Eucalyptus (1.4)

Eucalyptus version 1.4 incorporates several new features and improvements. Most notably, it introduces Walrus, our S3-compatible storage service, as well as a completely revamped virtual network implementation that has proved to be more stable, more efficient, and more flexible than our previous solution. Here is the summary of changes from version 1.3:

- Walrus: an Amazon S3 interface-compatible storage manager. Walrus handles storage of user data as well as filesystem images, kernels, and ramdisks.
- New layer-2 networking subsystem based on software VLANS that no longer depends on VDE
- Added support for elastic IP assignment and security groups using the 'MANAGED' networking mode
- Web-based interface for cloud configuration
- Improved key management
- Image registration and image attribute manipulation
- Instance Metadata service (including userData)
- Kernel and ramdisk registration using standard tools
- Configurable cluster controller scheduling policy (ROUNDROBIN and GREEDY currently supported)
  - supports multiple VM types
- Support for new instance control operations:
  - Reboot instance
  - Get console output
- Various Web interface improvements, including cloud and Walrus configuration and editing of user info
- Various Node Controller improvements, including Walrus support, more robust VM control, and better diagnostics
- Revamped logging throughout, with five debug levels

For a more detailed list, see the 1.4 Changelog:

http://eucalyptus.cs.ucsb.edu/wiki/ChangeLog_v1.4

For a Eucalyptus compatibility matrix of supported Amazon features:

http://eucalyptus.cs.ucsb.edu/wiki/API_v1.4

Eucalyptus Administrator's Guide (1.4)

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).

Prerequisites

1. For compiling from source ¶

- C compilers
- Java Developer Kit (SDK) version 1.6 or above
- Apache ant 1.6.5 or above
- Curl development package
- openssl development package

2. For running Eucalyptus ¶

There are a few different Eucalyptus components that run on either a cluster 'front-end', or on a cluster '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.
- **Apache ant** is needed to run the Cloud Controller.
- Eucalyptus *requires* the unlimited-strength policy files for JCE for the Java Virtual Machine. *The current installer installs them automatically*. Thus, if you are in a country where this is illegal, DO NOT install Eucalyptus. If you are unsure of your country's cryptography import restrictions, Sun's documentation may be able to answer that question: see "Other Downloads" at the bottom of http://java.sun.com/javase/downloads/index.jsp (for JDK 1.6).
- 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".
- DHCP Server compatible with ISC DHCP Daemon version 3.0.X (dhcp3-server)

- Eucalyptus supports several virtual networking 'modes' (see Eucalyptus Networking for details). Installing the following packages allow you to use all modes.
  - iptables,
  - vlan (vconfig).

- If installing from RPMs, the front-end will need the `xen-libs` RPM to satisfy dependencies, even though Xen does not have to be installed.

Node run-time dependencies

- Eucalyptus 1.4 supports **Xen** (version >= 3.0.x). To enable Eucalyptus to control Xen, ensure that the UNIX socket interface is enabled in **/etc/xen/xend-config.sxp** configuration file by the line **(xend-unix-server yes)**. Eucalyptus interacts with the hypervisor through libvirt (version >= 0.6.0), so make sure that your local libvirt installation is functional on any machine that is to run a Eucalyptus Node Controller.
- Eucalyptus supports several virtual networking 'modes' (see Eucalyptus Networking for details). Installing the following packages allow you to use all modes.
  - vlan (vconfig),
  - bridge-utils,
  - xen-utils.

All Eucalyptus components

- You *must* be **root** to install and run Eucalyptus. This document assumes that all commands will be executed as root.

Attention Rocks users: Eucalyptus 1.4 can be installed on a Rocks-based cluster of version 5 or higher. To satisfy the prerequisites, please, install Java on the front-end and the **xen** roll in each of your virtual machine containers. The JDK installed by the **java** roll of the current version of Rocks is unfortunately insufficient, so you will need to install JDK 1.6.0 "manually". For our testing we used Sun's JDK, which can be found at http://java.sun.com/javase/downloads/index.jsp.

3. For interacting with Eucalyptus ¶

To interact with Eucalyptus, you need to install functioning EC2 command-line tools from Amazon. The latest version of these tools that we support are ec2-api-tools-1.3-30349 and ec2-ami-tools-1.3-26357.

Installing Eucalyptus (1.4)

Before you proceed with the installation, be sure to take a look at the list of Eucalyptus' prerequisites.

A Eucalyptus cloud setup consists of three components -- the cloud controller, the cluster controller(s) and node controller(s). The cloud controller is a Java program that, in addition to high-level resource scheduling and system accounting, offers a Web services interface and a Web interface to the outside world. Cluster controller and node controller are written in C and deployed as Web services inside Apache.

Communication among these three types of components goes over SOAP with WS-security. There is one cluster controller per cluster, running on the head node; there is one node controller per each compute node, running as root in dom0. So, if you are installing Eucalyptus on one cluster, then one cloud and one cluster controller should be deployed on the head node and one node controller should be deployed on each compute node.

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

Eucalyptus can be installed from source or using a set of RPM packages. The former method is more general and should work on practically any Linux system, the latter should work on most RPM-based systems. (In case you're curious, converting RPMs into DEBs with `alien` will **not** work.) Furthermore, we have special advice for those using Rocks-based clusters.

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

Installing Eucalyptus from source (1.4)

NOTE - If you are upgrading from a Eucalyptus 1.3 or older installation, please consult the Upgrade Documentation for instructions that will explain how to preserve user account information and re-import the images. If you are re-installing version 1.4, you may want to delete cached images on all nodes (in `INSTANCE_PATH/eucalyptus/cache` so they are not wasting disk space).

1. Download Eucalyptus ¶

Download

- eucalyptus-1.4-src.tar.gz (Eucalyptus source) and
- eucalyptus-1.4-src-deps.tar.gz

From the Eucalyptus Web site:

- http://eucalyptus.cs.ucsb.edu/downloads

Unpack the Eucalyptus source:

```
tar zvxf eucalyptus-1.4-src.tar.gz
```

Now you should have a directory eucalyptus-1.4. 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:

```
cd eucalyptus-1.4
export EUCALYPTUS_SRC=`pwd`
export EUCALYPTUS=/opt/eucalyptus
```

2. Dependencies ¶

To install Eucalyptus, you need to build packages that Eucalyptus depends on, which we provide in the above-mentioned package eucalyptus-1.4-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-1.4-src-deps.tar.gz
mkdir -p $EUCALYPTUS/packages/
```

Build and install the dependencies as follows:

a. Axis2 ¶

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

b. Apache (Httpd) ¶

The Eucalyptus node and cloud controller depend on Apache 2. We rely on running apache as root, therefore we recommend that apache 2 be installed separately from the webserver that might be included as part of your distribution.

```
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf httpd-2.2.8.tar.gz
cd httpd-2.2.8
CFLAGS="-DBIG_SECURITY_HOLE" ./configure --prefix=$EUCALYPTUS/packages/httpd-2.2.8 --with-included-apr
make ; make install
```

c. Axis2/C ¶

```
export AXIS2C_HOME=$EUCALYPTUS/packages/axis2c-1.5.0
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf axis2c-src-1.5.0.tar.gz
cd axis2c-src-1.5.0
./configure --with-apache2=$EUCALYPTUS/packages/httpd-2.2.8/include --prefix=${AXIS2C_HOME}
make ; make install
```

d. Rampart/C ¶

```
export AXIS2C_HOME=$EUCALYPTUS/packages/axis2c-1.5.0
export LD_LIBRARY_PATH=${AXIS2C_HOME}/lib:$LD_LIBRARY_PATH
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf rampartc-src-1.2.0.tar.gz
cd rampartc-src-1.2.0
./configure --prefix=${AXIS2C_HOME} --enable-static=no --with-axis2=${AXIS2C_HOME}/include/axis2-1.5.0
make ; make install
```

Now edit the file $AXIS2C_HOME/axis2.xml: search for "Security" and change

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

to

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

and save the file.

e. Libvirt ¶

```
cd $EUCALYPTUS_SRC/eucalyptus-src-deps/
tar zvxf libvirt-0.4.6.tar.gz
cd libvirt-0.4.6
./configure --prefix=$EUCALYPTUS/packages/libvirt-0.4.6 --without-storage-disk --without-storage-iscsi --without-storage-fs --without-storage-lvm --without-libvirtd --without-remote --without-lxc --without-kvm --witho
make ; make install
```

f. Other software ¶

Additionally, the following dependencies need to be met:

- Java Development Kit (JDK), version 1.6 or above
- Apache ant, version 1.6 or above
- GNU C compiler
- Make

NOTE: The Eucalyptus dependencies may, in turn, depend on libraries or packages not present in your installation: Please, refer to the documentation of the dependent package for help (e.g., see apache.org Web site for help with compiling the httpd server).

Finally, we recommend that you install functioning EC2 command-line tools from Amazon to interact with Eucalyptus. The latest version of these tools that we support are ec2-api-tools-1.3-30349 and ec2-ami-tools-1.3-26357.

3. Building Eucalyptus ¶

```
cd $EUCALYPTUS_SRC
./configure --with-axis2=$EUCALYPTUS/packages/axis2-1.4 --with-axis2c=$EUCALYPTUS/packages/axis2c-1.5.0 --with-libvirt=$EUCALYPTUS/packages/libvirt-0.4.6/ --prefix=$EUCALYPTUS
make ; make install
```

4. Eucalyptus on multiple machines ¶

You need now to instruct Eucalyptus on where it is installed. We provide the 'euca_conf' script as a convenience for setting up the configuration file located in '$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf'. (Expert users may edit this file manually instead of using 'euca_conf'.)

```
$EUCALYPTUS/usr/sbin/euca_conf -d $EUCALYPTUS $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf
```

At this point you're ready to push the software out to the nodes. If you installed Eucalyptus in its own directory, you can just sync the entire package to all hosts using whatever mechanism you typically use to push changes to nodes (rsync, for instance):

```
rsync -a $EUCALYPTUS/ hostname1:$EUCALYPTUS/
rsync -a $EUCALYPTUS/ hostname2:$EUCALYPTUS/
...
```

Eucalyptus on a Rocks cluster

If you want to install Eucalyptus on a Rocks cluster, you can now follow the regular RPM instructions. The Cloud Controller and the Cluster Controller will need to be installed on your Rocks front-end, and the Node Controller will need to be installed on any of your Rocks nodes that have been configured as 'VM Containers'.

Eucalyptus will **not** run on a Rocks virtual cluster!

If you have previously installed Eucalyptus on your Rocks cluster, you should disable the old Eucalyptus rolls:

```
rocks disable roll eucalyptus
```

rebuild the rocks distribution:

```
cd /home/install
rocks-dist dist
```

and finally reinstall the nodes as 'VM Containers'. You can then follow the RPM instructions to install and configure Eucalyptus. Keep in mind that the `java` roll in Rocks V (and V.I) includes JDK version 1.5 which is **not** enough to run Eucalyptus. You have to install the 1.6 JDK. For our testing we used Sun's JDK, which can be found at http://java.sun.com/javase/downloads/index.jsp.

Installing Eucalyptus using RPMs (1.4)

1. Download RPMs ¶

Eucalyptus RPM installation is broken up into several packages. Besides the general files required on all nodes, we separately package Eucalyptus running components: the cloud controller (-cloud RPM), the cluster controller (-cc RPM), and the node controller (-nc RPM). Furthermore, to simplify the installation, we offer copies of third-party packages that Eucalyptus depends on (Apache -compiled to run as root-, libvirt and axis2c) bundled into a single "deps" tarball.

E.g., for an x86_64 installation, download from http://eucalyptus.cs.ucsb.edu/downloads

- eucalyptus-1.4-2.x86_64.rpm
- eucalyptus-nc-1.4-2.x86_64.rpm
- eucalyptus-cc-1.4-2.x86_64.rpm

- eucalyptus-gl-1.4-2.x86_64.rpm
- eucalyptus-cloud-1.4-2.x86_64.rpm
- eucalyptus-1.4-rpm-deps-x86_64.tar.gz

Change **x86_64** into **i386** to download 32 bit RPMs instead.

2. Install RPMs on the front end ¶

On front end, where cloud controller and cluster controller will run, install the -cloud and -cc RPMs, along with the third-party dependencies:

```
rpm -Uvh euca-axis2c-1.4-1.x86_64.rpm \
         euca-httpd-1.4-1.x86_64.rpm \
         euca-libvirt-1.4-1.x86_64.rpm \
         eucalyptus-1.4-2.x86_64.rpm \
         eucalyptus-cloud-1.4-2.x86_64.rpm \
         eucalyptus-gl-1.4-2.x86_64.rpm \
         eucalyptus-cc-1.4-2.x86_64.rpm
```

2.3 Install RPMs on the nodes ¶

On the compute nodes, install the node controller RPM, along with the third-party dependencies:

```
rpm -Uvh euca-axis2c-1.4-1.x86_64.rpm \
         euca-libvirt-1.4-1.x86_64.rpm \
         euca-httpd-1.4-1.x86_64.rpm \
         eucalyptus-1.4-2.x86_64.rpm \
         eucalyptus-gl-1.4-2.x86_64.rpm \
         eucalyptus-nc-1.4-2.x86_64.rpm
```

Upgrading to Eucalyptus (1.4)

These are instructions for those who would like to upgrade to Eucalyptus 1.4 from earlier source-based or RPM-based installations of Eucalyptus. Both upgrade paths involve making a backup of the key state from the old installation. The last section of this document explains how to roll back to the previous installation using the backup.

1. Clean up Eucalyptus running state ¶

- Terminate all Eucalyptus instances

  ```
  ec2-terminate-instances ...      # (as admin)
  ```

- Shut down Eucalyptus on all nodes.

  ```
  /etc/init.d/eucalyptus stop
  ```

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

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

2. Install Eucalyptus 1.4 ¶

Both source- and RPM-based installations can be upgraded:

- **Option A:** Source-based installation upgrade:
  - Move away the old installation on the head-node and all compute nodes. E.g.:

    ```
    export EUCALYPTUS_OLD=/opt/eucalyptus-1.3
    mv /opt/eucalyptus $EUCALYPTUS_OLD
    ```

  - Follow the steps in the Source Code Installation section of the Administrator's Guide and, afterwards, **return here**.
  - Copy back the old database and keys (we assume that `$EUCALYPTUS` points to the new location, such as `/opt/eucalyptus`):

    ```
    cp $EUCALYPTUS_OLD/var/eucalyptus/db/eucalyptus.* $EUCALYPTUS/var/eucalyptus/db/
    rm $EUCALYPTUS/var/eucalyptus/db/*.lck
    cp $EUCALYPTUS_OLD/etc/eucalyptus/cloud.d/*.bks $EUCALYPTUS/etc/eucalyptus/cloud.d/
    ```

- **Option B:** RPM-based installation upgrade:
  - Follow the steps in the RPM-based Installation section of the Administrator's Guide and, afterwards, **return here**.

3. Update the configuration ¶

- Edit `eucalyptus.conf` file on head node. Do **not** start with the old one, rather, copy over matching parts to new one. (If you updated an RPM-based install, the new configuration is in `eucalyptus.conf.rpmnew` while `eucalyptus.conf` contains your old one.) Specifically, copy

over the values of the following variables:
    - `*_PORT`
    - `NODES`
    - `INSTANCE_PATH`

- Note that the following parameters were renamed:
    - `VNET_PUB_INTERFACE` renamed to `VNET_INTERFACE`
    - `VNET_DHCPDEAMON` renamed to `VNET_DHCPDAEMON`

- Due to dramatic changes in the virtual network implementation, there is, unfortunately, no simple way to convert the old network configuration to 1.4. We suggest that to set the remaining `VNET_*` variables appropriately you read the Network Configuration section of the Administrator's Guide and, afterwards, **return here**.

- On each compute node, go over the config file, comparing it with the old one, same as with the head-node file. (Naturally, you are free to create a new config file for compute-nodes and `rsync` it with all the nodes.)

- Start NCs, the CC, and the CLC. On all node do

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

- In a Web browser, load `https://headnode:8443/` and log in as `admin` with your old password. User accounts should still be there, but not the images. In the new 'Configuration' tab:
    - Click 'Add Cluster' and fill out the fields (You can use host/port/name values from `<cluster host="" port="" name="....">` in `$EUCALYPTUS/etc/eucalyptus/cloud.d/eucalyptus.xml`.)
    - Verify that the *Walrus path* is set to a reasonable path (this is where all images, kernels, ramdisks, and generic buckets will be stored).

- Synchronize node keys

  ```
  $EUCALYPTUS/usr/sbin/euca_sync_key -c $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf
  ```

- Verify that the nodes are back up (if not, see the Troubleshooting section.)

  ```
  ec2-describe-availability-zones verbose
  ```

4. Re-import old images ¶

Storage of disk images, kernels, and ramdisks has changed in 1.4 due to the introduction of Walrus, an S3-compatible bucket-based storage service. Instead of using `euca add_image`, these files are now added using standard EC2/S3 tools. To upgrade, the files will have to be re-added (and, hence, their EMIs will change!).

Note that if you made any changes to `config.xml` files of your images, you will have to re-integrate these changes into the Perl script named `$EUCALYPTUS/usr/share/eucalyptus/gen_libvirt_xml` which now generates the equivalent XML document *for all images* on-the-fly. Be sure to sync that script to all your nodes.)

```
cd (the registered images directory)
```

- For each image in the registered images directory, add the kernel, the ramdisk (if any), and the root file system image. See Image Management section for more information.

```
cd (an image dir)
gunzip *
ec2-bundle-image -i (kernel image name) --kernel true -d .
Please peicfy a value for the arch [x86_64]: ENTER
...
ec2-upload-bundle -b (unique bucket) -m (kernel image name).manifest.xml
ec2-register (unique bucket)/(kernel image name).manifest.xml
IMAGE   eki-.......
```

- If the image has a ramdisk, then:

  ```
  ec2-bundle-image -i (ramdisk image name) --ramdisk true -d .
  Please peicfy a value for the arch [x86_64]: ENTER
  ...
  ec2-upload-bundle -b (unique bucket 2) -m (ramdisk image name).manifest.xml
  ec2-register (unique bucket 2)/(ramdisk image name).manifest.xml
  IMAGE   eri-.......
  ```

```
ec2-bundle-image -i (disk image name) --kernel (eki-......) [--ramdisk (eri-......)] -d .
Please peicfy a value for the arch [x86_64]: ENTER
...
ec2-upload-bundle -b (unique bucket 3) -m (disk image name).manifest.xml
ec2-register (unique bucket 3)/(disk image name).manifest.xml
IMAGE   emi-......
```

- Check that it is there:

  ```
  ec2-describe-images
  ```

- repeat for other images

5. Clean up old disk state ¶

Once you are confident that the new installation is working, delete the old state on disk.

```
rm -rf $EUCALYPTUS_OLD
rm -rf (the template path)
rm -rf (the registration path)
rm /etc/defaults/eucalyptus    # (this is no longer used)
```

If you upgraded using RPM packages, delete on all nodes the backup of old state that was created during the upgrade:

```
rm /root/eucalyptus-pre1.4-rollback.tar
```

6. Rolling back to an earlier installation ¶

- Stop Eucalyptus 1.4 processes, if any, on all nodes
- **Option A:** Rolling back source-based 1.4 upgrade:
  - Remove any files added during the failed upgrade. For example:

    ```
    rm -rf $EUCALYPTUS
    ```

  - Move back the old installation on all nodes:

    ```
    mv $EUCALYPTUS_OLD $EUCALYPTUS
    ```

- **Option B:** Rolling back RPM-based 1.4 upgrade:
  - Remove the failed RPMs on all affected nodes (depending on the failure, you might have to use the `--nopreun` option)

    ```
    rpm -e eucalyptus-cloud eucalyptus-cc euca-httpd euca-axis2c euca-libvirt eucalyptus
    ```

  - Download and install the 1.3 RPMs on all nodes as discussed in the Administrator's Guide (it may be unnecessary to install `euca-vde` as it does not get removed during the upgrade). For example, on the front-end you can use:

    ```
    rpm -ivh eucalyptus-1.3-1.i386.rpm euca-httpd-1.0-1.i386.rpm eucalyptus-cloud-1.3-1.i386.rpm eucalyptus-cc-1.3-1.i386.rpm
    ```

  - Copy the old state saved during the upgrade process on all nodes:

    ```
    cd $EUCALYPTUS
    tar xvf /root/eucalyptus-pre1.4-rollback.tar
    ```

- Start Eucalyptus, as before

Eucalyptus Configuration (1.4)

This document describes the steps for configuring Eucalyptus after the software has been installed on all nodes (either from source or using RPMs).

1. First-time Configuration ¶

We provide the 'euca_conf' script as a convenience for setting up the configuration file located in '$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf'. Expert users may edit this file manually instead of using 'euca_conf'. If you installed from RPMs, $EUCALYPTUS is /opt/eucalyptus/.

To configure eucalyptus you need to specify which services to start (cloud, cluster controller and/or node controller). On the front-end:

```
$EUCALYPTUS/usr/sbin/euca_conf -cc Y -cloud Y -nc N $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf
```

Next, add a list of hostnames on which you plan to run node controllers:

```
$EUCALYPTUS/usr/sbin/euca_conf -nodes "<hostname1> ... <hostnamen>" $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf
```

Eucalyptus provides some options when it comes to configuring your VM virtual network. By default, we enable the simplest but least feature-ful networking mode. Please consult the Eucalyptus Networking document if you wish to try other modes that will enable more features (security groups, elastic IPs, etc.).

On each compute node, create a local directory where VM images are placed at VM run-time. Instruct the nodes to run only the node controller, and point it to the path of VM images. This path is used to store temporary VM images and it's important that it be empty (everything in it will be removed!).

```
for x in hostname1 hostname2 ... hostnameN ; do \
  ssh $x "mkdir -p /usr/local/instances/; $EUCALYPTUS/usr/sbin/euca_conf -cc N -cloud N -nc Y -instances /usr/local/instances $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf"
done
```

Finally, ensure that the networking settings in 'eucalyptus.conf' on each of your nodes is configured properly. For instance, correct values for VNET_INTERFACE and VNET_BRIDGE may differ from your front-end. See Eucalyptus Networking for more details.

2. Running Eucalyptus ¶

To start eucalyptus run

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

on each of your systems running eucalyptus. To stop it, run:

```
$EUCALYPTUS/etc/init.d/eucalyptus stop
```

If you installed from RPMs you can now skip to step 3. If you installed from source and you want to have eucalyptus started automatically when your machines are (re)booted, you can add the following symlink

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

and add the symlink 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 defaults
```

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

```
chkconfig eucalyptus on
```

3. First-time runtime setup ¶

To configure eucalyptus, after you started all components, login to

> https://localhost:8443

where you should substitute localhost with the name of the host running the cloud controller. (WARNING: 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/password which are set to admin/admin. Upon logging in you will be guided through three first-time tasks:

1. You will be forced to change the admin password.
2. You will be asked to set the admin's email address.
3. You will be asked to confirm the URL of the Walrus service (the storage component of Eucalyptus) which should start with the hostname or IP address of the cluster head node where you are installing the ClC.

After completing the first-time tasks, you will see the 'Configuration' tab. At the very least, to enable Eucalyptus, you will need to add a cluster by clicking the 'Add Cluster' button. For a single-cluster installation, the hostname is likely to be same as the admin web interface hostname to which you are connected. Be sure to click 'Save clusters' before continuing. This generates cryptographic keys that you will need to manually propagate to all nodes (see instructions below).

To finish the first-time configuration, propagate the cryptographic keys to all nodes:

```
$EUCALYPTUS/usr/sbin/euca_sync_key -c $EUCALYPTUS/etc/eucalyptus/eucalyptus.conf
```

This command uses rsync and ssh for key propagation, so you may be prompted for the password of each of your cluster nodes.

Finally, to use the system with the EC2 client tools, you must generate user credentials. Click the 'Credentials' tab and download your certificates via the 'Download certificates' button. You will be able to use these x509 certificates with Amazon EC2 tools and third-party tools like rightscale.com.

Create a directory, for example $HOME/.euca,

```
mkdir $HOME/.euca
```

unpack the credentials into it, and source the included 'eucarc':

```
. $HOME/.euca/eucarc
```

Note that you will have to source this file every time you intend to use the EC2 command-line tools, or you may add it to your local default environment.

Eucalyptus Network Configuration (1.4)

Eucalyptus versions 1.4 and higher includes a highly configurable VM networking subsystem that can be adapted to a variety of network environments. There are three 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 three 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), and 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).

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 SYSTEM 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.

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.

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 Xen bridge device that is connected to your local ethernet. In Xen 3.0 (and some other versions), the name of the bridge, by default, was 'xenbr0'. If you have such an installation, specify it like so:

```
VNET_BRIDGE="xenbr0"
```

In Xen 3.2 and higher, the name of the bridge (most of the time) is set to the name of your ethernet device (generally 'eth0'). If this is the case on your system, set configure Eucalyptus like so:

```
VNET_BRIDGE="eth0"
```

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. Use the 'brctl show' command to inspect the status of your local bridges. Note that your front-end machine may not have any bridges if Xen is not installed (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, start an instance and log in to the node where the instance is running. Run 'xm list' to find the Xen ID that your instance is running under. Then, look at the output of 'brctl show', it should look something like this (assuming your VNET_BRIDGE is set to 'eth0', and the Xen ID of your instance was '18'):

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

note that Eucalyptus has correctly attached the VM's 'eth0' interface (vif18.0) to the bridge ('eth0') that is being used to attach VMs to the local ethernet ('peth0'). 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_INTERFACE
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_INTERFACE="eth0"
```

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>"
```

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 (typically 'xenbr0' on Xen 3.0 and 'eth0' on Xen >= 3.2). Run the command 'brctl show' to inspect your nodes' bridge setup.

```
VNET_BRIDGE="xenbr0"
```

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, start an instance and log in to the node where the instance is running. Run 'xm list' to find the Xen ID that your instance is running under. Then, look at the output of 'brctl show', it should look something like this (assuming your VNET_BRIDGE is set to 'eth0', and the Xen ID of your instance was '18'):

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

note that Eucalyptus has correctly attached the VM's 'eth0' interface (vif18.0) to the bridge ('eth0') that is being used to attach VMs to the local ethernet ('peth0'). 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 that Eucalyptus currently 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_INTERFACE
VNET_DHCPDAEMON
*VNET_DHCPUSER
VNET_SUBNET
VNET_NETMASK
VNET_DNS
VNET_ADDRSPERNET
*VNET_PUBLICIPS
```

On each node:

```
VNET_MODE="MANAGED"
VNET_INTERFACE
```

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_INTERFACE), 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_INTERFACE), 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_INTERFACE="eth0"
```

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_INTERFACE set properly. For example, with current Xen versions, this parameter (when your node's Xen bridge is 'eth0') is typically:

```
VNET_INTERFACE="peth0"
```

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>"
```

CAVEATS - When Eucalyptus is running in MANAGED mode, you cannot currently run an entire eucalyptus installation o n 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_INTERFACE 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_INTERFACE="peth0") to:

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

If this is not the case, check your VNET_INTERFACE 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 'ec2-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'.

Managing Eucalyptus Images (1.4)

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.

The latest version of EC2 API and AMI tools that we support are ec2-api-tools-1.3-30349 and ec2-ami-tools-1.3-26357. Download BOTH before proceeding, if you have not 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 and optionally, a ramdisk for the kernel 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:

```
cd $EUCALYPTUS_SRC/eucalyptus-src-deps
tar zxvf euca-ttylinux.tgz
```

Add the kernel to Walrus, and register it with Eucalyptus (**WARNING**: your bucket names must not end with a slash! In addition, you cannot upload an image to a bucket that already exists using ec2-upload-bundle. You need to create a new bucket for each image, or delete the existing bucket):

```
ec2-bundle-image -i ttylinux/vmlinuz-2.6.16.33-xen --kernel true
ec2-upload-bundle -b kernel-bucket -m /tmp/vmlinuz-2.6.16.33-xen.manifest.xml
ec2-register kernel-bucket/vmlinuz-2.6.16.33-xen.manifest.xml
```

You might see a warning about trying to use a different EC2 public certificate. Enter 'y' and proceed.

Next, add the root filesystem image to Walrus:

```
ec2-bundle-image -i ttylinux/ttylinux.img
ec2-upload-bundle -b image-bucket -m /tmp/ttylinux.img.manifest.xml
ec2-register image-bucket/ttylinux.img.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:

```
ec2-bundle-image -i <path/to/my/>initrd.img --ramdisk true
ec2-upload-bundle -b <bucket_name> -m <path/to/my/>initrd.img.manifest.xml
ec2-register <bucket_name>/initrd.img.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 'ec2-bundle-image' step

   ```
   ec2-bundle-image -i <path/to/my/>vmimage.img --kernel <eki-XXXXXXXX> --ramdisk <eri-XXXXXXXX>
   ```

2. A user may choose a specific kernel/ramdisk at instance run time as an option to 'ec2-run-instances'

   ```
   ec2-run-instances <emi-XXXXXXXX> --kernel <eki-XXXXXXXX> --ramdisk <eri-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:

```
ec2-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:

```
ec2-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:

```
ec2-delete-bundle -a $EC2_ACCESS_KEY -s $EC2_SECRET_KEY --url $S3_URL -b <bucket> -p <file prefix> --clear
```

Eucalyptus Management (1.4)

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

1. Administrator's environment ¶

Currently, some administrative tasks can only be done through the command-line interface. When running commands, environment variable EUCALYPTUS must be pointing to the root of Eucalyptus installation for them to work properly.

If you installed Eucalyptus using RPMs, the value of $EUCALYPTUS is **/opt/eucalyptus**

```
export EUCALYPTUS=/path/to/eucalyptus
```

Adding this command to administrator's shell startup script, such as .profile, is probably a good idea.

2. Image Management ¶

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

3. Node Management ¶

Once you have a running Eucalyptus system you can add and remove nodes (systems running Node Controllers) by editing `$EUCALYPTUS/etc/eucalyptus/eucalyptus.conf` and editing the list of `NODES`:

```
NODES="vm-container-0-0 vm-container-0-1"
```

Whenever you add new nodes into the system, be sure to immediately propagate cryptographic keys to them as follows:

```
${EUCALYPTUS}/usr/sbin/euca_sync_key
```

4. User Management ¶

4.1 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.

4.2 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.

4.3 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.

Troubleshooting

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

1. Restarting ¶

If an administrator ever needs to stop/start a Eucalyptus front-end because of a configuration change, or if the machine on which the front-end is running reboots unexpectedly, the administrator must terminate all running instances in the system before bringing Eucalyptus back online. (It is possible to restart the cloud controller using `/etc/init.d/eucalyptus restart` on the head-node without affecting the rest of the system, but then some of the configuration is not reloaded. Doing `stop` followed by `start` on the head-node will reload the configuration, but will also destroy the virtual network setup among the running VMs, making them inaccessible.)

If the restart is planned, the administrator can use the client tools to terminate all users instances before stopping/reconfiguring/starting Eucalyptus. If the restart was unplanned (front-end machine crashes), the admin can try to start Eucalyptus and immediately terminate all running instances, or can manually stop all eucalyptus components, destroy all running Xen instances using 'xm shutdown' or 'xm destroy' on the nodes, and starting all Eucalyptus components.

2. Diagnostics ¶

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 (source `eucarc`), and use the following command to see the 'status' of your cloud:

```
ec2-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
...
```

If the output is empty, missing even the *cluster* then first make sure that you've added one (consult the 'Configuration' tab of the admin Web interface) and then consult Cloud Controller (CLC) logs and Cluster Controller (CC) logs, described below, to figure out why CLC is not getting valid status information from the CC.

If the output of `ec2-describe-availability-zones verbose` has the cluster mentioned, but is missing some or all of the nodes, then you will need to figure out why the Cluster Controller (CC) is not getting valid status information from the Node Controller (NC) on that node. To do so, start with the CC logs, which may show that CC is trying to connect to an NC on the wrong host or port. If CC is invoking the right NC endpoint, then consult the NC logs.

Log files ¶

On each machine running a Eucalyptus component, the log files 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'. If these files do not exist, then CC is not invoking the NC on the node. 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).

3. Specific errors ¶

- If `ec2-run-instances` returns

  ```
  FinishedVerify PROBLEM: Not enough resources available. MSG-TYPE: RunInstancesType`
  ```

  then your Eucalyptus installation is running at maximum capacity. Most likely its capacity is actually 0 because of a misconfiguration. See Diagnostics above.

Eucalyptus User's Guide (1.4)

This guide is meant for people interested in using an existing installation of Eucalyptus. (If you have a cluster that you would like to install Eucalyptus on, then take a look at the Administrator's Guide first.)

Getting Started Using Eucalyptus (1.4)

We will guide you through getting access to a Eucalyptus-based cloud, as well as installing and using tools for controlling virtual instances. Those familiar with Amazon's EC2 system will find most of these instructions familiar because Eucalyptus can be used with EC2's command-line tools.

1. Sign up

If you are using the Eucalyptus Public Cloud, use mayhem9.cs.ucsb.edu instead of **your.cloud.server**.

Load in your browser the Web page of the Eucalyptus cloud installation that you would like to use. Ask your system administrator for the URL if you don't know it. (The URL will be of the form *https://your.cloud.server:8443/*, where *your.cloud.server* is likely to be the front-end of the cluster.)

Click the "Apply" link and fill out the form presented to you. You may not be able to use the system until the (human) administrator receives the notification of your application and approves it. The more information you supply the easier it may be for the administrator to make the decision.

Load the confirmation URL that you receive in the approval email message from the cloud administrator. **Log in** to the system with the login and password that you chose when filling out the application form.

2. Obtain Credentials ¶

Once you have logged in, you will see the 'Generate Certificate' button under the 'Credentials' tab. Generating a certificate for your account is necessary before you can use Amazon's EC2 command-line tools for querying and controlling Eucalyptus instances. Currently, the Web interface to Eucalyptus is limited and, hence, the use of command-line tools is practically inevitable.

Click the button to generate the certificate and save it. You can keep these keys in a secure place on any host. The following command-line instructions apply to any Unix-flavored machine with bash (not necessarily the cluster where Eucalyptus was installed). (See Amazon's Getting Started Guide for the similar instructions to use under Windows.)

Unzip the keys using the following command and **protect** them from exposure. The zip-file contains two files with the .pem extension; these are your public and private keys.

```
mkdir ~/.euca
cd ~/.euca
unzip name-of-the-key-zip.zip
chmod 0700 ~/.euca
chmod 0600 ~/.euca/*
```

3. Install EC2 command-line tools ¶

Download the EC2 command-line tools from Amazon.

```
wget http://s3.amazonaws.com/ec2-downloads/ec2-api-tools-1.3-30349.zip
unzip ec2-api-tools-1.3-30349.zip
```

If you are using the Eucalyptus Public Cloud, use mayhem9.cs.ucsb.edu instead of **your.cloud.server**.

Set the following environment variables and source the 'eucarc' file that came with your credentials to set other crucial Eucalyptus environment variables:

```
export EC2_HOME=/path/to/installed/ec2-commandline-tools
export PATH=$PATH:$EC2_HOME/bin
source ~/.euca/eucarc
```

Now you should be ready to start using the tools. To test if the tools work (and if the cloud server is running properly), execute the following EC2 command:

If you get an **Invalid timestamp** error when running any of the ec2 commands, make sure the clock on your client machine (and the server, if you are in charge of it) is accurate.

```
ec2-describe-availability-zones
```

In the output of the above command, you should see the cluster's front-end hostname displayed.

4. Quick Start ¶

Now you can begin running VM instances on the Eucalyptus cloud. Using the EC2 command-line tools, you can learn about installed images, start VM instances using those images, describe the running instances, and terminate them when you're finished with them.

The following EC2 commands will allow you to query the system:

```
ec2-describe-images
IMAGE <emi-id> ...

ec2-describe-instances
(will be empty until you start an instance, as shown below)

ec2-describe-availability-zones

ec2-describe-keypairs
(will be empty until you add key pairs, as shown below)
```

Before starting a VM, you need to create at least one key pair. This key pair will be injected into the VM, allowing you to SSH into the instance. Below we will use *mykey* as a handle, but you may choose any string you like instead:

```
ec2-add-keypair mykey >mykey.private
('mykey' is the name for the key in Eucalyptus, 'mykey.private' is the file to be used by ssh)

chmod 0600 mykey.private

ec2-run-instances <emi-id> -k mykey -n <number of instances to start>

ec2-describe-instances
(should now show the instance)
```

If your administrator has configured Eucalyptus to provide security groups and elastic IPs, you may be required to allow logins to your instance, allocate a public IP (if you have not done so before, check 'ec2-describe-addresses' as a reminder), and assign it to your running instance:

Allow 'ssh' connections from the Internet:

```
ec2-authorize default -P tcp -p 22 -s 0.0.0.0/0
```

Allocate a public IP if you have not done so already:

```
ec2-allocate-address
```

Associate an allocated IP with your running instance:

```
ec2-associate-address <IP from allocate> -i <instance ID>
```

Once the instance is shown as 'Running', it will also show two IP addresses assigned to it. You may log into it with the SSH key that you created:

```
ssh -i mykey.private root@<accessible-instance-ip>
```

To terminate instances, use:

```
ec2-terminate-instances <instance-id1> <instance-id2> ... <instance-idn>
```

Please, see Amazon's EC2 Getting Started Guide for more information about these command-line tools. Keep in mind that, depending on how the administrator has configured Eucalyptus, not all tools/operations are necessarily supported (security groups/elastic IPs). Consult your administrator for more information.

Interacting with Walrus (1.4)

Walrus is a storage service included with Eucalyptus that is interface compatible with Amazon's S3. Walrus allows users to store persistent data, organized as buckets and objects (see Amazon's S3 Getting Started Guide for more information). Walrus system options can be modified via the administrator web interface.

If you would like to use Walrus to manage Eucalyptus VM images, you can use Amazon's tools to store/register/delete them from Walrus.

Otherwise, you may use S3 Curl to interact with Walrus directly. S3 Curl is a command line tool that computes the signature and invokes curl.

You may create, delete, list buckets, put, get, delete objects, set access control policies, etc. Please refer to the Amazon S3 documentation for the S3 interface specification.

You will need the perl-Digest-HMAC package for S3 Curl, if it is not already installed on your system.

You will need to modify the file s3curl.pl in your favorite editor to change the hostname endpoint.

For example, change

```
my @endpoints = ( 's3.amazonaws.com' );
```

to

```
my @endpoints = ( 'your-host' );
```

where, *your-host* is the IP or the hostname on which Walrus runs. For authentication to succeed, it is **crucial** that *your-host* is the same as the host portion of the `$S3_URL` environment variable set by `eucarc`.

When modifying the file **s3curl.pl**, please specify just the hostname and not the port number or the entire path (e.g., you would specify **xyz.com** instead of **xyz.com:8773/services/Walrus**. However, you need to use the entire path in the command-line when using the tool).

Be sure to source your 'eucarc' file before running the commands below.

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY <curl options>
```

It is preferable to add your credentials to the ".s3curl" config file. Please read the README file bundled with s3curl for details.

- For example, to create a bucket,

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --put /dev/null -- -s -v $S3_URL/bucketName
```

where bucketName is the name of the bucket that you want to create.

- To put an object,

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --put <filename> -- -s -v $S3_URL/bucketName/objectName
```

where objectName is the name of the object that you want to create.

- To get the MD5 checksum, size and last modification time for an object.

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --head -- -s -v $S3_URL/bucketName/objectName > object.head
```

```
cat object.head
```

- To get an object

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --get -- -s -v $S3_URL/bucketName/objectName > object
```

- To delete an object,

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --del -- -s -v $S3_URL/bucketName/objectName
```

- To delete a bucket,

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --del -- -s -v $S3_URL/bucketName
```

Note that, according to the S3 specification, a bucket needs to be empty before it can be deleted.

You may pipe the output through "xmlindent" (you might need to install it, if it is not already installed on your system).

- For example, to get the access control policy for a bucket,

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --get -- -s -v $S3_URL/bucketName?acl | xmlindent > bucket.acl
```

- To get the contents of a bucket.

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --get -- -s -v $S3_URL/bucketName  | xmlindent > bucket.list
```

- To list all buckets for a user.

```
s3curl.pl --id $EC2_ACCESS_KEY --key $EC2_SECRET_KEY --get -- -s -v $S3_URL  | xmlindent > bucketlisting
```

Known problems with Eucalyptus 1.4

- Changing cluster controller configuration via the Web UI may not work. If you need to change cluster controller's host or port, please, see the support forum for details. Proper fix will appear in v1.5.
- Multiple VM Types do not work. If more than one VM Type is used simultaneously the cloud controller becomes unusable and must be restarted. In those situations `ec2-run-instances` will report `Server: SERVICE: FinishedVerify PROBLEM: null MSG-TYPE: RunInstancesType`. We recommend setting parameters for all 5 VM Types to the same values. The proper solution will appear in v1.5.
- After changing more than one value in VM Types and clicking "Save VmTypes" you may see the "Failed to save! (Make sure values in each column are ordered.)" error. If the values are actually in order, the solution is to change them one-at-a-time.
- `ec2-describe-availability-zones verbose` may incorrectly report the status of node certificates as 'false'. Unlike in version 1.3, non-administrative users cannot see the number of available nodes (this has been requested for the future versions).
- The file `axis2c.log` contains errors:

  ```
  [error] rampart_handler_util.c(241) [rampart][rampart_handler_utils] 0 parameter is not set.
  [error] error.c(94) OXS ERROR [x509.c:284 in openssl_x509_get_subject_key_identifier] oxs defualt error , The extenension index of NID_subject_key_identifier is not valid
  ```

  while the file `httpd-nc_error_log` (on the nodes) may contain errors:

  ```
  [error] server reached MaxClients setting, consider raising the MaxClients setting
  ```

  These errors are benign and can be ignored.
- EC2 command-line tools fail with *Server: An error was discovered processing the <wsse:Security> header. (WSSecurityEngine: Invalid timestamp The security semantics of message have expired)* Solution: Ensure that the clocks on the client and server machines are synchronized. This is not a Eucalyptus bug, but a consequence of the security policy enforced by the ec2 command-line tools.
- Support for multiple clusters can only be enabled manually

ChangeLog

Version 1.4 (2009-01-05)

- Added new networking subsystem that no longer depends on VDE
- Added support for elastic IP assignment and security using the 'MANAGED' networking mode
- Cluster controller scheduling policy can now be configured in eucalyptus.conf (ROUNDROBIN and GREEDY currently supported)
- Cluster controller now handles concurrent requests (no longer have to restrict apache to allow only one connection at a time)
- Added Walrus: a Amazon S3 interface compatible storage manager. Walrus handles storage of user data as well as filesystem images, kernels, and ramdisks.
- Node Controller improvements:
  - Retrieval of images from Walrus instead of NFS-mounted file system
  - New caching and cleanup code, including start-time integrity checks
  - On-the-fly script-based generation of libvirt XML configuration
  - Script-based discovery of node resources (no assumptions about stat)
  - MAX_CORES overrides actual number of cores both down and up
  - Moved libvirt errors to nc.log and suppressed harmless ones
  - Serialized some Xen invocations to guard against non-determinism
  - Added proper swap creation, also "ephemeral" disk support
  - More robust instance state reporting to Cluster Controller
- Web interface improvements:
  - Added cloud/Walrus configuration, including clusters and VM types
  - Revamped 'credentials' tab with new options to edit user information and hide/show "secret" strings
  - Editing of user information for the administrator, including confirmation dialog for deletion of users
  - User-initiated password recovery mechanism
  - Fixed a couple of bugs: ' ' in username, spurious double additions
- Cloud Controller:
  - User, Cluster, and System keys are now stored in PKCS12 keystores and have moved to var/eucalyptus/keys
  - Configuration is handled entirely through the Web interface
  - Clusters dynamically added/started/stopped
  - Webservices operations complete up to EC2 2008-05-05 (w/o EBS):
  - "Elastic IP" address support

  - Image registration and attribute manipulation
  - GetConsole and RebootInstances support
  - Working Security Groups support for clusters in MANAGED network mode
  - See website for additional details, extensions, and caveats: http://eucalyptus.cs.ucsb.edu/wiki/API_v1.4
  - Instance Metadata service (including userData)
  - Workaround to use Amazon's tools for registering kernels & ramdisks
- Revamped logging throughout, with five levels a la log4j
- More standard build procedure: configure; make; make install
- More robust start-time checking

Version 1.3 (2008-08-27)

- Added support for the new ec2 tools (1.3-24159 and newer)

Version 1.2 (2008-07-29) ¶

- Added stand-alone RPM packages for non-rocks installation
- Added image caching to reduce instance creation time
- Added instance networking configuration options to eucalyptus.conf
- Bug Fixes
  - Improved installation-time error checking
  - Removed possibility of instance ID collision
  - Improved VDE runtime management
  - Improved VDE cleanup
  - Resolved occasional NC instance loss problem
  - Resolved EC2 client timing issue that resulted in parsing errors on client

Version 1.1 (2008-07-01) ¶

- Added WS-security for internal communication
- Added URL Query Interface for interacting with Eucalyptus
- Cluster Controller improvements:
  - Instance caching added to improve performance under certain conditions
  - Thread locks removed to improve performance
  - NC resource information gathered asynchronously to improve scheduler performance
- Network control improvements:
  - Added ability to configure 'public' instance interface network parameters (instead of hardcoded 10. network)
  - Lots of reliability changes
- Cloud Controller improvements:
  - Pure in-memory database
  - Image registration over WS interface
  - Improved build procedure
- Web interface improvements:
  - For all users (query interface credentials, listing of available images)
  - For the administrator (addition, approval, disabling, and deletion of users; disabling of images)
- Numerous bug fixes, improving stability and performance. In particular, but not limited to:
  - Recovering Cloud Controller system state
  - Timeout-related error reporting
  - Slimmer log files, with timestamps

Version 1.0 (2008-05-29) ¶

- First public version (limited-feature binary-only beta)