar zvxf eucalyptus-$VERSION-src.tar.gz
```

Now you should have a directory eucalyptus-$VERSION. To simplify the remainder of the installation, define EUCALYPTUS_SRC environment variable to be the top of the source tree of eucalyptus and the variable EUCALYPTUS to be the directory where eucalyptus will be installed (we recommend using `/opt/eucalyptus/`):

```
cd eucalyptus-$VERSION
export EUCALYPTUS_SRC=`pwd`
export EUCALYPTUS=/opt/eucalyptus
```

3. Build Dependencies

To install Eucalyptus, you need to build packages that Eucalyptus depends on, which we provide in the above-mentioned package eucalyptus-$VERSION-src-deps.tar.gz. For the sake of this discussion, we are going to assume that all packages have been untarred inside "$EUCALYPTUS_SRC/eucalyptus-src-deps/" as above and will be installed in "$EUCALYPTUS/packages".

Unpack the dependencies and create the directory you'll use to install them:

```
cd $EUCALYPTUS_SRC
tar zvxf ../eucalyptus-$VERSION-src-deps.tar.gz
mkdir -p $EUCALYPTUS/packages/
```

Build and install the dependencies. The following instructions work on some Linux distributions, but aren't universal. *Please, consult the documentation for the specific packages for help with building them on your distribution.*

a. Axis2

```
cd $EUCALYPTUS/packages
tar zxvf $EUCALYPTUS_SRC/eucalyptus-src-deps/axis2-1.4.tgz
```

b. Axis2/C

To compile Axis2/C, you will need to locate development headers for Apache and for APR. On some distributions (e.g., Ubuntu and Debian) the following values should work:

```
export APACHE_INCLUDES=/usr/include/apache2
export APR_INCLUDES=/usr/include/apr-1.0
```

On CentOS 5, the headers should be in the following location:

```
export APACHE_INCLUDES=/usr/include/httpd/
export APR_INCLUDES=/usr/include/apr-1/
```

while on OpenSuse 11 you may find them at:

```
export APACHE_INCLUDES=/usr/include/apache2/
export APR_INCLUDES=/usr/include/apr-1/
```

With the two environment variables set, you can build and install Axis2/C as follows:

```
export AXIS2C_HOME=$EUCALYPTUS/packages/axis2c-1.6.0
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf axis2c-src-1.6.0.tar.gz
cd axis2c-src-1.6.0
CFLAGS="-w" ./configure --prefix=${AXIS2C_HOME} --with-apache2=$APACHE_INCLUDES --with-apr=$APR_INCLUDES --enable-multi-thread=no
make ; make install
```

c. Rampart/C

```
export AXIS2C_HOME=$EUCALYPTUS/packages/axis2c-1.6.0
export LD_LIBRARY_PATH=${AXIS2C_HOME}/lib:$LD_LIBRARY_PATH
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf rampartc-src-1.3.0-0euca1.tar.gz
cd rampartc-src-1.3.0
./configure --prefix=${AXIS2C_HOME} --enable-static=no --with-axis2=${AXIS2C_HOME}/include/axis2-1.6.0
make ; make install
```

Next, change the following in $AXIS2C_HOME/axis2.xml. In the 'inflow' section, change:

```
        <!--phase name="Security"/-->
```

to

```
        <phase name="Security"/>
        <phase name="Rahas"/>
```

In the 'outflow' section, change:

```
        <!--phase name="Security"/-->
```

to

```
        <phase name="Security"/>
```

4. Building Eucalyptus

Make sure JAVA_HOME is defined.

```
cd $EUCALYPTUS_SRC
./configure --with-axis2=$EUCALYPTUS/packages/axis2-1.4 --with-axis2c=$EUCALYPTUS/packages/axis2c-1.6.0 --enable-debug --prefix=$EUCALYPTUS
make ; make install
```

5. Deploying Eucalyptus

a. At this point, if you plan to use Eucalyptus on more than one node, you're ready to push the software out to the other nodes (although not all software components are required on all nodes, it is simpler to just mirror everything and selectively enable components via start-up scripts). If you installed Eucalyptus in its own directory, you can just sync the entire package to all of the hosts listed above using whatever mechanism you typically use to push changes to nodes (rsync, for instance).

```
rsync -a $EUCALYPTUS/ root@{node-host-1}:$EUCALYPTUS/
rsync -a $EUCALYPTUS/ root@{node-host-1}:$EUCALYPTUS/
...
```

On installations without a root user, such as Ubuntu, it may be easier to pull the software from each node one at a time:

```
node1# rsync -a {user}@{front-end}:$EUCALYPTUS/ $EUCALYPTUS/
node2# rsync -a {user}@{front-end}:$EUCALYPTUS/ $EUCALYPTUS/
...
```

NOTE: Installing Eucalyptus in the same directory on all nodes will make it easier to manage it, so we strongly advise you to do so.

6.) Configure hosts

a. Set up a 'eucalyptus' user on all machines

Eucalyptus will run as regular user on your systems, which you'll need to add before running Eucalyptus (we will use `eucalyptus`) on **all machines**. For most distributions, this task is accomplished by running the command:

```
useradd eucalyptus
```

For OpenSUSE, use:

```
groupadd eucalyptus
useradd eucalyptus -m -g eucalyptus
```

b. Configure your hypervisor

Ensure that this user can control your hypervisor through libvirt on **all compute nodes**. On some distributions, this can be accomplished by adding `eucalyptus` to group `libvirt` or `libvirtd` in file `/etc/group`. Please, consult the documentation for libvirt on your distribution for instructions. For more detailed information, see Hypervisor Configuration for more information.

c. Configure your networking

Eucalyptus provides several network configuration options from which to choose, depending on your local network setup, capabilities, and the networking features that you wish to take advantage of within Eucalyptus. Most networking options require that, on your node controllers, the primary interface is configured to be a bridge (this is the default configuration with some distribution's Xen hypervisor configuration). See Network Configuration for more information and set-up instructions. Once you have decided which network mode you will be using, you may be required to set up ethernet bridges on Eucalyptus component machines.

d. Configure Eucalyptus components

On your **compute nodes**, create a local directory where VM images are to be placed temporarily while VMs are running (images will be cached under the same path, too). It is important that the directory is empty as *everything in it will be removed*. Be sure to pick a partition with ample disk space as VM images can be large. We use `/usr/local/eucalyptus` in the example below.

Place the mandatory parameters (including the type of hypervisor you plan to use) into the configuration file and set up the permissions on Eucalyptus files appropriately on **all nodes**. Both tasks can be accomplished with flags to `euca_conf` tool:

- **-d** specifies the root of Eucalyptus installation ($EUCALYPTUS)
- **--hypervisor** specifies the hypervisor ('xen' or 'kvm')
- **--instances** specifies where, on compute nodes, instance files will be stored
- **--user** specifies the user that you created for running Eucalyptus
- **--setup** invokes the first-time setup tasks

```
$EUCALYPTUS/usr/sbin/euca_conf -d $EUCALYPTUS --hypervisor kvm --instances /usr/local/eucalyptus --user eucalyptus --setup
```

e. Distribution-specific post configuration steps

Some linux distributions require that the admin perform a few extra steps before bringing up Eucalyptus. This section details some of these steps:

For Ubuntu 9.04 "Jaunty" and 9.10 "Karmic", apparmor needs to be configured to allow dhcpd3 to write to the filesystem. Add the following lines to '/etc/apparmor.d/usr.sbin.dhcp3':

```
/opt/eucalyptus/var/run/eucalyptus/net/ r,
/opt/eucalyptus/var/run/eucalyptus/net/** r,
/opt/eucalyptus/var/run/eucalyptus/net/*.pid lrw,
/opt/eucalyptus/var/run/eucalyptus/net/*.leases* lrw,
/opt/eucalyptus/var/run/eucalyptus/net/*.trace lrw,
```

where you substitute '/opt/eucalyptus' with the path to where you have chosen to install Eucalyptus. Then, restart apparmor (NOTE: sometimes changes don't take effect right away - either wait or reboot the system to be sure):

```
/etc/init.d/apparmor stop
/etc/init.d/apparmor start
```

Also, since Ubuntu DHCP daemon is configured to run as 'dhcpd' and not root, ensure that the following two variables are set as follows in the `$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf` file **on the Cluster head-node**:

```
VNET_DHCPDAEMON="/usr/sbin/dhcpd3"
VNET_DHCPUSER="dhcpd"
```

At this point you should be ready to start Eucalyptus processes on all nodes but before doing so you may want to configure the Eucalyptus network: you can read more about it at Network Configuration.

f. Configure your startup scripts

If you want to have eucalyptus started automatically when your machines are (re)booted, you can add the following symlinks on the appropriate hosts: add `eucalyptus-cloud` on the Cloud head-node, add `eucalyptus-cc` on the Cluster head-node(s), and add `eucalyptus-nc` on the compute node(s)

```
ln -sf $EUCALYPTUS/etc/init.d/eucalyptus-cloud /etc/init.d/eucalyptus-cloud
ln -sf $EUCALYPTUS/etc/init.d/eucalyptus-cc /etc/init.d/eucalyptus-cc
ln -sf $EUCALYPTUS/etc/init.d/eucalyptus-nc /etc/init.d/eucalyptus-nc
```

and then add the symlinks to the distribution's booting process. This process differs from distribution to distribution. For example if you have `update-rc.d` available you can run:

```
update-rc.d eucalyptus-cloud defaults
```

or if you have `chkconfig` available you can run:

```
chkconfig eucalyptus-cloud on
```

6. Running Eucalyptus

Eucalyptus comes with the `euca_conf` script for configuring Eucalyptus. For some requests it modifies the configuration file located in '$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf' (which can also be edited manually), for other requests it modifies the database maintained by the Cloud Controller (much of that functionality is duplicated in the Web interface, to be described later).

In addition to modifying the configuration, `euca_conf` attempts to synchronize x509 credentials across the nodes of a Eucalyptus installation by relying on `rsync` and `scp`. *We highly recommend setting up password-less SSH access for the* `root` *user across all nodes of your Eucalyptus installation* (otherwise, `euca_conf` will prompt you for remote system passwords).

As explained in the overview, a Eucalyptus installation consists of five types of components: cloud controller (CLC), Walrus, cluster controller (CC), storage controller (SC), and the node controller(s) (NCs). In following instructions we assume that all components except the NCs are co-located on a machine that we will refer to as the *front end* and that NCs run on *compute nodes*.

First, make sure that you have all of the runtime dependencies of Eucalyptus installed, based on your chosen set of configuration parameters. If there is a problem with runtime dependencies (for instance, if Eucalyptus cannot find/interact with them), all errors will be reported in log files located in $EUCALYPTUS/var/log/eucalyptus.

Next, inspect the contents of $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf carefully, on each machine, to make sure that the settings are appropriate for your environment.

Once everything is configured properly, enable the cloud services that you wish to run on your front-end, then use the init-scripts to start each component on the appropriate host. Most likely, on the front-end you would run:

```
# enable services on the front-end
$EUCALYPTUS/usr/sbin/euca_conf --enable cloud --enable walrus --enable sc

# start enabled front-end services
$EUCALYPTUS/etc/init.d/eucalyptus-cloud start

# start the cluster controller

$EUCALYPTUS/etc/init.d/eucalyptus-cc start
```

And on each of the compute nodes you would run:

```
$EUCALYPTUS/etc/init.d/eucalyptus-nc start
```

To stop them you call the script with *stop* instead of start.

NOTE: if you later decide to make changes to $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf that will effect the cluster-controller, make sure to use the 'cleanstart', 'cleanstop', and/or 'cleanrestart' directives to the init scripts (as opposed to start/stop/restart). This will both remove all existing CC state, and will cause it to re-read the configuration file.

Installation from distribution-specific binary packages

Choose a linux distribution:

Installing Eucalyptus (1.6) on CentOS 5

Eucalyptus can be installed on CentOS 5 from source or by using binary RPM packages. This document details the steps required to install Eucalyptus from RPMs. In what follows, the value of **$VERSION** must be substituted accordingly (e.g., as **1.6.1**, **1.6.2**, etc.) for example we can set the value of 1.6.2 using bash:

```
export VERSION=1.6.2
```

Notice: Before you begin, please ensure that you have an up-to-date CentOS installation on your target machine(s). In particular, CentOS 5.4, which was recentely released, carries libvirt 0.6.3 which is **required** to run Eucalyptus.

Prerequisites

If you start with a standard CentOS installation, you will satisfy all prerequisites with the following steps:

1. Front-end, node and client machine system clocks are synchronized (e.g., using NTP).

   ```
   yum install -y ntp
   ntpdate pool.ntp.org
   ```

2. Front end needs java, command to manipulate a bridge and the binaries for dhcp server (do not configure it nor run it on the CC):

   ```
   yum install -y java-1.6.0-openjdk ant ant-nodeps dhcp bridge-utils httpd
   ```

3. Node has a fully installed and configured installation of Xen that allows controlling the hypervisor via HTTP from localhost.

   ```
   yum install -y xen
   sed --in-place 's/#(xend-http-server no)/(xend-http-server yes)/' /etc/xen/xend-config.sxp
   sed --in-place 's/#(xend-address localhost)/(xend-address localhost)/' /etc/xen/xend-config.sxp
   /etc/init.d/xend restart
   ```

4. Firewall rules must permit the Eucalyptus components to communicate with one another, and clients to communicate with Eucalyptus. On the front-end, ports 8443, 8773, 8774 and 9001 must be open; on the node, port 8775 must be open. If you are planning on using Elastic IPs and/or Security Groups, you may want to consider disabling the firewall and use Eucalyptus facilities for enabling custom firewall rules (see Network configuration for more information). On both the front-end and the nodes:

   For example, from a text console:

   - run `system-config-securitylevel`
   - select `Security Level: Disabled`
   - select `OK`

   From an X terminal:

   - run `system-config-security-level`
   - select 'Disabled' for 'Firewall'
   - select the 'SELinux' tab
   - select either 'Permissive' or 'Disabled' for SELinux Setting

Download and Install RPMs

Eucalyptus binary installation is broken up into several packages: one for each of the components (CLC, Walrus, CC, etc.), as well as a couple of common packages. There are two options for obtaining the packages:

1. **Yum option:** Packages are available from our yum repository. To use it, create '/etc/yum.repos.d/euca.repo' file with the following four lines:

   ```
   [euca]
   name=Eucalyptus
   baseurl=http://www.eucalyptussoftware.com/downloads/repo/eucalyptus/$VERSION/yum/centos/
   enabled=1
   ```

   where $VERSION is either 1.6.1 or 1.6.2. Now install eucalyptus on the front-end,

   ```
   yum install eucalyptus-cloud.$ARCH eucalyptus-cc.$ARCH eucalyptus-walrus.$ARCH eucalyptus-sc.$ARCH --nogpgcheck
   ```

   or the node

   ```
   yum install eucalyptus-nc.$ARCH --nogpgcheck
   ```

   where $ARCH is the architecture of your host (either 'i386' or 'x86_64').
2. **Tarball option**: The packages are available in a single tarball, wherein we also include copies of third-party CentOS packages that Eucalyptus depends on (Rampart, Axis2C, many Java libraries), at http://www.eucalyptus.com/download/eucalyptus (look for a CentOS tarball of the right Eucalyptus version and architecture). Untar the bundle in a temporary location:

   ```
   tar zxvf eucalyptus-$VERSION-*.tar.gz
   cd eucalyptus-$VERSION-*
   ```

   In the examples below we use `x86_64`, which should be replaced with `i386` or `i586` on 32-bit architectures. Install the third-party dependency RPMs on the front end:

   ```
   cd eucalyptus-$VERSION*-rpm-deps-x86_64

   rpm -Uvh aoetools-21-1.el4.x86_64.rpm \
            euca-axis2c-1.6.0-1.x86_64.rpm \
            euca-rampartc-1.3.0-1.x86_64.rpm \
            vblade-14-1mdv2008.1.x86_64.rpm \
            groovy-1.6.5-1.noarch.rpm \
            vtun-3.0.2-1.el5.rf.x86_64.rpm \
            lzo2-2.02-3.el5.rf.x86_64.rpm
   cd ..
   ```

   Install the -cloud, -walrus, -cc and -sc RPMs on the front end:

   ```
   rpm -Uvh eucalyptus-$VERSION-*.x86_64.rpm \
            eucalyptus-common-java-$VERSION-*.x86_64.rpm \
            eucalyptus-cloud-$VERSION-*.x86_64.rpm \
            eucalyptus-walrus-$VERSION-*.x86_64.rpm \
            eucalyptus-sc-$VERSION-*.x86_64.rpm \
            eucalyptus-cc-$VERSION-*.x86_64.rpm \
            eucalyptus-gl-$VERSION-*.x86_64.rpm
   ```

   Install the dependency packages on each compute node:

   ```
   cd eucalyptus-$VERSION*-rpm-deps-x86_64
   rpm -Uvh aoetools-21-1.el4.x86_64.rpm \
            euca-axis2c-1.6.0-1.x86_64.rpm \
            euca-rampartc-1.3.0-1.x86_64.rpm
   cd ..
   ```

   Install the node controller RPM with dependencies on each compute node:

   ```
   rpm -Uvh eucalyptus-$VERSION-*.x86_64.rpm \
            eucalyptus-gl-$VERSION-*.x86_64.rpm \
            eucalyptus-nc-$VERSION-*.x86_64.rpm
   ```

Post-Install Steps

The last step in the installation is to make sure that the user 'eucalyptus', which is created at RPM installation time, is configured to interact with the hypervisor through libvirt on all of your compute nodes. The easiest way to check this is to run the following command on each node:

```
su eucalyptus -c "virsh list"
```

The output of that command *may* include error messages (`failed to connect to xend`), but as long as it includes a listing of all domains (at least `Domain-0`), the configuration is in order.

Now start up your Eucalyptus services. On the front-end:

```
/etc/init.d/eucalyptus-cloud start
/etc/init.d/eucalyptus-cc start
```

On the node:

```
/etc/init.d/eucalyptus-nc start
```

At this point you should be ready to go through the first-time configuration.

Installing Eucalyptus (1.6) on openSUSE 11

Eucalyptus can be installed on openSUSE 11 from source, or by using binary RPM packages. This document details the steps required to install Eucalyptus from RPMs. In what follows, the value of **$VERSION** must be substituted accordingly (e.g., as **1.6.1**, **1.6.2**, etc.) for example we can set the value of 1.6.2 using bash:

```
export VERSION=1.6.2
```

Prerequisites

If you start with a standard openSUSE installation, you will satisfy all prerequisites with the following steps:

1. Front-end, node and client machine system clocks are syncronized (i.e. using NTP).

   ```
   sntp -P no -r pool.ntp.org
   yast2 -i ntp
   /etc/init.d/ntp restart
   ```

2. Install all dependency packages that are required for Eucalyptus to run on the front-end

   ```
   yast2 -i apache2 apache2-prefork java-1_6_0-openjdk java-1_6_0-openjdk-devel libvirt curl vlan dhcp-server bridge-utils
   ```

   and on the node

   ```
   yast2 -i xen libvirt vlan apache2
   ```

3. Node has a fully installed and configured installation of Xen.
   - RECOMMENDED: verify your Xen installation by manually bringing up a VM and testing that it has network connectivity using bridged networking.
   - To set up bridged networking on your node, use the 'yast2' command and go through the following steps:
     - Network Devices
     - Network Settings
     - Select 'OK' to get past information box
     - Traditional Method with ifup
     - Overview
     - Add
     - Device Type: Bridge
     - Next
     - Bridged Devices: select eth0 (or the name of your primary interface)
     - Next
     - Continue
     - Ok
   - make sure that the libvirt daemon (libvirtd) is running and configured properly
     - /etc/init.d/libvirtd start
     - check eucalyptus can interact with libvirt

       ```
       su eucalyptus -c "virsh list"
       ```

       you may see this error (which could show up in the logs too)

       ```
       Attempting to obtain authorization for org.libvirt.unix.manage.
       polkit-grant-helper: given auth type (8 -> yes) is bogus
       Failed to obtain authorization for org.libvirt.unix.manage.
       ```

       which is harmless.

4. Firewall rules must permit the Eucalyptus components to communicate with one another, and clients to communicate with Eucalyptus.
   - NOTE: on the front-end, ports 8443, 8773, 8774 and 9001 must be open. On the node, port 8775 must be open
   - If you are planning on using Elastic IPs and/or Security Groups, you may want to consider disabling the firewall and use Eucalyptus facilities for enabling custom firewall rules (see Network configuration for more information).
     - yast2 firewall startup manual
     - /etc/init.d/SuSEfirewall2_init stop
     - reboot

Download and Install RPMs

Eucalyptus binary installation is broken up into several packages: one for each of the components (CLC, Walrus, CC, etc.), as well as a couple of common packages. There are two options for obtaining the packages:

1. **Zypper option:** packages are available from our repository. To use it:

   ```
   zypper ar --refresh http://www.eucalyptussoftware.com/downloads/repo/eucalyptus/$VERSION/yum/opensuse Eucalyptus
   ```

   answer question about trusting packages from this repository then refresh it

   ```
   zypper refresh Eucalyptus
   ```

   and now install eucalyptus on the front-end

   ```
   zypper install eucalyptus-cloud eucalyptus-cc eucalyptus-walrus eucalyptus-sc
   ```

   or the node

   ```
   zypper install eucalyptus-nc
   ```

2. **Tarball option:** the packages are available in a single tarball, wherein we also include copies of third-party openSUSE packages that Eucalyptus depends on (Rampart, Axis2C, many Java libraries), at http://www.eucalyptus.com/download/eucalyptus (look for a openSUSE tarball of the right Eucalyptus version and architecture). Untar the bundle in a temporary location:

   ```
   tar zxvf eucalyptus-$VERSION-*.tar.gz
   cd eucalyptus-$VERSION-*
   ```

   In the examples below we use `x86_64`, which should be replaced with `i386` or `i586` on 32-bit architectures. Install the third-party dependency RPMs on the front end:

   ```
   cd eucalyptus-$VERSION*-rpm-deps-x86_64
   rpm -Uvh aoetools-21-1.el4.x86_64.rpm \
            euca-axis2c-1.6.0-1.x86_64.rpm \
            euca-rampartc-1.3.0-1.x86_64.rpm \
            vblade-14-1mdv2008.1.x86_64.rpm \
            groovy-1.6.5-1.noarch.rpm \
            vtun-3.0.1-1.x86_64.rpm
   cd ..
   ```

   Install the -cloud, -walrus, -cc and -sc RPMs on the front end:

   ```
   rpm -Uvh eucalyptus-$VERSION-*.x86_64.rpm \
            eucalyptus-common-java-$VERSION-*.x86_64.rpm \
            eucalyptus-cloud-$VERSION-*.x86_64.rpm \
            eucalyptus-sc-$VERSION-*.x86_64.rpm \
            eucalyptus-walrus-$VERSION-*.x86_64.rpm \
            eucalyptus-cc-$VERSION-*.x86_64.rpm \
            eucalyptus-gl-$VERSION-*.x86_64.rpm
   ```

Install RPMs on the nodes

Install the dependency packages on each node:

```
cd eucalyptus-$VERSION*-rpm-deps-x86_64
rpm -Uvh aoetools-25-2.49.x86_64.rpm \
         euca-axis2c-1.6.0-1.x86_64.rpm \
         euca-rampartc-1.3.0-1.x86_64.rpm \
         vblade-15-2.49.x86_64.rpm
cd ..
```

On the compute nodes, install the node controller RPM with dependencies:

```
rpm -Uvh eucalyptus-$VERSION-*.x86_64.rpm \
         eucalyptus-gl-$VERSION-*.x86_64.rpm \
         eucalyptus-nc-$VERSION-*.x86_64.rpm
```

Post-Install Steps

Now start up your Eucalyptus services. On the front-end:

```
/etc/init.d/eucalyptus-cloud start
/etc/init.d/eucalyptus-cc start
```

On the node:

```
/etc/init.d/eucalyptus-nc start
```

Installing Eucalyptus (1.6) on Debian "squeeze"

Eucalyptus can be installed on Debian squeeze using binary DEB packages. Squeeze has not been released yet, so things can change quickly and without warning.

Download DEBs

Eucalyptus binary installation is broken up into several packages: one for each of the components (CLC, Walrus, CC, etc.), as well as a couple of common packages. To install them, along with a *significant* number of dependencies, add our repository to the list of repositories for your system to use. To do so, add somewhere in `/etc/apt/sources.list` file the following line:

For 1.6.1:

```
deb http://eucalyptussoftware.com/downloads/repo/eucalyptus/1.6.1/debian/ squeeze contrib
```

For 1.6.2 (including release candidates):

```
deb http://eucalyptussoftware.com/downloads/repo/eucalyptus/1.6.2/debian/ squeeze main
```

And then run:

```
apt-get update
```

After installation you may remove the entry from `sources.list` if you don't want to update Eucalyptus packages automatically.

Prerequisites

If you start with a standard Debian squeeze installation, you will satisfy all Eucalyptus prerequisites with the following steps:

1. Synchronize clocks (e.g., using NTP: `ntpdate pool.ntp.org`) across all Eucalyptus machines and client machines.
2. If using a firewall, permit the Eucalyptus components to communicate with one another, and permit clients to communicate with Eucalyptus.
    - NOTE: on the front-end, ports 8443, 8773, 8774 and 9001 must be open. On the node, port 8775 must be open.
3. Ensure that `locales` are properly configured (use `dpkg-reconfigure locales`).
4. Install libvirt/qemu-kvm and configure it to run as user 'eucalyptus':

```
sudo apt-get install libvirt-bin qemu-kvm
# set the field user to be: user = "eucalyptus"
sudo vi /etc/libvirt/qemu.conf
# restart libvirt
sudo /etc/init.d/libvirt-bin restart
```

4. If running in SYSTEM networking mode, which is the default, your node machine(s) must be configured with a bridge as the primary interface. For example, you may try:

```
sudo apt-get install bridge-utils
sudo vi /etc/network/interfaces
```

Comment out any entry for your existing interfaces (eth0, eth1, etc) and add a bridge entry with your interfaces attached. For example, to have your bridge come up with all physical Ethernet devices added to it, and have DHCP assign an address to the bridge, use:

```
auto br0
iface br0 inet dhcp
      bridge_ports all
```

For a static configuration with just eth0 attached (substitute your actual network parameters):

```
auto br0
iface br0 inet static
      address 192.168.12.20
      netmask 255.255.255.0
      network 192.168.12.0
      broadcast 192.168.12.255
      gateway 192.168.12.1
      dns-nameservers 192.168.12.1
      dns-search foobar foobar.com
      bridge_ports eth0
```

Finally, restart the network by either by restarting the network with '/etc/init.d/networking restart' or by rebooting the machine.

Install DEBs on the front end

On front end, where cloud controller, Walrus, cluster controller, and storage controller will run, install the appropriate DEBs:

```
aptitude install eucalyptus-common eucalyptus-cloud eucalyptus-walrus eucalyptus-sc eucalyptus-cc
```

Install DEBs on the nodes

On the compute nodes, install the node-controller DEB:

```
aptitude install eucalyptus-nc
```

(You may safely ignore the error `adduser: The group 'libvirtd' does not exist.`)

Installing Eucalyptus (1.6.1) on Ubuntu Jaunty (9.04)

Eucalyptus 1.6.1 **(no longer the most current stable release)** can be installed on Ubuntu Jaunty using binary DEB packages. (Ubuntu Lucid users can install the latest release from the standard Ubuntu supported repository, or users of any Ubuntu release from Jaunty and later can always install Eucalyptus from source.)

Download DEBs

Eucalyptus binary installation is broken up into several packages: one for each of the components (CLC, Walrus, CC, etc.), as well as a couple of common packages. To install them, along with a *significant* number of dependencies, add our repository to the list of repositories for your system to use. To do so, add somewhere in `/etc/apt/sources.list` file the following line:

```
deb http://www.eucalyptussoftware.com/downloads/repo/eucalyptus/1.6.1/ubuntu jaunty universe
```

And run:

```
apt-get update
```

After installation you may remove the entry from `sources.list` if you don't want to update Eucalyptus packages automatically.

Prerequisites

If you start with a standard Ubuntu Jaunty installation, you will satisfy all prerequisites with the following steps:

1. Front-end, node and client machine system clocks are syncronized (i.e., using NTP).

   ```
   ntpdate-debian -s
   apt-get install openntpd
   ```

2. Firewall rules must permit the Eucalyptus components to communicate with one another, and clients to communicate with Eucalyptus.
   - NOTE: on the front-end, ports 8443, 8773, 8774 and 9001 must be open. On the node, port 8775 must be open

3. If running in SYSTEM networking mode, which is the default, your node machine(s) must be configured with a bridge as the primary interface. You must first uninstall or disable Network Manager (default with Ubuntu Desktop), then follow the procedure below (example):

   ```
   sudo apt-get install bridge-utils
   sudo vi /etc/network/interfaces
   ```

   Comment out any entry for your existing interfaces (eth0, eth1, etc) and add a bridge entry with your interfaces attached. For example, to have your bridge come up with all physical Ethernet devices added to it, and have DHCP assign an address to the bridge:

   ```
   auto br0
   iface br0 inet dhcp
         bridge_ports all
   ```

   For a static configuration with just eth0 attached (substitute your actual network parameters):

   ```
   auto br0
   iface br0 inet static
         address 192.168.12.20
         netmask 255.255.255.0
         network 192.168.12.0
         broadcast 192.168.12.255
         gateway 192.168.12.1
         dns-nameservers 192.168.12.1
         dns-search foobar foobar.com
         bridge_ports eth0
   ```

   Finally, restart the network by either by restarting the network with '/etc/init.d/network restart' or by rebooting the machine.

Install DEBs on the front end

On front end, where cloud controller and cluster controller will run, install the -cloud and -cc DEBs:

```
aptitude install eucalyptus-cc eucalyptus-cloud eucalyptus-common eucalyptus-walrus eucalyptus-sc
```

Install DEBs on the nodes

On the compute nodes, install the node controller DEB:

```
aptitude install eucalyptus-nc eucalyptus-common
```

(You may safely ignore the error `adduser: The group 'libvirt' does not exist.`)

Upgrading to Eucalyptus 1.6.2 from 1.6.1

These instructions are for those who would like to upgrade to Eucalyptus 1.6.2 from a source-based or package-based 1.6.1 installation. If you're still running 1.5.2, please, follow the instructions for upgrading to 1.6.1 before following these instructions. Instructions below assume that **$EUCALYPTUS** points to the root of the new Eucalyptus installation and **$OLD_EUCA** points to the root of the old installation (which can be the same as the new one).

1. Stop and back up your current installation

We highly recommend backing up your installation before performing an upgrade. The general approach to backup is outlined in the first part of the Backup section. Starting from Eucalyptus 1.6, a script called **euca_upgrade** can be used to perform partial backups by creating copies of the configuration file, database, and keys (it does not back up buckets and volumes because of their potentially large disk space requirements and because they are unlikely to be harmed during an upgrade).

- If you are using our binary packages (RPMs or DEBs), euca_upgrade will be invoked automatically, creating backups in
    - /root/eucalyptus.backup.$TIMESTAMP.
- If you are upgrading a source-based installation, you must still invoke euca_upgrade, as shown below, even if you've backed up your installation using some other method.
- Regardless of whether you choose to back up, be sure to stop all Eucalyptus processes on all machines before proceeding.

2. Install Eucalyptus 1.6

- If upgrading using **binary packages**, follow the steps in the installation instruction for a specific distribution:
    - CentOS 5.4
    - OpenSUSE 11
    - Debian "Squeeze" (see the warning)

and, afterwards, **return here**.

- If upgrading a **source-based** installation, follow the steps in the Source Code Installation section of the Administrator's Guide and, afterwards, **return here**. To upgrade the front-end, run:

  ```
  $EUCALYPTUS/usr/share/eucalyptus/euca_upgrade --new $EUCALYPTUS --old $OLD_EUCA --conf --keys --db
  ```

  and to upgrade the nodes run

  ```
  $EUCALYPTUS/usr/share/eucalyptus/euca_upgrade --new $EUCALYPTUS --old $OLD_EUCA --conf --keys
  ```

3. Restart Eucalyptus and verify the upgrade

- Start the services on the appropriate machines (after a DEB-based install they should already be running):
  ```
  $EUCALYPTUS/etc/init.d/eucalyptus-nc start
  $EUCALYPTUS/etc/init.d/eucalyptus-cc cleanstart
  $EUCALYPTUS/etc/init.d/eucalyptus-cloud start
  ```
- In a Web browser, load https://front-end:8443/ and log in as before. Verify that the user accounts and the images are there.
- **Important:** Verify that the Buckets Path and Volumes Path settings under the Configuration tab of the Web interface matches the actual locations of the buckets and volumes before running any instances or using buckets.
- Verify that the nodes are back up and that they can run your old instances (if not, see the Troubleshooting section.)
  ```
  euca-describe-availability-zones verbose
  ```

4. Optionally: Roll back to an earlier installation

- Follow the steps in the second part of the Backup section, called "Restoration". If you are relying on the backup created by euca_upgrade during a package-based upgrade, then after re-installing the old packages, copy back the saved state (the backed up copies of db/*, keys/*, etc/eucalyptus/eucalyptus.conf) to your restored installation. Then, start Eucalytpus, as before.

Upgrading to Eucalyptus 1.6.1 from 1.5.2

These instructions are for those who would like to upgrade to Eucalyptus 1.6.1 from a source-based or package-based 1.5.2 installation. If you're still running 1.4, please, follow the instructions for upgrading to 1.5.2 before following these instructions. Instructions below assume that **$EUCALYPTUS** points to the root of the new Eucalyptus installation and **$EUCA_OLD** points to the root of the old installation (which can be the same as the new one).

1. Stop and back up your current installation

We highly recommend backing up your installation before performing an upgrade. The general approach to backup is outlined in the first part of the Backup section. Starting from Eucalyptus 1.6, a script called **euca_upgrade** can be used to perform partial backups by creating copies of the configuration file, database, and keys (it *does not* back up buckets and volumes because of their potentially large disk space requirements and because they are unlikely to be harmed during an upgrade).

- If you are using our binary packages (RPMs or DEBs), `euca_upgrade` will be invoked automatically, creating backups in
    - `$EUCALYPTUS/var/lib/eucalyptus/db/1.5`,
    - `$EUCALYPTUS/var/lib/eucalyptus/keys/1.5`, and
    - `$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf-1.5`.
- If you are upgrading a source-based installation, you must still invoke `euca_upgrade`, as shown below, even if you've backed up your installation using some other method.
- Regardless of whether you choose to back up, be sure to stop all Eucalyptus processes on all machines before proceeding.

2. Install Eucalyptus 1.6

- If upgrading using **binary packages**, follow the steps in the installation instruction for a specific distribution:
    - CentOS 5.4
    - OpenSUSE 11
    - Debian "Squeeze" (see the warning)
    - Ubuntu "Jaunty" 9.04

  and, afterwards, **return here**.

- If upgrading a **source-based** installation, follow the steps in the Source Code Installation section of the Administrator's Guide and, afterwards, **return here**. To upgrade the front-end, run

  ```
  $EUCALYPTUS/usr/share/eucalyptus/euca_upgrade --new $EUCALYPTUS --old $OLD_EUCA
  ```

  and to upgrade the nodes run

  ```
  $EUCALYPTUS/usr/share/eucalyptus/euca_upgrade --new $EUCALYPTUS --old $OLD_EUCA --conf
  ```

3. Restart Eucalyptus and verify the upgrade

- Start the services on the appropriate machines (after a DEB-based install they should already be running):

  ```
  $EUCALYPTUS/etc/init.d/eucalyptus-nc start
  $EUCALYPTUS/etc/init.d/eucalyptus-cc start
  $EUCALYPTUS/etc/init.d/eucalyptus-cloud start
  ```

- In a Web browser, load `https://front-end:8443/` and log in as before. Verify that the user accounts and the images are there.
- **Important:** Verify that the Buckets Path and Volumes Path settings under the Configuration tab of the Web interface matches the actual locations of the buckets and volumes *before* running any instances or using buckets.
- Verify that the nodes are back up and that they can run your old instances (if not, see the Troubleshooting section.)

  ```
  euca-describe-availability-zones verbose
  ```

4. Optionally: Roll back to an earlier installation

- Follow the steps in the second part of the Backup section, called "Restoration". If you are relying on the backup created by `euca_upgrade` during a package-based upgrade, then after re-installing the old packages, copy back the saved state as follows:

  ```
  cp -a $EUCALYPTUS/etc/eucalyptus.conf-1.5 $OLD_EUCA/etc/eucalyptus.conf
  cp -a $ECUALYPTUS/var/lib/eucalyptus/db/1.5/* $OLD_EUCA/var/lib/eucalyptus/db
  cp -a $EUCALYPTUS/var/lib/eucalyptus/keys/1.5/* $OLD_EUCA/var/lib/eucalyptus/keys
  ```

- Start Eucalyptus, as before

Configuration

First-time Setup (1.6)

This document describes the steps for activating and possibly further configuring Eucalyptus after the software has been installed on all nodes (either from source or using binary packages).

After you've started all components, you will need to perform registration so that they can communicate with each other.

Registering Eucalyptus Components

This section will assume that you have installed all Eucalyptus components and they are up and running. We will assume that your Eucalyptus setup consists of one front end and one or more nodes.

First, you will need to register various front end components. To do this, run the following commands on the front end.

```
$EUCALYPTUS/usr/sbin/euca_conf --register-walrus <front end IP address>

$EUCALYPTUS/usr/sbin/euca_conf --register-cluster <clustername> <front end IP address>

$EUCALYPTUS/usr/sbin/euca_conf --register-sc <clustername> <front end IP address>
```

Finally, you need to register nodes with the front end. To do so, run the following command on the front end,

```
$EUCALYPTUS/usr/sbin/euca_conf --register-nodes "<Node 0 IP address> <Node 1 IP address> ... <Node N IP address>"
```

where "<Node X IP address>" is the IP address of host X that is running the Node Controller (NC).

At this point, you have successfully registered Eucalyptus components and you are ready to proceed to configuration.

First-time Configuration

Point your browser to,

https://front-end-ip:8443

Since Eucalyptus is using a self-signed certificate, your browser is likely to prompt you to accept the certificate. On some machines it may take few minutes after the starting of the Cloud Controller for the URL to be responsive the first time you run Eucalyptus. You will be prompted for a user and password both of which are set to `admin` initially.

Upon logging in the first time you will be asked to

1. change the admin password,
2. set the admin's email address, and
3. confirm the IP of the Cloud Controller host.

After clicking 'Submit', you will see the 'Configuration' tab. Since you've used `euca_conf` to register Walrus and a cluster, they will be listed along with a few configurable parameters. Look over the parameters to see if any need adjustment. For more information, see the Management section.

To use the system with client tools, you must obtain user credentials. From the 'Credentials' tab, Eucalyptus users can obtain two types of credentials: x509 certificates and query interface credentials. Use the 'Download Credentials' button to download a zip-file with both or click on the 'Show Keys' to see the query interface credentials. You will be able to use your credentials with Euca2ools, Amazon EC2 tools and third-party tools like rightscale.com. Create a directory to store your credentials, unpack the zip-file into it, and source the included 'eucarc':

```
mkdir $HOME/.euca
unzip euca2-admin-x509.zip -d $HOME/.euca
. $HOME/.euca/eucarc
```

Note that you will have to source this file every time you intend to use the command-line tools, or you may add it to your local default environment.

Hypervisor Configuration

Eucalyptus deploys instances (i.e., virtual machines) on a hypervisor. Eucalyptus can use either xen or kvm hypervisors. To interact with them, Eucalyptus employs libvirt virtualization API. The best choice for the hypervisor depends on its support for your hardware, on the support for the hypervisor in your OS (some distros support KVM better, some support Xen better), as well as personal preferences.

Another consideration is support for Eucalyptus features in the hypervisor. Because Eucalyptus uses features that only recently have been added to hypervisors, some combinations of hypervisor and kernel do not function as intended. The most common problem we encounter has to do with support for attaching and removing block devices. On some kernels, for example, you may see a lot of WARN_ON messages in the logs (similar to kernel oops), with KVM you will not be able to specify the exact device block (it will be chosen by the system), and on some hypervisor-kernel combinations EBS will not work at all (e.g., Debian "squeeze" with 2.6.30-2-amd64 kernel and KVM v88).

Running a test VM with hypervisor tools

First of all, before even installing Eucalyptus, install a hypervisor of your choice and, based on the hypervisor's documentation, try to construct and run a test VM from the command line.(If you cannot run a VM *outside* Eucalyptus, you will not be able to run any VMs *through* Eucalyptus.)

Running a Xen VM usually involves creating a configuration file and passing it to the `xm create` command. Running a KVM VM usually involves invoking `kvm` with many parameters on the command-line.

If the hypervisor doesn't work out of the box on your distro, you may want to experiment with options. For Xen, the options are specified in:

```
/etc/xend/xend-config.sxp
```

We had good luck with these:

```
(xend-http-server yes)
(xend-unix-server yes)
(xend-unix-path /var/lib/xend/xend-socket)
(xend-address localhost)
(network-script network-bridge)
(vif-script vif-bridge)
(dom0-min-mem 196)
(dom0-cpus 0)
(vncpasswd '')
```

Running a test VM with libvirt's virsh

Since Eucalyptus interacts with hypervisors through libvirt, it is also a good idea to ensure that libvirt is set up properly, particularly for user "eucalyptus". A way to do so is to try

```
virsh list
```

as the "eucalyptus" user (root usually can always connect). If that fails, the solutions are distribution-specific: for example, on some Debian-based distros, the user "eucalyptus" needs to be in the group **libvirt** or **libvirtd**.

On distros using **PolicyKit**, you may want to ensure that in

```
/etc/PolicyKit/PolicyKit.conf
```

there is something like

```
<config version="0.1">
<match action="org.libvirt.unix.manage">
   <match user="eucalyptus">
      <return result="yes"/>
   </match>
</match>
</config>
```

As the last resort, you may want to look into

```
/etc/libvirt/libvirtd.conf
```

and keep an eye on logs in

```
/var/log/libvirt
```

Backup of Eucalyptus (1.6)

Backing up and restoring a Eucalyptus installation involves saving and restoring the contents of *five file-system locations*. Three are on the CLC machine:

- The **configuration file** ($EUCALYPTUS/etc/eucalyptus.conf)
- The **database files** ($EUCALYPTUS/var/lib/eucalyptus/db)
- The **cryptographic keys** ($EUCALYPTUS/var/lib/eucalyptus/keys)

One on the Walrus machine (which is the same as CLC machine in a single-cluster installation):

- The **Walrus buckets** ("Buckets path" in Web configuration, by default $EUCALYPTUS/var/lib/eucalyptus/bukkits)

And one on each of the cluster head nodes (again, same as the CLC machine in a single-cluster installation):

- The **SC volumes** ("Volumes path" in Web configuration, by default $EUCALYPTUS/var/lib/eucalyptus/volumes)

If the files at these locations are backed up, a Eucalyptus installation can be fully restored after a crash or a failed upgrade. What follows is a step-by-step guide to backup and restoration.

Part I: Backup

1. Clean up Eucalyptus running state

- Note the value of the "Buckets path" and "Volumes path" for each cluster, listed under the "Configuration" tab of the Web interface. (That is where all uploaded images, user buckets, user volumes, and snapshots are located.)

- Terminate **all** Eucalyptus instances on **all** nodes

  ```
  euca-terminate-instances ...      # (as admin)
  ```

- Shut down all Eucalyptus components on **all** nodes, issuing the commands relevant for a node:

  ```
  $EUCALYPTUS/etc/init.d/eucalyptus-nc stop
  $EUCALYPTUS/etc/init.d/eucalyptus-cc stop
  $EUCALYPTUS/etc/init.d/eucalyptus-cloud stop
  ```

- Check for errant Eucalyptus processes on **all** nodes and kill them

  ```
  ps aux | grep euca
  kill -9 ...
  ```

2. Back up the current installation

- Calculate the disk space required to store the files about to be backed up (this is most relevant for buckets and volumes, which can be large). E.g., on a single-cluster installation with default Buckets and Volumes paths:

  ```
  du -sh $EUCALYPTUS/var/lib/eucalyptus/
  ```

- Create a directory for storing these ($BACKUP) on a volume with enough disk space

  ```
  export BACKUP=/path/to/backup/directory
  mkdir -p $BACKUP
  ```

- Mirror the five locations, taking care to preserve the permissions on all files. E.g., on a single-cluster installation with default Buckets and Volumes paths:

  ```
  cp -a $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf \
  $EUCALYPTUS/var/lib/eucalyptus/keys \
  $EUCALYPTUS/var/lib/eucalyptus/db \
  $EUCALYPTUS/var/lib/eucalyptus/bukkits \
  $EUCALYPTUS/var/lib/eucalyptus/volumes \
  $BACKUP
  ```

  or, alternatively, with `tar`:

  ```
  tar cvf $BACKUP/eucalyptus-backup.tar \
  $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf \
  $EUCALYPTUS/var/lib/eucalyptus/keys \
  $EUCALYPTUS/var/lib/eucalyptus/db \
  $EUCALYPTUS/var/lib/eucalyptus/bukkits \
  $EUCALYPTUS/var/lib/eucalyptus/volumes
  ```

  In either case, be careful if any of the above are symbolic links as they may be copied instead of the directories they point to. Check that the backup indeed contains files from the original locations.

Part II: Restoration

1. Clean up Eucalyptus running state

Same as in the Backup step, make sure no Eucalyptus components are running on any of the nodes

2. Optionally update/downgrade Eucalyptus-related binary packages

If you are trying to recover from a broken upgrade by rolling back or by trying the upgrade again, this would be the right time to

- remove all software components related to Eucalyptus (e.g., `rpm -e` or `apt-get remove`) and
- install the appropriate version by following the instructions in the Installation section.

Warning: DEBs will restart the services: be sure you stop them again before copying back the backed-up files.

3. Replace the saved state

- Depending on how you backed up, copy the files back either with `cp`:

```
cp -a $BACKUP/eucalyptus.conf $EUCALYPTUS/etc/eucalyptus
cp -a $BACKUP/keys $BACKUP/db $BACKUP/bukkits $BACKUP/volumes $EUCALYPTUS/var/lib/eucalyptus
```

  or with `tar`:

```
cd $EUCALYPTUS
tar xvf $BACKUP/eucalyptus-backup.tar
```

Eucalyptus Network Configuration (1.6)

Eucalyptus versions 1.5 and higher include a highly configurable VM networking subsystem that can be adapted to a variety of network environments. There are four high level networking "modes", each with its own set of configuration parameters, features, benefits and in some cases restrictions placed on your local network setup. The administrator must select one of these four modes before starting Eucalyptus on the front-end and nodes via modification of the 'eucalyptus.conf' configuration file on each machine running a Eucalyptus component. Brief descriptions of each mode follows:

> **SYSTEM Mode** - This is the simplest networking mode, but also offers the smallest number of networking features. In this mode, Eucalyptus simply assigns a random MAC address to the VM instance before booting and attaches the VM instance's ethernet device to the physical ethernet through the node's local Xen bridge. VM instances typically obtain an IP address using DHCP, the same way any non-VM machine using DHCP would obtain an address. Note that in this mode, the Eucalyptus administrator (or the administrator that manages the network to which Eucalyptus components are attached) must set up a DHCP server that has a dynamic pool of IP addresses to hand out as VMs boot. In other words, if your laptop/desktop/server gets an IP address using DHCP on the same network as the Eucalyptus nodes, then your VMs should similarly obtain addresses. This mode is most useful for users who want to try out Eucalyptus on their laptops/desktops.
>
> **STATIC Mode** - This mode offers the Eucalyptus administrator more control over VM IP address assignment. Here, the administrator configures Eucalyptus with a 'map' of MAC address/IP Address pairs. When a VM is instantiated, Eucalyptus sets up a static entry within a Eucalyptus controlled DHCP server, takes the next free MAC/IP pair, assigns it to a VM, and attaches the VMs ethernet device to the physical ethernet through the Xen bridge on the nodes (in a manner similar to SYSTEM mode). This mode is useful for administrators who have a pool of MAC/IP addresses that they wish to always assign to their VMs.
>
> **NOTE** - Running Eucalyptus in SYSTEM or STATIC mode disables some key functionality such as the definition of ingress rules between collections of VMs (termed security groups in Amazon EC2), the user-controlled, dynamic assignment of IPs to instances at boot and run-time (elastic IPs in Amazon EC2), isolation of network traffic between VMs (that is, the root user within VMs will be able to inspect and potentially interfere with network traffic from other VMs), and the availability of the meta-data service (use of the http://169.254.169.254/ URL to obtain instance specific information).
>
> **MANAGED Mode** - This mode is the most featureful of the three modes, but also carries with it the most potential constraints on the setup of the Eucalyptus administrator's network. In MANAGED mode, the Eucalyptus administrator defines a large network (usually private, unroutable) from which VM instances will draw their IP addresses. As with STATIC mode, Eucalyptus will maintain a DHCP server with static mappings for each VM instance that is created. Eucalyptus users can define a number of 'named networks', or 'security groups', to which they can apply network ingress rules that apply to any VM that runs within that 'network'. When a user runs a VM instance, they specify the name of such a network that a VM is to be a member of, and Eucalyptus selects a subset of the entire range of IPs that other VMs in the same 'network' can reside. A user can specify ingress rules that apply to a given 'network', such as allowing ping (ICMP) or ssh (TCP, port 22) traffic to reach their VMs. This capability allows Eucalyptus expose a capability similar to Amazon's 'security groups'. In addition, the administrator can specify a pool of public IP addresses that users may allocate, then assign to VMs either at boot or dynamically at run-time. This capability is similar to Amazon's 'elastic IPs'. Eucalyptus administrators that require security groups, elastic IPs, and VM network isolation must use this mode.
>
> **MANAGED-NOVLAN Mode** - This mode is identical to MANAGED mode in terms of features (dynamic IPs and security

groups) but does not provide VM network isolation. Admins who want dynamic assignable IPs and the security groups, but are not running on a network that is 'VLAN clean' or don't care if their VMs are isolated from one another on the network should choose this mode.

Each Eucalyptus network mode has its own set of infrastructure requirements, configuration parameters, and caveats. These are described in more detail in the following sections.

Bridges

For some of the network modes, you'll be required to set up an Ethernet bridge in order for the network mode to function. If you use Xen, the distros typically set up a bridge for you, and you'll simply have to find its name. For Xen versions 3.0 or earlier the bridge name is typically

```
xenbr0
```

while if you use Xen 3.2 the bridge name is typically

```
eth0
```

If you use KVM, or you wish to configure a bridge manually, the following describes how to set up a bridge on various distributions. In these examples, it is assumed that your bridge device will obtain its IP address using DHCP, and that your physical ethernet device is named 'eth0'.

Centos

```
# create a new ethernet bridge configuration file '/etc/sysconfig/network-scripts/ifcfg-br0'
# and populate it with the following

DEVICE=br0
BOOTPROTO=dhcp
ONBOOT=yes
TYPE=Bridge

# add your physical ethernet device to the bridge by editing  your physical ethernet device
# configuration file (in this example, 'eth0') '/etc/sysconfig/network-scripts/ifcfg-eth0'

DEVICE=eth0
TYPE=Ethernet
BRIDGE=br0
```

OpenSUSE

```
# All network configuration is done using the 'yast2' configuration tool
Run yast2
Network Devices
Network Settings
Add
Device Type->Bridge
Next
Bridged Devices->select your pysical device (eth0)
Next
Ok
Quit
```

Ubuntu and Debian

```
# create a new ethernet bridge device and attach your physical
# ethernet device to the bridge by adding the following to '/etc/network/interfaces'

auto br0
iface br0 inet dhcp
      bridge_hello 2
      bridge_fd 1
      bridge_ports eth0

# and comment out your settings for 'eth0' in the same file
```

When you are finished configuring a bridge, it is advised that you restart the machine (or, at least, restart the networking subsystem). Once it is back up and running, the

```
brctl show
```

command will list all available bridges, which you can use to check that your system is properly configured to run Eucalyptus.

NOTE: the bridge name

```
virbr0
```

is created by libvirt is shouldn't not be used.

For the reminder of this document, we assume that you correctly identified the bridge and that such bridge is called

```
br0
```

SYSTEM Mode

There is very little Eucalyptus configuration to use SYSTEM mode, as in this mode, Eucalyptus mostly stays 'out of the way' in terms of VM networking. The options in 'eucalyptus.conf' that must be configured correctly in 'SYSTEM' mode are as follows:

On the front-end:

```
VNET_MODE="SYSTEM"
```

On each node:

```
VNET_MODE="SYSTEM"
VNET_BRIDGE
```

In each Eucalyptus node controller's (NC) 'eucalyptus.conf' file, make sure that the parameter 'VNET_BRIDGE' is set to the name of the bridge device that is connected to your local ethernet

```
VNET_BRIDGE="br0"
```

Make sure that what you are specifying in this field is actually a bridge, and that it is the bridge that is connected to an ethernet network that has a DHCP server running elsewhere that is configured to hand out IP addresses dynamically. Note that your front-end machine does not need to have any bridges (this is fine, as VNET_BRIDGE is only a relevant for node controllers, and will be safely ignored by the front-end components).

To test whether this mode is working properly at run-time, you can check on a node before and after an instance is running the configure bridge. You should see a new interface associate with the bridge for example you could see

```
; brctl show
bridge name  bridge id          STP enabled     interfaces
eth0         8000.000c29369858  no              peth0
                                                vif18.0
```

on a node controller running Xen 3.2: note that Eucalyptus has correctly attached the VM's 'eth0' interface (vif18.0) to the bridge ('br0') that is being used to attach VMs to the local ethernet ('peth0').

In the case of kvm you may see something like

```
; brctl show
bridge name bridge id  STP enabled interfaces
br0  8000.00005a00083d no  eth0
                                                        vnet0
```

At this point, the VM should be sending DHCP requests to the local ethernet, and the DHCP server on the network should be sending a reply.

CAVEATS - In this mode, as mentioned previously, VMs are simply started with their ethernet interfaces attached to the local ethernet without any isolation. Practically, this means that you should treat a VM the same way that you would treat a non-VM machine running on the network. Eucalyptus does it's best to discover the IP address that was assigned to a running VM via a third-party DHCP server, but can be unsuccessful depending on the specifics of your network (switch types/configuration, location of CC on the network, etc.). Practically, if Eucalyptus cannot determine the VM's IP, then the user will see '0.0.0.0' in the output of 'describe-instances' in both the private and public address fields. The best workaround for this condition is to instrument your VMs to send some network traffic to your front end on boot (after they obtain an IP address). For instance, setting up your VM to ping the front-end a few times on boot should allow Eucalyptus to be able to discover the VMs IP.

STATIC Mode

In this mode, Eucalyptus will manage VM IP address assignment by maintaining its own DHCP server with one static entry per VM. The options in 'eucalyptus.conf' that must be configured correctly in 'STATIC' mode are as follows:

On the front-end (options annotated with a '*' may be required depending on your installation, see below for details):

```
VNET_MODE="STATIC"
VNET_PUBINTERFACE
VNET_PRIVINTERFACE
VNET_DHCPDAEMON
*VNET_DHCPUSER
VNET_SUBNET
```

```
VNET_NETMASK
VNET_BROADCAST
VNET_ROUTER
VNET_DNS
VNET_MACMAP
```

On each node:

```
VNET_MODE="STATIC"
VNET_BRIDGE
```

The Eucalyptus administrator must configure the front-end's 'eucalyptus.conf' first with a valid, configured ethernet device that is attached to the same physical ethernet as the Eucalyptus nodes:

```
VNET_PRIVINTERFACE="eth0"
```

If the front-end has a second ethernet device which is used to access the public network, let's say eth1, then you need to configured it as

```
VNET_PUBINTERFACE="eth1"
```

Next, the admin must ensure that there is a DHCP server binary installed on the front-end and Eucalyptus knows where it is located:

```
VNET_DHCPDAEMON="/usr/sbin/dhcpd3"
```

If your DHCP daemon binary is configured to run as 'non-root' (say, as the user 'dhcpd' as is the case in Ubuntu >= 8.10), then you must configure Eucalyptus to be aware of that user:

```
VNET_DHCPUSER="<dhcpusername>"
```

Then, the admin must input IP subnet information for that device. For example, if the front-end's 'eth0' interface has the IP address '192.168.1.254' on the '192.168.1.0/24' network, with a gateway at '192.168.1.1' and a DNS at '192.168.1.2', the values in 'eucalyptus.conf' would look like so:

```
VNET_SUBNET="192.168.1.0"
VNET_NETMASK="255.255.255.0"
VNET_BROADCAST="192.168.1.255"
VNET_ROUTER="192.168.1.1"
VNET_DNS="192.168.1.2"
```

Finally, the administrator must supply a list of static MAC/IP mappings that will be assigned, first come first served, to VM instances. Note that each IP must reside in the subnet defined above, and must not be in use by any other machine on the network.

```
VNET_MACMAP="AA:DD:11:CE:FF:ED=192.168.1.3 AA:DD:CE:FF:EE=192.168.1.4"
```

On the nodes, you must ensure that the bridge is entered

```
VNET_BRIDGE="br0"
```

Once you have configured Eucalyptus properly, start up the node controllers and the front-end components. To test whether this mode is working properly at run-time, you can follow the last paragraph of the SYSTEM mode, in which the bridge is inspected.

Make sure that the DHCP server has been started properly on the front-end ('ps axww | grep -i dhcpd | grep -i euca'). At this point, the VM should be sending DHCP requests to the local ethernet, and the DHCP server on the front-end should be sending a reply with one of the static MAC/IP mappings the admin has defined in 'eucalyptus.conf'.

CAVEATS - In this mode, as mentioned previously, VMs are started with their ethernet interfaces attached to the local ethernet without any isolation. Practically, this means that you should treat a VM the same way that you would treat a non-VM machine running on the network. Eucalyptus does not verify that your settings are valid, thus, you must enter them correctly in order for your VMs to obtain IP addresses. Finally, we assume that the installed DHCP daemon is, or is compatible with, ISC DHCP Daemon version 3.0.X. If it is not, we recommend either installing a version that is (common in most distributions) or writing a wrapper script around your installed DHCP server and point Eucalyptus at it (via VNET_DHCPDAEMON in 'eucalyptus.conf').

MANAGED Mode

In this mode, Eucalyptus will fully manage the local VM instance network and provides all of the networking features Eucalyptus current supports (VM network isolation, user controllable VM firewalls (ingress rules/security groups), dynamic public IP assignment). The options in 'eucalyptus.conf' that must be configured correctly in 'MANAGED' mode are as follows:

On the front-end (options annotated with a '*' may be required depending on your installation, see below for details):

```
VNET_MODE="MANAGED"
VNET_PUBINTERFACE
```

```
VNET_PRIVINTERFACE
VNET_DHCPDAEMON
*VNET_DHCPUSER
VNET_SUBNET
VNET_NETMASK
VNET_DNS
VNET_ADDRSPERNET
*VNET_PUBLICIPS
*VNET_CLOUDIP
*VNET_LOCALIP
```

On each node:

```
VNET_MODE="MANAGED"
VNET_PUBINTERFACE
VNET_PRIVINTERFACE
```

Be advised that this mode requires that your local network/configuration conforms to certain requirements that Eucalyptus depends upon.

Requirements for MANAGED mode

Before using 'MANAGED' mode, you must confirm that:

> 1.) there is an available range of iP addresses that is completely unused on the network (192.168..., 10....., other).
>
> 2.) your network is 'VLAN clean', meaning that all switch ports that Eucalyptus components are connected to will allow and forward VLAN tagged packets.
>
> 3.) you are not running a firewall on the front-end (CC) or your firewall is compatible with the dynamic changes that Eucalyptus will make to the front-end's netfilter rules.

All three of these requirements must be met before MANAGED mode should be attempted. Failure to verify the above will, at least, result VM instances being unavailable on the network.

For requirement '1', choose a IP range that you know is completely unused on your network. Choose a range that is as large as possible. Typical examples are:

if the network 10.0.0.0 - 10.255.255.255 is completely unused:

```
VNET_MODE="MANAGED"
VNET_SUBNET="10.0.0.0"
VNET_NETMASK="255.0.0.0"
VNET_DNS="<your DNS>"
VNET_ADDRSPERNET="128"
```

or if the network 192.168.0.0 - 192.168.255.255 is completely unused:

```
VNET_MODE="MANAGED"
VNET_SUBNET="192.168.0.0"
VNET_NETMASK="255.255.0.0"
VNET_DNS="<your DNS>"
VNET_ADDRSPERNET="64"
```

Next, the admin must verify that the local network will allow/forward VLAN tagged packets between machines running Eucalyptus components. To verify, perform the following test:

on the front-end, choose the interface that is on the local ethernet (and will be set in eucalyptus.conf as VNET_PRIVINTERFACE), and run:

```
vconfig add <interface> 10
ifconfig <interface>.10 192.168.1.1 up
```

replace '192.168.1.1' with an IP from the range you selected above.

On the node, choose the interface on the local network (will be set in eucalyptus.conf as VNET_PRIVINTERFACE and VNET_PUBINTERFACE), and run:

```
vconfig add <interface> 10
ifconfig <interface>.10 192.168.1.2 up
```

again, replace '192.168.1.2' with another IP in the range you selected above.

Then, try a ping between hosts. On the front-end:

```
ping 192.168.1.2
```

on the node:

```
ping 192.168.1.1
```

If this does not work, then your switch needs to be configured to forward VLAN tagged packets (if it is a managed switch, see your switch's documentation to determine how to do this).

Finally, you need to carefully inspect the firewall on the front-end to make sure that it will not interfere with Eucalyptus, or vice-versa. Eucalyptus will flush the 'filter' and 'nat' tables upon boot in MANAGED mode, but provides a way for the administrator to define special rules that are loaded when Eucalyptus starts (see below for details).

Configuring MANAGED mode

The Eucalyptus administrator must configure the front-end's 'eucalyptus.conf' first with a valid, configured ethernet device that is attached to the same physical ethernet as the Eucalyptus nodes:

```
VNET_PRIVINTERFACE="eth0"
```

If the front-end has a second ethernet device which is used to access the public network, let's say eth1, then you need to configured it as

```
VNET_PUBINTERFACE="eth1"
```

Next, the admin ust ensure that there is a DHCP server binary installed on the front-end and Eucalyptus knows where it is located:

```
VNET_DHCPDAEMON="/usr/sbin/dhcpd3"
```

If your DHCP daemon binary is configured to run as 'non-root' (say, as the user 'dhcpd' as is the case in Ubuntu >= 8.10), then you must configure Eucalyptus to be aware of that user:

```
VNET_DHCPUSER="<dhcpusername>"
```

Nodes must have VNET_PRIVINTERFACE and VNET_PUBINTERFACE set properly (they should be set with the same value, the device facing the CC). For example, with current Xen versions, this parameter (when your node's Xen bridge is 'eth0') is typically:

```
VNET_PUBINTERFACE="peth0"
VNET_PRIVINTERFACE="peth0"
```

while for kvm it should be something like

```
VNET_PUBINTERFACE="eth0"
VNET_PRIVINTERFACE="eth0"
```

Once you have verified that your network configuration meets the requirements for running in MANAGED mode, the rest of the configuration is fairly simple. For example, if the 192.168.0.0/16 network is free and unused on your network:

```
VNET_MODE="MANAGED"
VNET_SUBNET="192.168.0.0"
VNET_NETMASK="255.255.0.0"
VNET_DNS="<your dns>"
VNET_ADDRSPERNET="64"
VNET_PUBLICIPS="<publicIPa> <publicIPb> ... <publicIPn>"
```

SUBNET, NETMASK, and DNS have been described previously. VNET_ADDRSPERNET is used to control how many VM instances may simultaneously be part of an individual user's named network (called a 'security group' in Amazon EC2). Choosing the right value for this parameter depends on how many IPs you have made available using VNET_SUBNET/VNET_NETMASK, how many VLANs your network supports simultaneously, and how many concurrent active user networks the administrator wishes to support. In the above example, there are 65536 addresses available (192.168.0.0/16). If we divide by the number of addresses per network (set to 64 above), we find the maximum number of simultaneous active named networks that can be in use at any one point in time (65536 / 64 == 1024). If your eucalyptus installation has 100 users, then each user could have at most 10 active security groups in operation at any point in time (of course, they can define as many as they wish, but can only have sets of running VMs residing in at most 10 networks). Each security group could support up to 61 instances (64 addresses minus 1 address for the subnet, broadcast, and router IPs). If your installation favors more VMs per network and fewer active security groups per user, the administrator may adjust the VNET_ADDRSPERNET parameter accordingly. Setting it to '256' would result in each active user's security group supporting up to 253 VM instances, and each of 100 users could simultaneously have 2 active security groups.

If you would like users to log in to their instances from outside the cluster/cluster front-end, you must find a set of public IP addresses, that are not in use, and allow Eucalyptus to dynamically route them to VM instances at instance boot time or dynamically at run time. For each IP address you choose, your front-end must be capable of being configured with that IP address. To test, choose some free public IP addresses and perform the following test for each one:

on the front-end:

```
ip addr add <publicIP>/32 dev <interface>
```

on some external machine representative of where users will wish to log into their VM instances:

```
ping <publicIP>
```

if this works, then dynamic IP assignment to VM instances will work. Remove the assigned address with the following command:

```
ip addr del <publicIP>/32 dev <interface>
```

Once you have compiled a list of available public IP addresses, allow Eucalyptus to use them by listing the IPs in 'eucalyptus.conf':

```
VNET_PUBLICIPS="<publicIPa> <publicIPb> ... <publicIPn>"
```

or, you can specify a range of IPs with:

```
VNET_PUBLICIPS="<publicIPa>-<publicIPb>"
```

where publicIPa and publicIPb are in the same /24 subnet.

If your cluster-controller and cloud-controller are running on separate hosts, you need to set:

```
VNET_CLOUDIP="<ip-of-cloud-controller>"
```

And, if you are running multiple clusters in your installation, and wish to manually specify the IP of the cluster-controller that all other cluster-controllers can reach, you may set:

```
VNET_LOCALIP="<ip-of-cluster-controller"
```

The cluster-controller will attempt to determine this value automatically if it is not set.

CAVEATS - When Eucalyptus is running in MANAGED mode, you cannot currently run an entire eucalyptus installation on a single machine as this mode depends upon traffic between named networks passing through a front-end router (instead of going through the loopback device). If you wish to run Eucalyptus on a single machine (laptop), you must use SYSTEM or STATIC mode. In MANAGED mode, Eucalyptus will flush the front-end's iptables rules for both 'filter' and 'nat'. Next, it will set the default policy for the 'FORWARD' chain in 'filter' to 'DROP'. At run time, the front-end will be adding and removing rules from 'FORWARD' as users add/remove ingress rules from their active security groups. In addition, the 'nat' table will be configured to allow VMs access to the external network using IP masquerading, and will dynamically add/remove rules in the 'nat' table as users assign/unassign public IPs to VMs at instance boot or run-time. If the administrator has some rules that they wish to apply o the front-end, they should perform the following procedure on the front-end, before eucalyptus is started or while eucalyptus is not running. **WARNING** if the admin chooses to perform this operation to define special iptables rules that are loaded when Eucalyptus starts, they could inadvertently cause Eucalyptus VM networking to fail. It is suggested that you only do this only if you are completely sure that it will not interfere with the operation of Eucalyptus.

```
<use iptables to set up your iptables rules>
```

```
iptables-save > $EUCALYPTUS/var/run/eucalyptus/net/iptables-preload
```

Troubleshooting MANAGED Mode

If you start an instance believe that it is running but is not available on the network, here are some things to check.

First, verify that the requirements of MANAGED mode have been met as described above (unused range of IPs, VLAN capable network, no interfering firewall rules on the nodes or front-end). Test whether you can get to the instance from the front-end using it's private address (from the range you specified). If you cannot, next, inspect the interfaces on the front-end and nodes:

on front-end:

```
ifconfig -a
```

You should see an interface '<interface>.<vlan>' with an IP address that is up and running. For instance, if may be 'eth0.10'. If it is not, check your VNET_PUBINTERFACE and VNET_PRIVINTERFACE parameter and inspect the eucalyptus log files for errors.

on the node:

```
brctl show
```

You should see a number of bridges called 'eucabr<vlan>', where '<vlan>' is a number that typically starts from '10'. The output should be similar (if VNET_PRIVINTERFACE="peth0") to:

```
; brctl show
bridge name  bridge id          STP enabled     interfaces
eucabr10     8000.000c29369858  no              peth0.10
                                                vif18.0
```

If this is not the case, check your VNET_PRIVINTERFACE setting, and inspect the logfiles for details.

Back on the front-end, make sure that 'dhcpd' is running:

```
ps axww | grep <dhcpd>
```

where '<dhcpd>' is what you have set for VNET_DHCPDAEMON. Make sure that, in the output of 'ps', you see that the daemon is listening on the vlan tagged interface from above (<interface>.<vlan>). If it is not running, check the eucalyptus logs for the reason why (if the command failed, you will see this information in 'cc.log', if the daemon failed at runtime, you can inspect the reason in the daemon's output itself in 'http-cc_error_log'.

If you can access the private IP of the instance from the front-end, but public IPs are not being forwarded properly, first confirm that the user's security group is set up properly by having them run 'euca-describe-group <group of instance>'. '<group of instance>' is set to 'default' by default or if unspecified when the instance was started. If the group has appropriate ingress rules set, check that the rules have been implemented on the front-end:

```
iptables -L <username>-<groupname>
```

If there are no rules here, check the 'cc.log' for errors applying the table rules for more insight. Next, check the 'nat' table:

```
iptables -L -t nat
```

You should see one DNAT rule for routing traffic from a public IP to the instance IP, and one SNAT rule for setting the source IP of outgoing packets from that instance. If you do not, check 'cc.log' to determine the cause.

If all of these checks pass and the instance still is experiencing network problems, please prepare the following information and send it along to the Eucalyptus discussion board:

on front-end and one representative node, capture the output of the following commands:

```
netstat -rn
ifconfig -a
brctl show
iptables-save
```

and send us 'cc.log', 'nc.log', 'httpd-cc_error_log' and 'httpd-nc_error_log'.

MANAGED-NOVLAN Mode

In this mode, Eucalyptus will fully manage the local VM instance network and provides all of the networking features Eucalyptus current supports (user controllable VM firewalls (ingress rules/security groups), dynamic public IP assignment), but does not provide VM network isolation. The options in 'eucalyptus.conf' that must be configured correctly in 'MANAGED-NOVLAN' mode are as follows:

On the front-end (options annotated with a '*' may be required depending on your installation, see below for details):

```
VNET_MODE="MANAGED-NOVLAN"
VNET_PRIVINTERFACE
VNET_PUBINTERFACE
VNET_DHCPDAEMON
*VNET_DHCPUSER
VNET_SUBNET
VNET_NETMASK
VNET_DNS
VNET_ADDRSPERNET
*VNET_PUBLICIPS
*VNET_CLOUDIP
*VNET_LOCALIP
```

On each node:

```
VNET_MODE="MANAGED-NOVLAN"
VNET_BRIDGE
```

Be advised that this mode requires that your local network/configuration conforms to certain requirements that Eucalyptus depends upon.

Requirements for MANAGED-NOVLAN mode

Before using 'MANAGED-NOVLAN' mode, you must confirm that:

1.) there is an available range of iP addresses that is completely unused on the network (192.168..., 10....., other).

2.) you are not running a firewall on the front-end (CC) or your firewall is compatible with the dynamic changes that Eucalyptus will make to the front-end's netfilter rules.

Both of these requirements must be met before MANAGED-NOVLAN mode should be attempted. Failure to verify the above will, at least, result VM instances being unavailable on the network.

For requirement '1', choose a IP range that you know is completely unused on your network. Choose a range that is as large as possible. Typical examples are:

if the network 10.0.0.0 - 10.255.255.255 is completely unused:

```
VNET_MODE="MANAGED-NOVLAN"
VNET_SUBNET="10.0.0.0"
VNET_NETMASK="255.0.0.0"
VNET_DNS="<your DNS>"
VNET_ADDRSPERNET="128"
```

or if the network 192.168.0.0 - 192.168.255.255 is completely unused:

```
VNET_MODE="MANAGED-NOVLAN"
VNET_SUBNET="192.168.0.0"
VNET_NETMASK="255.255.0.0"
VNET_DNS="<your DNS>"
VNET_ADDRSPERNET="64"
```

You will need to carefully inspect the firewall on the front-end to make sure that it will not interfere with Eucalyptus, or vice-versa. Eucalyptus will flush the 'filter' and 'nat' tables upon boot in MANAGED-NOVLAN mode, but provides a way for the administrator to define special rules that are loaded when Eucalyptus starts (see below for details).

Configuring MANAGED-NOVLAN mode

The Eucalyptus administrator must configure the front-end's 'eucalyptus.conf' first with a valid, configured ethernet device that is attached to the same physical ethernet as the Eucalyptus nodes:

```
VNET_PRIVINTERFACE="eth0"
```

If the front-end has a second ethernet device which is used to access the public network, let's say eth1, then you need to configured it as

```
VNET_PUBINTERFACE="eth1"
```

Next, the admin ust ensure that there is a DHCP server binary installed on the front-end and Eucalyptus knows where it is located:

```
VNET_DHCPDAEMON="/usr/sbin/dhcpd3"
```

If your DHCP daemon binary is configured to run as 'non-root' (say, as the user 'dhcpd' as is the case in Ubuntu >= 8.10), then you must configure Eucalyptus to be aware of that user:

```
VNET_DHCPUSER="<dhcpusername>"
```

Nodes must have VNET_BRIDGE set properly:

```
VNET_BRIDGE="br0"
```

Once you have verified that your network configuration meets the requirements for running in MANAGED-NOVLAN mode, the rest of the configuration is fairly simple. For example, if the 192.168.0.0/16 network is free and unused on your network:

```
VNET_MODE="MANAGED-NOVLAN"
VNET_SUBNET="192.168.0.0"
VNET_NETMASK="255.255.0.0"
VNET_DNS="<your dns>"
VNET_ADDRSPERNET="64"
VNET_PUBLICIPS="<publicIPa> <publicIPb> ... <publicIPn>"
```

SUBNET, NETMASK, and DNS have been described previously. VNET_ADDRSPERNET is used to control how many VM instances may simultaneously be part of an individual user's named network (called a 'security group' in Amazon EC2). Choosing the right value for this parameter depends on how many IPs you have made available using VNET_SUBNET/VNET_NETMASK, how many VLANs your network supports simultaneously, and how many concurrent active user networks the administrator wishes to support. In the above example, there are 65536 addresses available (192.168.0.0/16). If we divide by the number of addresses per network (set to 64 above), we find the maximum number of simultaneous active named networks that can be in use at any one point in time (65536 / 64 == 1024). If your eucalyptus installation has 100 users, then each user could have at most 10 active security groups in operation at any point in time (of course, they can define as many as they wish, but can only have sets of running VMs residing in at most 10 networks). Each security group could support up to 61 instances (64 addresses minus 1 address for the subnet, broadcast, and router IPs). If your installation favors more VMs per network and fewer active security groups per user, the administrator may adjust the VNET_ADDRSPERNET parameter accordingly. Setting it to '256' would result in each active user's security group supporting up to 253 VM instances, and each of 100 users could simultaneously have 2 active security groups.

If you would like users to log in to their instances from outside the cluster/cluster front-end, you must find a set of public IP addresses, that are not in use, and allow Eucalyptus to dynamically route them to VM instances at instance boot time or dynamically at run time. For each IP address you choose, your front-end must be capable of being configured with that IP address. To test, choose some free public IP addresses and perform the following test for each one:

on the front-end:

```
ip addr add <publicIP>/32 dev <interface>
```

on some external machine representative of where users will wish to log into their VM instances:

```
ping <publicIP>
```

if this works, then dynamic IP assignment to VM instances will work. Remove the assigned address with the following command:

```
ip addr del <publicIP>/32 dev <interface>
```

Once you have compiled a list of available public IP addresses, allow Eucalyptus to use them by listing the IPs in 'eucalyptus.conf':

```
VNET_PUBLICIPS="<publicIPa> <publicIPb> ... <publicIPn>"
```

or, you can specify a range of IPs with:

```
VNET_PUBLICIPS="<publicIPa>-<publicIPb>"
```

where publicIPa and publicIPb are in the same /24 subnet.

If your cluster-controller and cloud-controller are running on separate hosts, you need to set:

```
VNET_CLOUDIP="<ip-of-cloud-controller>"
```

And, if you are running multiple clusters in your installation, and wish to manually specify the IP of the cluster-controller that all other cluster-controllers can reach, you may set:

```
VNET_LOCALIP="<ip-of-cluster-controller"
```

The cluster-controller will attempt to determine this value automatically if it is not set.

CAVEATS - When Eucalyptus is running in MANAGED-NOVLAN mode, you cannot currently run an entire eucalyptus installation on a single machine as this mode depends upon traffic between named networks passing through a front-end router (instead of going through the loopback device). If you wish to run Eucalyptus on a single machine (laptop), you must use SYSTEM or STATIC mode. In MANAGED-NOVLAN mode, Eucalyptus will flush the front-end's iptables rules for both 'filter' and 'nat'. Next, it will set the default policy for the 'FORWARD' chain in 'filter' to 'DROP'.

At run time, the front-end will be adding and removing rules from 'FORWARD' as users add/remove ingress rules from their active security groups. In addition, the 'nat' table will be configured to allow VMs access to the external network using IP masquerading, and will dynamically add/remove rules in the 'nat' table as users assign/unassign public IPs to VMs at instance boot or run-time. If the administrator has some rules that they wish to apply o the front-end, they should perform the following procedure on the front-end, before eucalyptus is started or while eucalyptus is not running. **WARNING** if the admin chooses to perform this operation to define special iptables rules that are loaded when Eucalyptus starts, they could inadvertently cause Eucalyptus VM networking to fail. It is suggested that you only do this only if you are completely sure that it will not interfere with the operation of Eucalyptus.

```
<use iptables to set up your iptables rules>

iptables-save > $EUCALYPTUS/var/run/eucalyptus/net/iptables-preload
```

If you edit a networking related value in eucalyptus.conf, you will need to restart the CC ($EUCALYPTUS/etc/init.d/eucalyptus-cc restart) for changes to take effect. If you change the networking mode, you will need to perform a cleanrestart ($EUCALYPTUS/etc/init.d/eucalyptus-cc cleanrestart)

If you are running Eucalyptus in multi-cluster mode, we strongly recommend that you configure all your clusters to have an identical networking mode. In addition we **strongly recommend** that you do not run multiple clusters in the same broadcast domain. Each cluster should be in a separate domain.

Multi-cluster networking

Eucalyptus versions >= 1.6 support multiple clusters within a single Eucalyptus cloud installation. This section briefly describes how Eucalyptus manages the networking aspect of a multi-cluster setup. First, in SYSTEM or STATIC networking modes, Eucalyptus does not perform any special configuration for a multi-cluster setup. In MANAGED and MANAGED-NOVLAN modes, Eucalyptus will set up layer two tunnels between your clusters, so that virtual machines that are in the same security group, but distributed across clusters (potentially each in their own broadcast domain), can communicate with one another. We use the 'vtun' package to handle all layer two tunneling between clusters.

For the most part, as long as 'vtun' is installed on your cluster controllers, multi-cluster tunneling is handled automatically by the cluster controller software. There are a few caveats to be aware of, depending on your chosen networking mode and network topology.

MANAGED mode: during normal operation, you will see many tunnel interfaces being created and destroyed as virtual networks are constructed and torn down.

MANAGED-NOVLAN mode: your CC will need to be configured with a bridge as it's primary, public interface (VNET_PUBINTERFACE) in order for vtun tunneling to work in this mode.

BOTH modes: the CC attempts to auto-discover it's list of local IP addresses upon startup, but if the IP that was used to register the CC is not locally available, you can override the CC's notion of 'self' by setting the 'VNET_LOCALIP' variable in eucalyptus.conf.

BOTH modes: do not run two CCs in the same broadcast domain with tunneling enabled, this will potentially lead to a broadcast storm as tunnels start forwarding packets in a loop on your local network.

If you wish to disable tunneling altogether, set 'VNET_LOCALIP=0.0.0.0' in eucalyptus.conf.

Management

Managing Eucalyptus Images (1.6)

First, be sure to source your 'eucarc' file before running the commands below. Note that all users may upload and register images (depending on access granted to them by the Eucalyptus administrator), but only the admin user may ever upload/register kernels or ramdisks.

Second, the instructions below rely on the euca2ools command-line tools distributed by the Eucalyptus Team. Please, install them if you haven't done so already.

1. Adding Images

To enable a VM image as an executable entity, a user/admin must add a root disk image, a kernel/ramdisk pair (ramdisk may be optional) to Walrus and register the uploaded data with Eucalyptus. Each is added to Walrus and registered with Eucalyptus separately, using three EC2 commands. The following example uses the test image that we provide. Unpack it to any directory:

Add the kernel to Walrus, and register it with Eucalyptus (**WARNING**: your bucket names must not end with a slash!):

```
euca-bundle-image -i <kernel file> --kernel true
euca-upload-bundle -b <kernel bucket> -m /tmp/<kernel file>.manifest.xml
euca-register <kernel-bucket>/<kernel file>.manifest.xml
```

Next, add the root filesystem image to Walrus:

```
euca-bundle-image -i <vm image file>
euca-upload-bundle -b <image bucket> -m /tmp/<vm image file>.manifest.xml
euca-register <image bucket>/<vm image file>.manifest.xml
```

Our test kernel does not require a ramdisk to boot. If the administrator would like to upload/register a kernel/ramdisk pair, the procedure is similar to the above:

```
euca-bundle-image -i <initrd file> --ramdisk true
euca-upload-bundle -b <initrd bucket> -m /tmp/<initrd file>.manifest.xml
euca-register <initrd bucket>/<initrd file>.manifest.xml
```

2. Associating kernels and ramdisks with instances

There are three ways that one can associate a kernel (and ramdisk) with a VM instance.

1. A user may associate a specific kernel/ramdisk identifier with an image at the 'euca-bundle-image' step

   ```
   euca-bundle-image -i <vm image file> --kernel <eki-XXXXXXXX> --ramdisk <eri-XXXXXXXX>
   ```

2. A user may choose a specific kernel/ramdisk at instance run time as an option to 'euca-run-instances'

   ```
   euca-run-instances --kernel <eki-XXXXXXXX> --ramdisk <eri-XXXXXXXX> <emi-XXXXXXXX>
   ```

3. The administrator can set 'default' registered kernel/ramdisk identifiers that will be used if a kernel/ramdisk is unspecified by either of the above options. This is accomplished by logging in to the administrative interface (https://your.cloud.server:8443), clicking on the 'Configuration' tab and adding an <eki-xxxxxxxx> and optionally an <eri-

xxxxxxxx> as the defaults kernel/ramdisk to be used.

3. Deleting Images

In order to delete an image, you must first de-register the image:

```
euca-deregister <emi-XXXXXXXX>
```

Then, you can remove the files stored in your bucket. Assuming you have sourced your 'eucarc' to set up EC2 client tools:

```
euca-delete-bundle -a $EC2_ACCESS_KEY -s $EC2_SECRET_KEY --url $S3_URL -b <bucket> -p <file prefix>
```

If you would like to remove the image and the bucket, add the '--clear' option:

```
euca-delete-bundle -a $EC2_ACCESS_KEY -s $EC2_SECRET_KEY --url $S3_URL -b <bucket> -p <file prefix> --clear
```

Examples

Following is an example using the Ubuntu pre-packaged image that we provide using the included KVM compatible kernel/ramdisk (a Xen compatible kernel/ramdisk is also included). See this page to get more pre-packaged images.

```
tar zxvf euca-ubuntu-9.04-x86_64.tar.gz

euca-bundle-image -i euca-ubuntu-9.04-x86_64/kvm-kernel/vmlinuz-2.6.28-11-generic --kernel true
euca-upload-bundle -b ubuntu-kernel-bucket -m /tmp/vmlinuz-2.6.28-11-generic.manifest.xml
euca-register ubuntu-kernel-bucket/vmlinuz-2.6.28-11-generic.manifest.xml
(set the printed eki to $EKI)

euca-bundle-image -i euca-ubuntu-9.04-x86_64/kvm-kernel/initrd.img-2.6.28-11-generic --ramdisk true
euca-upload-bundle -b ubuntu-ramdisk-bucket -m /tmp/initrd.img-2.6.28-11-generic.manifest.xml
euca-register ubuntu-ramdisk-bucket/initrd.img-2.6.28-11-generic.manifest.xml
(set the printed eri to $ERI)

euca-bundle-image -i euca-ubuntu-9.04-x86_64/ubuntu.9-04.x86-64.img --kernel $EKI --ramdisk $ERI
euca-upload-bundle -b ubuntu-image-bucket -m /tmp/ubuntu.9-04.x86-64.img.manifest.xml
euca-register ubuntu-image-bucket/ubuntu.9-04.x86-64.img.manifest.xml
```

Now, the newly uploaded image(s) should be ready to start using (see User's Guide for more information on using Eucalyptus).

Eucalyptus Management (1.6)

This part of the Administrator's Guide describes tasks that can be performed on a completed Eucalyptus installation, whether it was installed from source or from packages.

1. Image Management

To use Eucalyptus, images must be added and registered with the system. We have a document detailing the steps of this process in Image Management.

2. Node Management

Once you have a running Eucalyptus system you can add and remove nodes (systems running Node Controllers) using

```
$EUCALYPTUS/usr/sbin/euca_conf --register-nodes "<nodename1> ... <nodenameN>"
```

you will be asked for password to login to <nodenameX>: this is needed to propagate the cryptographic keys. Similarly to remove a node

```
$EUCALYPTUS/usr/sbin/euca_conf --deregister-nodes "<nodename1> ... <nodenameN>"
```

3. User Management

3.1 User sign-up

Users interested in joining the cloud should be directed to the front-end Web page (note the **https** prefix!):

> https://your.front.end.hostname:8443/

As soon as the administrator logs in for the first time and enters the email address to be used for application requests, thus activating the Web site for use by others, the login box of the Web site will have an "Apply for account" link underneath it. After a user fills out the application form, an email is sent to the administrator, containing two URLs, one for accepting and one for rejecting the user.

Note that there is no authentication performed on the people who fill out the form. It is up to the administrator to perform this authentication! The only "guarantee" the administrator has is that the account will not be active unless the person who requested the account (and, hence, knows the password) can read email at the submitted address. Therefore, if the administrator is willing to give the account to the person behind the email address, it is safe to approve the account. Otherwise, the administrator may use the additional information submitted (such as the telephone number, project PI, etc.) to make the decision.

Accepting or rejecting a signup request causes an email message to be sent to the user who made the request. In the case of an acceptance notification, the user will see a link for activating the account. Before activating the account, the user will have to log in with the username and password that they chose at signup.

3.2 Adding users

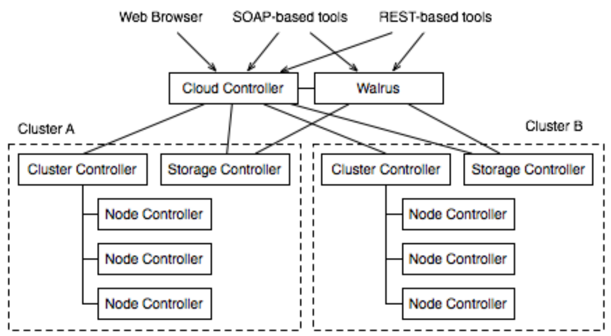
Users can be added by the administrator explicitly by logging into the Eucalyptus web interface, as an administrative user, clicking the 'Users' tab, clicking on the 'Add User' button, and filling out the same user form that a user would fill out if they applied themselves. The user will be automatically 'approved' using this method, but their account will not be active until the user clicks the link that is sent via email similar to the above method.

3.3 Managing users

If the administrator wishes to disable or delete a user, they can do so through the web interface, as an administrative user, clicking the 'Users' tab, and clicking either the 'disable' or 'delete' link respectively.

Guides

Advanced Eucalyptus Setup



A Eucalyptus setup consists of a number of web services components -- The Cloud Controller (**CLC**), Walrus, Storage Controller (**SC**), Cluster Controller (**CC**) and the Node Controller (**NC**). In a single cluster setup, all front end web services (all services except the NCs) run on a single physical host. In a more advanced configuration, you can choose to run the CLC, Walrus, SC and CC on a separate physical machines, or you can combine them as you see fit. For example, one reason to separate Eucalyptus components is to improve the overall performance of the system by distributing different types of work (i.e. place Walrus on a machine that has a fast disk subsystem, while the CLC can be placed on another machine with a fast CPU).

In addition, in a multi-cluster setup, you may configure more than one cluster with Eucalyptus, either for performance or management purposes. This guide briefly describe multi-component and multi-cluster setups. In a multi-cluster installation, there must be a single Cloud Controller, a single Walrus, one Cluster Controller and Storage Controller pair per cluster, and one or more Node Controllers grouped within each cluster, as illustrated in the adjacent figure.

We assume that you familiar and comfortable with single cluster Eucalyptus installation, configuration, management and usage before proceeding to more advanced configurations. We also assume that you are able to install individual Eucalyptus components on the distribution and architecture of your choice. Please refer to the previous installation sections of the Administrator's Guide for information regarding distribution specific installation methods, package names, dependencies, etc.

Multi-component and Multi-cluster Setup

A Eucalyptus installation consists of five components: A Cloud Controller (CLC), Walrus, one or more Cluster Controller (CC) and Storage Controller (SC) pairs, and one or more Node Controllers (NC). Once you have the components installed on your physical hosts, using the topology of your choice, you're ready to enable the services and stitch together your Eucalyptus cloud by 'registering' the components with one another.

Enabling services

If you installed Eucalyptus from source and rsynced the build to your physical hosts, you will need to explicitly enable or disable the services that you wish to run (all are disabled by default). This can be done by running

```
$EUCALYPTUS/usr/sbin/euca_conf {--enable,--disable} cloud

$EUCALYPTUS/usr/sbin/euca_conf {--enable,--disable} walrus

$EUCALYPTUS/usr/sbin/euca_conf {--enable,--disable} sc
```

to enable and disable the CLC, Walrus and SC respectively. CLC, Walrus and SC execution and termination is controlled using the same script, "$EUCALYPTUS/etc/init.d/eucalyptus-cloud {start,stop,restart}"

The CC is controlled using "$EUCALYPTUS/etc/init.d/eucalyptus-cc {start,stop,restart,cleanstart,cleanstop,cleanrestart}"

If you have installed from packages, service enabling is done during package install, and service startup may have been done depending on your packaging method. If you install multiple services on the same host, but do not wish to enable them all, you can "--disable" the service, followed by a 'restart' of 'eucalyptus-cloud'.

After the services are up and running, you'll need to stitch together your Eucalyptus cloud by registering them, so that they can start to communicate with one another.

Registering Services

First, you will need to inform the Cloud Controller (CLC) the location of Walrus. To do this, run the following command on the **CLC host**.

```
$EUCALYPTUS/usr/sbin/euca_conf --register-walrus <Walrus IP address>
```

where "Walrus IP address" is the IP address of the host on which you have installed Walrus. In case your host has multiple IP addresses, pick the one that is visible from the CLC and clients, and note that 'localhost' or '127.*.*.*' are not valid Walrus endpoint addresses, even if your Walrus service is running on the same host as your CLC.

Next, you will need to register a Cluster Controller (CC) and a Storage Controller (SC) with the CLC. There is one CC/SC pair per cluster.

To do so, run the following commands on the **CLC host**,

```
$EUCALYPTUS/usr/sbin/euca_conf --register-cluster <clustername> <CC IP address>
$EUCALYPTUS/usr/sbin/euca_conf --register-sc <clustername> <SC IP address>
```

where "CC IP address" and "SC IP address" are IP addresses of the CC host and the SC host respectively and "clustername" is the name you want to use for your cluster. Again, note that 'localhost' and '127.*.*.*' addresses are invalid.

If you are installing the Cloud Controller (CLC), Cluster Controller (CC), Walrus and the Storage Controller (SC) on same machine, the IP address above will be the same in all steps.

If you setting up multiple clusters, you will need to install a CC and SC per cluster and you will need to run the above "--register-cluster" and "--register-sc" commands for each cluster.

If you are setting up multiple clusters, the networking options on each CC must be identical with the exception of VNET_PUBLICIPS, which may be set to CC specific values. In addition, if you are running in MANAGED-NOVLAN networking mode, then each CC's VNET_PRIVINTERFACE must be set to a valid bridge that is configured and running properly before the Eucalyptus CCs are started.

Finally, you need to register nodes with the CC. To do so, run the following command on the **CC host**

```
$EUCALYPTUS/usr/sbin/euca_conf --register-nodes "<Node0 IP address> <Node1 IP address> ... <NodeN IP address>"
```

where "<NodeX IP address>" is the IP address of a host that is running a Node Controller (NC).

In a multi-cluster setup, you will need to run this command on each CC in your setup.

At this point, you have successfully registered Eucalyptus components and you are ready to proceed to configuration.

Monitoring

Eucalyptus 1.6 provides some facilities to help in monitoring the status of running components, of running VMs and statistics on storage uses.

We provide two example script to integrate with nagios and ganglia. Nagios will be able to monitor the status of the machine running Eucalyptus components as well as the components itself, while ganglia can provide more information on the resource usage.

The examples we provides, requires a running installation of nagios and ganglia. For example on debian/ubuntu, one can start followign nagios installation until the monitoring page is accessible from the web browser. After that, one can run the example script (followig the instruction in README.Monitoring) to add the rules to monitor Eucalyptus, and restarting nagios will provide basic monitoring facilities.

Similarly for ganglia, one has to have a running ganglia installation, after which the appropriate script (provided in the directory extras) will injects some extra informations which will eventually be visible. It can take sometime to have the information to show up on ganglia's graphs.

The scripts can be used as a template for other monitoring infrastructures.

Setting up Dynamic DNS

The Eucalyptus cloud front end has a Domain Name System (DNS) service built into it that will respond to DNS requests in order to support virtual hosting of buckets (mapping bucket names to IP addresses) and instance DNS (mapping of hostnames to instance public and private IP addresses).

Enabling DNS in Eucalyptus

Since most Linux distributions have "dnsmasq" or "bind" or another DNS server running on the same physical host as the CLC, DNS in Eucalyptus is disabled by default. The Eucalyptus administrator can enable DNS after disabling or re-configuring any other services that listen on port 53.

To do so, edit $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf on the CLC machine and change the value of "DISABLE_DNS" to,

```
DISABLE_DNS=N
```

then, restart "eucalyptus-cloud".

Note: To verify that the DNS service is running, try running "netstat -al" and look for 53. Port 53 should be bound by the process "eucalyptus-cloud"

Configuring DNS in Eucalyptus

First, login to the front end admin web interface (https://<front end IP>:8443) and click on the "Configuration" tab.

Next, pick a subdomain within your domain that the Eucalyptus DNS system will service. Set the nameserver IP to the IP of the cloud front end that is accessible by your master DNS system. Then, set the domain that the Eucalyptus front end will service to <subdomain>.<domain>

Finally, you will need to change your master DNS configuration to point to the CLC public IP (entered in the above step) as the nameserver for your chosen subdomain within your organization. Optionally, you can change your client's config (/etc/resolv.conf on Linux) to point to the cloud front end IP directly, but note that you will have to do this on each system that needs to access the Eucalyptus DNS service.

Using DNS

After DNS is setup correctly, creation and deletion of buckets will automatically create DNS entries. The DNS service embedded in the CLC will respond with the Walrus IP when a DNS request is made for <bucketname>.walrus.<subdomain>.

In addition, instances IPs will be mapped as euca-A.B.C.D.eucalyptus.<subdomain>, where A.B.C.D is the IP address (or addresses) assigned to your instance.

Eucalyptus Troubleshooting (1.6)

Eucalyptus cloud admins are encouraged to consult the Known Bugs page before diving into the investigation of unexpected behavior.

The instructions below rely on the euca2ools command-line tools distributed by the Eucalyptus Team. Please, install them if you haven't done so already.

1. Restarting

Eucalyptus components can be restarted using the init scripts at any time with the 'restart' operation:

```
/etc/init.d/eucalyptus-cloud restart
/etc/init.d/eucalyptus-cc restart
/etc/init.d/eucalyptus-nc restart
```

If you need to make a change to the cluster controller or node controller configuration through modification of $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf, you will typically be required to 'stop' and then 'start' the service after the modification has been made:

```
/etc/init.d/eucalyptus-cc stop
/etc/init.d/eucalyptus-cc start
/etc/init.d/eucalyptus-nc stop
/etc/init.d/eucalyptus-nc start
```

Warning: depending on your configuration of Eucalyptus, making changes to eucalyptus.conf that drastically alter the way Eucalyptus is handling non-eucalyptus resources (network, hypervisor, etc) may require that all currently running VMs be terminated before the configuration changes can be successfully applied. In addition, if you are running in any network mode (VNET_MODE) other than SYSTEM, correct VM network connectivity is only ensured while the CC that launched the VMs is running. If the machine that hosts a CC that has previously launched VMs fails or reboots, then the VMs will lose network connectivity.

If the administrator needs to terminate running VMs for the reasons described above, they can use the client tools to terminate all instances. Optionally, the admin can manually stop all eucalyptus components, destroy all running Xen instances using 'xm shutdown' or 'xm destroy' on the nodes, and start all Eucalyptus components to return to a clean state.

2. Diagnostics

Installation/Discovering resources

If something is not working right with your Eucalyptus installation, the best first step (after making sure that you have followed the installation/configuration/networking documents faithfully) is to make sure that your cloud is up and running, that all of the components are communicating properly, and that there are resources available to run instances. After you have set up and configured Eucalyptus, set up your environment properly with your admin credentials, and use the following command to see the 'status' of your cloud:

```
euca-describe-availability-zones verbose
```

You should see output similar to the following:

```
AVAILABILITYZONE        cluster <hostname of your front-end>
AVAILABILITYZONE        |- vm types     free / max   cpu   ram  disk
AVAILABILITYZONE        |- m1.small     0128 / 0128   1    128    10
AVAILABILITYZONE        |- c1.medium    0128 / 0128   1    256    10
AVAILABILITYZONE        |- m1.large     0064 / 0064   2    512    10
AVAILABILITYZONE        |- m1.xlarge    0064 / 0064   2   1024    20
AVAILABILITYZONE        |- c1.xlarge    0032 / 0032   4   2048    20
AVAILABILITYZONE        |- <node-hostname-a>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
AVAILABILITYZONE        |- <node-hostname-b>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
AVAILABILITYZONE        |- <node-hostname-c>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
AVAILABILITYZONE        |- <node-hostname-d>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
AVAILABILITYZONE        |- <node-hostname-e>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
AVAILABILITYZONE        |- <node-hostname-f>        certs[cc=true,nc=true] @ Sun Jan 04 15:13:30 PST 2009
...
```

Next, the administrator should consult the Eucalyptus logfiles. On each machine running a Eucalyptus component, the logfiles are located in:

```
$EUCALYPTUS/var/log/eucalyptus/
```

On the front-end, the Cloud Controller (CLC) logs primarily to 'cloud-output.log' and 'cloud-debug.log'. Consult these files if your client tool (ec2 API tools) output contains exception messages, or if you suspect that none of your operations are ever being executed (never see Xen activity on the nodes, network configuration activity on the front-end, etc.).

The Cluster Controller (CC) also resides on the front-end, and logs to 'cc.log' and 'httpd-cc_error_log'. Consult these logfile in general, but especially if you suspect there is a problem with networking. 'cc.log' will contain log entries from the CC itself, and 'httpd-cc_error_log' will contain the STDERR/STDOUT from any external commands that the CC executes at runtime.

A Node Controller (NC) will run on every machine in the system that you have configured to run VM instances. The NC logs to 'nc.log' and 'httpd-nc_error_log'. Consult these files in general, but especially if you believe that there is a problem with VM instances actually running (i.e., it appears as if instances are trying to run - get submitted, go into 'pending' state, then go into 'terminated' directly - but fail to stay running).

Node Controller troubleshooting

- If nc.log reports "Failed to connect to hypervisor," xen/kvm + libvirt is not functioning correctly.
- If the NC cannot be contacted, make sure that you have synchronized keys to the nodes and that the keys are owned by the user that you are running the NC as (EUCA_USER in eucalyptus.conf).

Walrus troubleshooting

- "ec2-upload-bundle" will report a "409" error when uploading to a bucket that already exists. This is a known compatibility issue when using ec2 tools with Eucalyptus. The workaround is to use ec2-delete-bundle with the "--clear" option to delete the bundle and the bucket, before uploading to a bucket with the same name, or to use a different bucket name.

Note: If you are using Euca2ools, this is not necessary.

- When using "ec2-upload-bundle," make sure that there is no "/" at the end of the bucket name.

Block storage troubleshooting

- Unable to attach volumes when the front end and the NC are running on the same machine. This is a known issue with ATA over Ethernet (AoE). AoE will not export to the same machine that the server is running on. The workaround is to run the front end and the node controller on different hosts.
- Volume ends up in "deleted" state when created, instead of showing up as "available." Look for error messages in $EUCALYPTUS/var/log/eucalyptus/cloud-error.log. A common problem is that ATA-over-Ethernet may not be able to export the created volume (this will appear as a "Could not export..." message in cloud-error.log). Make sure that "VNET_INTERFACE" in eucalyptus.conf on the front end is correct.
- Failure to create volume/snapshot. Make sure you have enough loopback devices. If you are installing from packages, you will get a warning. On most distributions, the loopback driver is installed as a module. The following will increase the number of loopback devices available,

  ```
  rmmod loop ; modprobe loop max_loop=256
  ```
- If block devices do not automatically appear in your VMs, make sure that you have the "udev" package installed.
- If you are running gentoo and you get "which: no vblade in ((null)).", try compiling "su" without pam.