



Eucalyptus 3.2.2 Administration Guide

2013-04-01 Eucalyptus Systems

Contents

Welcome.....	4
Overview of Eucalyptus.....	4
Accessing Eucalyptus.....	4
Command Line Interface.....	4
Eucalyptus Administrator Console Overview.....	4
Managing the Cloud.....	8
Cloud Overview.....	8
Cloud Best Practices.....	9
Working with vSphere.....	9
Securing Your Cloud.....	9
Configuring SSL.....	10
High Availability.....	11
Storage Volumes.....	13
Caching Images on the Cluster Controller.....	14
Cloud Tasks.....	15
Inspect System Health.....	15
View User Resources.....	16
List Arbitrators.....	17
Add a Node Controller.....	17
Remove a Node Controller.....	17
Restart Eucalyptus.....	18
Shut Down Eucalyptus.....	19
Back Up and Restore Eucalyptus.....	21
Managing Access.....	23
Access Overview.....	23
Access Concepts.....	23
Policy Overview.....	25
LDAP/AD Integration.....	34
Access Tasks.....	40
Use Case: Creating an Administrator.....	41
Use Case: Creating a User.....	42
Accounts.....	43
Groups.....	45
Users.....	50
Credentials.....	53
Synchronize LDAP/AD.....	55
Managing Images.....	56
Image Overview.....	57
Image Tasks.....	57
Add an Image.....	57

Browse and Install Images from EuStore.....	60
Modify an Image.....	63
Creating an Image.....	64
Create Windows Image.....	65
Install Base Windows OS.....	65
Install Eucalyptus Windows Integration.....	68
Configure Active Directory.....	71
Configure Remote Desktop.....	73
Run Sysprep.....	74
Convert the VMDK to an Image.....	74
Add Image to Eucalyptus.....	75
Create Linux Image (Xen/KVM).....	76
Associate a Kernel and Ramdisk.....	77
Bundle an Image for Amazon EC2.....	78
Bundle a Windows Instance.....	78
Managing Reporting.....	80
Reporting Overview.....	80
Reporting Best Practices.....	80
Reporting Tasks.....	81
Reporting Tasks: CLC.....	81
Reporting Tasks: Data Warehouse.....	81
Appendix.....	84
Troubleshooting Eucalyptus.....	84
Log Files.....	84
Network Information.....	86
Walrus and Storage.....	87
Access and Identities.....	87
Windows Images.....	87
Instances.....	89
High Availability.....	89
Recovering from a Failure: Walrus.....	90
Configuring SSL for the Admin Console UI.....	91
Advanced Storage Configuration.....	92
EMC VNX Advanced Configuration.....	92
NetApp Advanced Configuration.....	95
Glossary.....	97

Welcome

This is the Eucalyptus Administration Guide. The guide shows you how to access Eucalyptus with a web-based GUI and with command line tools. This guide also describes how to perform common administration tasks in three areas: system management, identity management, and resource management.

This document is intended to be a reference. You do not need to read it in order, unless you are following the directions for a particular task.

Overview of Eucalyptus

Eucalyptus is a Linux-based software architecture that implements scalable, efficiency-enhancing private and hybrid clouds within an enterprise's existing IT infrastructure. Because Eucalyptus provides Infrastructure as a Service (IaaS), you can provision your own resources (hardware, storage, and network) through Eucalyptus on an as-needed basis.

A Eucalyptus cloud is deployed across your enterprise's on-premise data center. As a result, your organization has a full control of the cloud infrastructure. You can implement and enforce various level of security. Sensitive data managed by the cloud does not have to leave your enterprise boundaries, keeping data completely protected from external access by your enterprise firewall.

Eucalyptus was designed from the ground up to be easy to install and non-intrusive. The software framework is modular, with industry-standard, language-agnostic communication. Eucalyptus is also unique in that it provides a virtual network overlay that isolates network traffic of different users as well as allows two or more clusters to appear to belong to the same Local Area Network (LAN).

Eucalyptus also is compatible with Amazon's EC2, S3, and IAM services. This offers you hybrid cloud capability.

Accessing Eucalyptus

There are two ways to interact with Eucalyptus. You can use the administrative command line interface for making requests to Eucalyptus, or you can use the web-based user interface, called the Eucalyptus Administrator Console.



Tip: This guide will show both CLI and Eucalyptus Administrator Console steps for performing a task, when the task can be performed by both methods.

Command Line Interface

Eucalyptus supports two command line interfaces (CLIs). The administration CLI is installed when you install Eucalyptus server-side components. The administration CLI is for maintaining and modifying Eucalyptus.

The other user CLI, called `Euca2ools`, can be downloaded and installed on clients. `Euca2ools` are for end users and can be used with both Eucalyptus and Amazon Web Services (AWS).

The commands used in this guide assume that the environment variables exported by a `eucarc` file for an administrative Eucalyptus user have been set. For more information, see the [Eucalyptus Installation Guide](#).



Important: If you haven't already done so, change the default password for the administration user. You can do this using the `euare-usermodloginprofile` or by logging in to the Eucalyptus Administrator Console. The first time you log in to the console, you are prompted for a new password.

Eucalyptus Administrator Console Overview

The Eucalyptus Administrator Console provides cloud administrators with a way to perform several management tasks in a web user interface.

The Eucalyptus Administrator Console provides **Quick Links** for standard administrative actions and queries. These links, located on the left side of the screen, provide a convenient way to navigate through the Eucalyptus Administrator Console. For example, if you click **Accounts**, the Eucalyptus Administrator Console displays the **Accounts** page, listing all accounts in your system. Any property of a link, e.g., an account ID, displays on the right side of the screen as a link.

For more experienced users, the Eucalyptus Administrator Console provides a robust search engine. You can search for information or tasks quickly by building your own search. Because the Eucalyptus Administrator Console is search-based, even the **Quick Links** and other returned URLs from searches are themselves searches. Because each link is search, any Eucalyptus Administrator Console link you bookmark is also a search.

So the Eucalyptus Administrator Console offers you two ways to get information: by search or by following links. To show the member users of an account, you can click **Accounts** in quick links, select the account, and then click on the **Member users** link in the **Properties** section. Or, you use the Search box and type:

```
user:account=<account_name>.
```

Signing in to the Eucalyptus Administrator Console

This section describes how to sign in to the Eucalyptus Administrator Console.

The Eucalyptus Administrator Console is web-based interface that allows you to manage your system, identities, and resources.

To sign in to the Eucalyptus Administrator Console:

1. Open a browser window and go to `https://<CLC_IP_address>:8443`
Your browser displays a warning.
2. Accept the self-signed SSL certificate and continue.
The Eucalyptus sign-in page displays.
3. Enter your account name in the **Account** field.
 - For system admins, the account name is `eucalyptus`.
4. Enter your user name in the **User** field.
5. Enter your password in the **Password** field.
6. Click the **Sign in** button.
The Eucalyptus Administrator Console **Start Guide** page displays.

You can now use the Eucalyptus Administrator Console to manage your system, identities, and resources.

Understanding the Eucalyptus Administrator Console

This section explains the components of the Eucalyptus Administrator Console screen.

The Eucalyptus Administrator Console screen has the following areas:

Header This area includes the logo, the link to a user profile setting menu, and the big search box.

User Profile The current login user identity displays on the right side of the logo area. It shows the user name and the account name of the user identity, in the format of

```
<user_name>@<account_name>
```

Click the profile name to display the user profile menu. The menu provides the following functions:

- **View/change profile:** Displays the search result of the current user. In the search result page, you can view or change your identity's profile.
- **View access key:** Displays the search result of the current user's access key.
- **Change password:** Displays a dialog to change password.
- **Download new credentials:** Downloads the current user's credential package in a zip file.

Quick Links The left side of the Eucalyptus Administrator Console screen contains the **Quick Links** area. This area provides links to various contents of the Eucalyptus Administrator Console.

The **Quick Links** area is organized into sections made up of two levels. The top level is a heading for that section. Under each heading is a second section that contains a list of links. Each link is a search query in the form of the URL. Click a link to return the associated search result. For example, **Your Keys** is a search query of all the access keys belonging to you.

You can hide the **Quick Links** area by clicking the arrow of the vertical separator between **Quick Links** and the main content area.

Main Content The center part of the main screen displays the main content, usually the search result list. In many content displays, the Eucalyptus Administrator Console displays a toolbar that contains action buttons. The bottom of the content area provides the page navigation controls.

The search result list usually has multiple columns, some of which are sortable. Click the title in the column to sort the column display. If the list is too long, the Eucalyptus Administrator Console partitions the list into multiple pages. By default, each page displays a maximum of 25 rows, but you can configure this number.

.

When you select an item in the main content area, the Eucalyptus Administrator Console highlights the entire row and displays the **Properties** area. To select multiple items, use the **Ctrl** key for individual items, or the **Shift** key for a continuous block of items.

Properties The **Properties** area displays the detailed information about a selected search result item. The properties are displayed in two columns: the property name is on the left, and the property value is on the right.

Working with the **Properties** area:

- The Eucalyptus Administrator Console displays values of editable properties in a white input box. If you make any changes to value, the Eucalyptus Administrator Console displays the **Save** button at the bottom. Click this button to save any changed values.
- Some properties are of complex types. For example, the list of member users of an account. In these cases, the property names are displayed in hyperlinks with a magnifying glass icon. These hyperlinks invoke a search query.
- Other properties display an "action" icon. For example, **Password** displays a pencil icon. Click that icon to change the password.
- The Eucalyptus Administrator Console allows you to customize the displayed information in **Properties**. Click the plus icon to add a new property to the display. Click the minus icon to delete a property from the display.
- Click the **X** next to the **Properties** title to hide the area.

Status The bar at the bottom of the main screen shows system status messages, log window toggle button and the software version (from left to right).

Logs Click the **LOG** button on the status bar to pop up the log window. The log window records important dashboard events, especially any operations that modify system states, e.g. adding a new account, etc. The log windows records the latest 1024 log messages.

Using Search

This section details the **Search** function in the Eucalyptus Administrator Console.

Experienced users can use search box to get any information provided by the dashboard. The basic syntax of a search is as follows:

```
<type>: <field1>=<value1>,<value2>
<field2>=<value1>,<value2>
```

The <type> specifies the information type provided by the Eucalyptus Administrator Console. Currently Eucalyptus supports the following types:

Type Name	Description	Fields
start	Start page	None
config	Service components configuration	None
account	Accounts	name , id
group	User groups	account , name , id, path , user
user	Users	account , name , id, path , enabled, registration, group, [custom keys...] <ul style="list-style-type: none"> User's custom keys can be used as fields. group field means the user has membership in that group.
policy	Eucalyptus policies	account , user , group , name , id, version , text
key	Access keys	account , user , id, active, user
cert	X509 certificates	account , user , id, revoked, active
image	VM images	None
vmtype	VM types	None
report	Report page	None



Note: In the table, the field names in bold font means that for that field, the query evaluator does a partial match for the value.

The minimal search query contains the type name and a colon. For example, to display the **Start Guide** page, you would enter:

```
start:
```

After the colon, you enter a list of conditions, if any are accepted by the type name. Each condition has a field name and a list of values. The field name and values are separated by an equal sign. There is no space between the field name and value. Separate values with a comma, and don't include a space. Separate multiple conditions with a space.

To evaluate the search query, all conditions must be satisfied. For each condition, only one of the value for the field needs to be matched. For example, to find all users in the accounts whose names contain "testaccount", and whose user names contain "user1" or "user2", and who are enabled, enter the following:

```
user:account=testaccount name=user1,user2 enabled=true
```

After entering a search query in the search box in the header area of the main screen, press the **Enter** key. The search result displays in the content area. The browser URL will also change to reflect the search. Actually, the search query itself is part of the URL (after the pound sign). For example:

```
https://localhost:8443/#account:name=test
```

In fact, you can type a search directly in the URL box of the browser. But remember that the URL itself is URL encoded. This also enables Eucalyptus to construct a search URL and add to any web page.

Managing the Cloud

The **System Management** section of the **Quick Links** area allows you to go to the **Start Guide** or the **Service Components** page.

Cloud Overview

Eucalyptus is comprised of six components: Cloud Controller, Walrus, Cluster Controller, Storage Controller, Node Controller, and an optional VMware Broker. Each component is a stand-alone web service. This architecture allows Eucalyptus both to expose each web service as a well-defined, language-agnostic API, and to support existing web service standards for secure communication between its components.

Cloud Controller	The Cloud Controller (CLC) is the entry-point into the cloud for administrators, developers, project managers, and end-users. The CLC queries other components for information about resources, makes high-level scheduling decisions, and makes requests to the Cluster Controllers (CCs). As the interface to the management platform, the CLC is responsible for exposing and managing the underlying virtualized resources (servers, network, and storage). You can access the CLC through command line tools that are compatible with Amazon's Elastic Compute Cloud (EC2) and through a web-based Eucalyptus Administrator Console.
Walrus	Walrus allows users to store persistent data, organized as buckets and objects. You can use Walrus to create, delete, and list buckets, or to put, get, and delete objects, or to set access control policies. Walrus is interface compatible with Amazon's Simple Storage Service (S3). It provides a mechanism for storing and accessing virtual machine images and user data. Walrus can be accessed by end-users, whether the user is running a client from outside the cloud or from a virtual machine instance running inside the cloud.
Cluster Controller	The Cluster Controller (CC) generally executes on a machine that has network connectivity to both the machines running the Node Controller (NC) and to the machine running the CLC. CCs gather information about a set of NCs and schedules virtual machine (VM) execution on specific NCs. The CC also manages the virtual machine networks. All NCs associated with a single CC must be in the same subnet.
Storage Controller	The Storage Controller (SC) provides functionality similar to the Amazon Elastic Block Store (Amazon EBS). The SC is capable of interfacing with various storage systems (NFS, iSCSI, SAN devices, etc.). Elastic block storage exports storage volumes that can be attached by a VM and mounted or accessed as a raw block device. EBS volumes persist past VM termination and are commonly used to store persistent data. An EBS volume cannot be shared between VMs and can only be accessed within the same availability zone in which the VM is running. Users can create snapshots from EBS volumes. Snapshots are stored in Walrus and made available across availability zones. Eucalyptus with SAN support lets you use your enterprise-grade SAN devices to host EBS storage within a Eucalyptus cloud.
Node Controller	The Node Controller (NC) executes on any machine that hosts VM instances. The NC controls VM activities, including the execution, inspection, and termination of VM instances. It also fetches and maintains a local cache of instance images, and it queries and controls the system software (host OS and the hypervisor) in response to queries and control requests from the CC. The NC is also responsible for the management of the virtual network endpoint.
VMware Broker	VMware Broker (Broker or VB) is an optional Eucalyptus component activated only in versions of Eucalyptus with VMware support. Broker enables Eucalyptus to deploy virtual machines (VMs) on VMware infrastructure elements. Broker mediates all interactions between the CC and VMware hypervisors (ESX/ESXi) either directly or through VMware vCenter. For more information about working with vSphere Server, see Working with vSphere .

Cloud Best Practices

This section details Eucalyptus best practices for your private cloud.

Working with vSphere

Eucalyptus with the VMware Broker option can create and manage virtual machines on all or a subset of vSphere infrastructure resources. Since Eucalyptus takes over the task of managing Virtual Machines, to avoid interfering with the operation of your cloud, it is important to avoid performing some operations through vSphere-specific tools, such as the vSphere Client. Conversely, since Eucalyptus does not manage vSphere hosts, network, or storage, the administrator will have to continue using vSphere-specific tools for other operations. The following is a comprehensive, but not exhaustive list of operations that should and should not be performed with vSphere-specific tools.

Actions that may be performed with vSphere tools at any time

- vSphere management tasks that do not involve resources (VMs, hosts, networks, datastores, folders, vCenter Server sessions, etc.) that have never been and are not being used by Eucalyptus.
- Adding new resources -- hosts, networks, datastores, folders -- to vCenter. Such resources may be discovered and used by Eucalyptus automatically, either immediately or after a VMware Broker restart, if the VMware Broker configuration allows that (see 'discover' option in the section on configuring the VMware Broker).
- While we recommend using Eucalyptus's API to manage instances, a VM created by Eucalyptus may be powered off using vSphere Server tools. (It is not necessary to delete such a VM as Eucalyptus will delete it.)
- Managing roles and permissions in ways that do not reduce privileges of Eucalyptus since the time it became active.
- Changing vCenter or ESX(i) host settings that do not interfere with ongoing sessions and operations. For instance, a license on a host utilized by Eucalyptus may be changed as long as the host remains operational.
- Changing of vCenter ID or Name (see below regarding the change of IP address).

Actions that may be performed when Eucalyptus is not active (specifically, when the VMware Broker is shut down):

- Deleting the templates (VMs whose names start with 'emi-'). Those will be recreated if needed, albeit at the cost of additional instance start-up delay. To control the space used by and the number of templates, use VMware Broker's configuration properties `vmwarebroker.vsphere_cache_limit_bytes` and `vmwarebroker.vsphere_cache_max_elements`.
- Managing roles and permissions for Eucalyptus-managed resources, as long as new roles, if any, are reflected in VMware Broker configuration (see 'login' parameter) and as long as no Eucalyptus-created running VMs are taken out of Eucalyptus's control.
- vCenter IP address may be changed as long as VMware Broker's configuration is modified accordingly. IP addresses of ESX hosts may be changed as long as there are no running Eucalyptus VMs on the host (furthermore, a change of IP address may require adjustment of configuration unless the host can be discovered).

Actions to avoid with vSphere tools at all times:

- Renaming, modifying the settings for, or cloning Eucalyptus-created inventory objects (the name of the top-level folder on vCenter, VM templates, VMs). This includes changing the virtual hardware characteristics of VMs created by Eucalyptus.
- Migrating, snapshotting, and failing over VMs or templates (emi-...) created by Eucalyptus to a different host or a different datastore with VMotion, VMware HA, or VMware DRS.
- Changing default ports (80 for HTTP and 443 for HTTPS).

Securing Your Cloud

Eucalyptus components receive and exchange messages using either Query or SOAP interfaces (or both). Messages received over these interfaces are required to have some form of a time stamp (as defined by AWS specification) to prevent message replay attacks. Because Eucalyptus enforces strict policies when checking timestamps in the received messages, for the correct functioning of the cloud infrastructure, it is crucial to have clocks constantly synchronized (for

example, with ntpd) on all machines hosting Eucalyptus components. To prevent user commands failures, it is also important to have clocks synchronized on the client machines.

Following the AWS specification, all Query interface requests containing the Timestamp element are rejected as expired after 15 minutes of the timestamp. Requests containing the Expires element expire at the time specified by the element. SOAP interface requests using WS-Security expire as specified by the WS-Security Timestamp element.

When checking the timestamps for expiration, Eucalyptus allows up to 20 seconds of clock drift between the machines. This is a default setting. You can change this value for the CLC at runtime by setting the `bootstrap.webservices.clock_skew_sec` property as follows:

```
euca-modify-property -p
bootstrap.webservices.clock_skew_sec=<new_value_in_seconds>
```

For additional protection from the message replay attacks, the CLC implements a replay detection algorithm and rejects messages with the same signatures received within 15 minutes.



Important: To protect against replay attacks, the CLC only caches messages for 15 minutes. So it's important that any client tools used to interact with the CLC have the Expires element set to a value less than 15 minutes from the current time. This is usually not an issue with standard tools, such as euca2ools and Amazon EC2 API Tools.

You can configure replay detection in the CLC to allow replays of the same message for a set time period. This might be needed to ensure that legitimate requests submitted by automated scripts closely together (such as two requests to describe instances issued within the same second) are not rejected as malicious. The time within which messages with the same signatures are accepted is controlled by the `bootstrap.webservices.replay_skew_window_sec` property. The default value of this property is three seconds. To change this value, enter the following command:

```
euca-modify-property -p
bootstrap.webservices.replay_skew_window_sec=<new_value_in_seconds>
```

If you set this property to 0, Eucalyptus will not allow any message replays. This setting provides the best protection against message replay attacks, but may break some of the client-side scripts that issue commands too quickly.

If you set this property to any value greater than 15 minutes plus the values of `ws.clock_skew_sec` (that is, to a value ≥ 920 sec in the default installation), Eucalyptus disables replay detection completely.

Configuring SSL

In order to connect to Eucalyptus using SSL, you must have a valid certificate for the Cloud Controller (CLC). You must also be running the Cloud Controller and Cluster Controller (CC) on separate machines.

Create a keystore

Eucalyptus uses a PKCS12-format keystore. If you are using a certificate signed by a trusted root CA, use the following command to convert your trusted certificate and key into an appropriate format:

```
openssl pkcs12 -export -in [YOURCERT.crt] -inkey [YOURKEY.key] \
-out tmp.p12 -name [key_alias]
```

Note: this command will request an export password, which is used in the following steps.

Save a backup of the Eucalyptus keystore, at `/var/lib/eucalyptus/keys/euca.p12`, and then import your keystore into the Eucalyptus keystore as follows:

```
keytool -importkeystore \
-srckeystore tmp.p12 -srcstoretype pkcs12 -srcstorepass [export_password] \
-destkeystore /var/lib/eucalyptus/keys/euca.p12 -deststoretype pkcs12 \
```

```
-deststorepass eucalyptus -alias [key_alias] \
-srckeypass [export_password] -destkeypass [export_password]
```

Enable the Cloud Controller to use this keystore

Run the following commands on the Cloud Controller (CLC):

```
euca-modify-property -p bootstrap.webservices.ssl.server_alias=[key_alias]
euca-modify-property -p \
bootstrap.webservices.ssl.server_password=[export_password]
```

Restart the CLC by running `service eucalyptus-cloud restart` or `/etc/init.d/eucalyptus-cloud restart`

Optional: Configure the Cloud Controller and Walrus to redirect requests on port 443 to port 8773

The Cloud Controller and Walrus listen for both SSL and non-SSL connections on port 8773. If you have other tools that expect to speak SSL on port 443, you should forward requests on that port to port 8773. For example, the following `iptables` command can be used:

```
iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-ports 8773
```

High Availability

High availability is the result of the combination of functionality provided by Eucalyptus and the environmental and operational support to maintain the constituent systems's proper operation. Eucalyptus provides functionality aimed at enabling highly available operation:

- Detection of service faults and monitoring of system health: gather service status, determine current service topology, admit requests which can be satisfied using only healthy services in that topology
- Tools for interrogating the system's health: access to service state information
- Error gathering to aid in determining the cause: access to fault information as it impacts service function
- Automated failover when redundant services are configured: removal of faulty services and enabling of healthy services
- Service state control: ability to remove individual component-services (when configured with HA pair) from operation without disrupting service
- Replacement/restoration of component-services: procedures for restoring/replacing a component service after a total-loss failure (e.g., disk failure, host combustion, etc.)

In addition to previously detailed deployment recommendations, delivering highly available services with Eucalyptus depends on appropriate operational and maintenance support. The following sections detail the related system functionality and procedures.

Understanding Service State

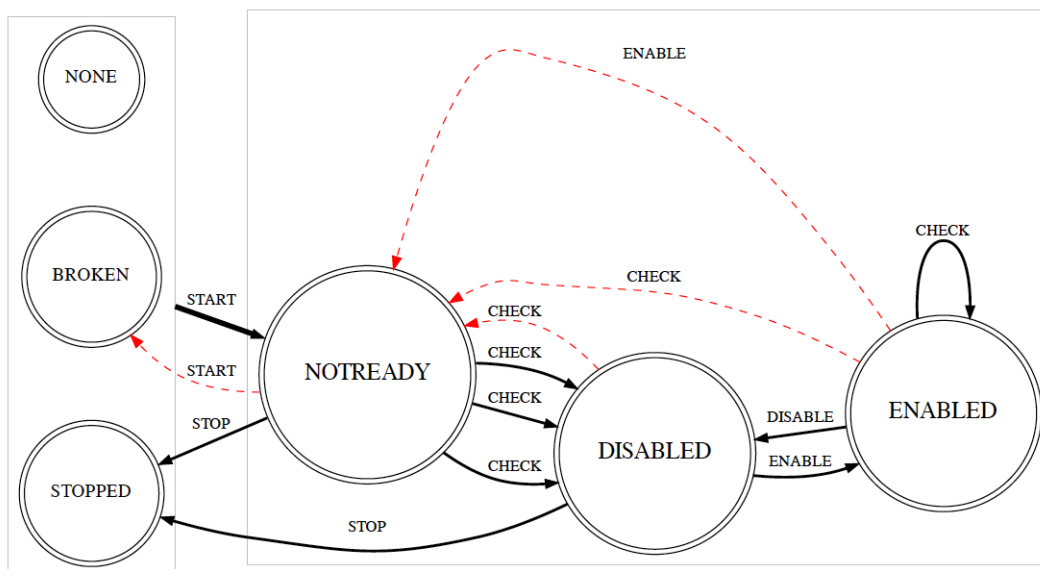
The system monitors service health and enables healthy services to process user requests while marking faulty services as being NOTREADY. Each component service is interrogated by the system to determine its current state. Faults are detected either:

- The service reporting a fault has been detected (for example, due to misconfiguration, dependency service failure, environmental fault, etc.)
- Failure to contact the service

The following table overviews the relevant states

State	Operational	In use by system	Description
ENABLED	Yes	Yes	Service is operating correctly and is selected for processing requests
DISABLED	Yes	No	Service is operating correctly but is not selected for processing requests
NOTREADY	No	No	Service is failing to operate correctly
BROKEN	No	No	Service is not contactable by the system
STOPPED	N/A	No	Service has been stopped by an administrator

The following diagram indicates the set of relevant states and transitions between them. Black arrows indicate a transition between states that is initiated by the system or an administrator request. Red errors indicate a failure to transition into the originating state that results in a transition to the destination error state.



Based on the collected service state, the system will:

- Attempt to advance previously non-functioning services to a functional state
- Determine whether any functioning services can be **ENABLED** and added to the set used for serving requests

On the Cloud Controller host, with eucalyptus admin credentials loaded, run `euca-describe-services` to see up-to-date service information including the state of each service as described in the above table.

Understanding System Availability

The impact of a service fault on the system's availability depends upon the deployment and configuration of the system. The following table details the scope a service fault can have on system availability for each component type.

Component	Scope (Fault Region)	Description
Cloud Controller	Cloud	The CLC is a cloud-wide service and must have at least one operation service.
Walrus	Cloud	Walrus is a cloud-wide service and must have at least one operation service.
Cluster Controller	Availability Zone	CCs are associated with a partition and service requests specific to an availability zone. Should an availability zone not have an operational CC, instance requests will be rejected for the corresponding zone.
Storage Controller	Availability Zone	Storage controllers are associated with a partition and service requests specific to an availability zone. Should an availability zone not have an operational storage controller volume and snapshot creation requests will be rejected for the corresponding zone
VMware Broker	Availability Zone	VMware Broker are associated with a partition and service requests specific to an availability zone. Should an availability zone not have an operational VMware Broker instance requests will be rejected for the corresponding zone
Arbitrators	User-facing Service Host	Arbitrators are associated with a host that runs user-facing component services (CLC, Walrus). Each host must have an operational Arbitrator. Should a component service host have a configured but faulty Arbitrator, a fail-stop condition occurs and locally hosted services report a NOTREADY error.
Node Controller	Compute Host	NCs are associated with each node and interact with the hypervisor to service node-specific requests.

A quick way to evaluate system availability is to determine whether:

- The cloud has an enabled CLC
- The cloud has an enabled Walrus
- The availability zone has an enabled CC
- The availability zone has an enabled SC
- The user-facing service host has one reachable Arbitrator per host (if you configure an Arbitrator)

Storage Volumes

Eucalyptus manages storage volumes for your private cloud. Volume management strategies are application specific, but some general guidelines are included here.

When setting up your Storage Controller, consider whether performance (bandwidth and latency of read/write operations) or availability is more important for your application. For example, using several smaller volumes will allow snapshots to be taken on a rolling basis, decreasing each snapshot creation time and potentially making restore operations faster if the restore can be isolated to a single volume. However, a single larger volume allows for faster read/write operations from the VM to the storage volume.

An appropriate network configuration is an important part of optimizing the performance of your storage volumes. For best performance, each Node Controller should be connected to a distinct storage network that enables the NC to communicate with the SC or SAN, without interfering with normal NC/VM-instance network traffic.

Eucalyptus includes configurable limits on the size of a single volume, as well as the aggregate size of all volumes on an SC. The SC can push snapshots from the SAN device, where the volumes reside, to Walrus, where the snapshots become available across multiple clusters. Smaller volumes will be much faster to snapshot and transfer, whereas large volumes will take longer. However, if many concurrent snapshot requests are sent to the SC, operations may take longer to complete.

Although an SC can manage an arbitrary number of volumes, intermittent issues have been reported with some hypervisors when attaching more than 16 volumes to a single NC. Where possible, limiting the number of volumes to no more than 16 per NC is advisable.

EBS volumes are created from snapshots on the SC or SAN, after the snapshot has been downloaded from Walrus to the device. Creating an EBS volume from a snapshot on the same cluster as the source volume of the snapshot will reduce delays caused by having to transfer snapshots from Walrus.

Caching Images on the Cluster Controller

To reduce calls to Walrus, Eucalyptus provides a means for images, including ramdisk and kernel images, to be cached on a cluster controller (CC). If this feature is enabled, when Eucalyptus starts an instance, it will first look for the instance image in the CC image cache location. If the image is not found in the CC image cache, it will be loaded from Walrus, and stored in the cache if space is available.

1. Edit `/var/eucalyptus/eucalyptus.conf` as follows:

a) Uncomment the `CC_IMAGE_PROXY` line and specify the IP of the CC host on which to cache images.

```
# Set this to make the CC cache images, kernels and ramdisks.  NCs must
# be able to reach the CC with the specified value.
CC_IMAGE_PROXY="192.168.0.100"
```

b) Set `CC_IMAGE_PROXY_PATH` to point to the location of the image cache.

```
# Set this to the location where the CC image proxy should store cached
# images.  The default is /var/lib/eucalyptus/dynserv/
CC_IMAGE_PROXY_PATH="/disk1/storage/cc_cache"
```

c) Set `CC_IMAGE_PROXY_CACHE_SIZE` to the maximum size of the image cache.

```
# Set this to the maximum size (in megabytes) of the CC image proxy cache.
# The default is 32768, or 32 gigabytes.
CC_IMAGE_PROXY_CACHE_SIZE="32768"
```



Important: Setting `CC_IMAGE_PROXY_CACHE_SIZE` to 0 will cause any attempts to create an instance from an uncached image to remain pending indefinitely. To disable image caching, comment out the `CC_IMAGE_PROXY` line.

2. Create a data directory at the location specified in `CC_IMAGE_PROXY_PATH`, and give the “eucalyptus” user full access to the directory.

```
mkdir -p /disk1/storage/cc_cache/data
chmod -R 777 /disk1/storage/cc_cache
```

3. Perform a clean restart of the cluster controller.



Important: A clean restart should only be performed when no instances are running, or when instance service interruption can be tolerated. A clean restart causes the CC to reset its networking configuration, regardless of whether or not it is in use. A clean restart of a CC managing active VMs can cause the network connectivity of those VMs to be irreparably lost.

```
service eucalyptus-cc cleanrestart
```

Cloud Tasks

Listing of the cloud-related tasks.

You can perform the following cloud-related tasks listed in this section:

- [Inspect System Health](#)
- [List Arbitrators](#)
- [Add a Node Controller](#)
- [Remove a Node Controller](#)
- [Restart Eucalyptus](#)
- [Shut Down Eucalyptus](#)
- [Back Up and Restore Eucalyptus](#)

Inspect System Health

Eucalyptus provides access to the current view of service state and the ability to manipulate the state. You can inspect the service state to either ensure system health or to identify faulty services. You can modify a service state to maintain activities and apply external service placement policies.

View Service State

Use the `euca-describe-services` command to view the service state. The output indicates:

- Component type of the service
- Partition in which the service is registered
- Unique name of the service
- Current view of service state
- Last reported epoch (this can be safely ignored)
- Service URI
- Fully qualified name of the service (This is needed for manipulating services that did not get unique names during registration. For example: internal services like reporting or DNS)

The default output includes the services that are registered during configuration, as well as information about the DNS service, if present. You can obtain additional service state information, such as internal services, by providing the `-system-internal` flag.

You can also make requests to retrieve service information that is filtered by either:

- current state (for example, NOTREADY)
- host where service is registered
- partition where service is registered
- type of service (for example, CC or Walrus)

When you investigate service failures, you can specify `-events` to return a summary of the last fault. You can retrieve extended information (primarily useful for debugging) by specifying `-events -events-verbose`.

Modify Service State

To modify a service:

Enter the following command on the CLC, Walrus, SC, or VMWareBroker machines:

```
eucalyptus-cloud stop
```

On the CC, use the following command:

```
eucalyptus-cc stop
```

If, for example, you have SCs that are correctly configured and operating in HA mode. However, you want to shut down the primary SC for maintenance. The primary SC is SC00 and the secondary SC is SC01. SC00 is `ENABLED` and SC01 is `DISABLED`.

To stop SC00 and cause SC01 to take over, you would enter the following command on SC00:

```
eucalyptus-cloud stop
```

To check status of services, you would enter:

```
euca-describe-services
```

When SC01 starts, the `eucalyptus-cloud` process on the host that SC00 is shutdown and maintenance tasks can be performed. When maintenance is complete, you can start the `eucalyptus-cloud` process on SC00. SC00 will enter the `DISABLED` state by default. You can chose to let SC01 continue to be the primary and SC00 will be the secondary.

If you want to designate SC00 as the primary, make sure no volumes or snapshots are being created and that no volumes are being attached or detached, and then enter on SC01:

```
eucalyptus-cloud stop
```

Monitor the state of services using `euca-describe-services` until SC01 is marked `DISABLED` and SC00 is `ENABLED`.

View User Resources

To see resource use by your cloud users, Eucalyptus provides the following commands with the `-verbose` flag:

- `euca-describe-groups verbose`: Returns information about security groups in your account, including output type identifier, security group ID, security group name, security group description, output type identifier, account ID of the group owner, name of group granting permission, type of rule, protocol to allow, start of port range, end of port range, source (for ingress rules) or destination (for egress rules), and any tags assigned to the security group.

- `euca-describe-instances verbose`: Returns information about your instances, including output type identifier, reservation ID, name of each security group the instance is in, output type identifier, instance ID for each running instance, EMI ID of the image on which the instance is based, public DNS name associated with the instance (for instances in the running state), private DNS name associated with the instance (for instances in running state), instance state, key name, launch index, instance type, launch time, availability zone, kernel ID, ramdisk ID, monitoring state, public IP address, private IP address, type of root device (ebs or instance-store), placement group the cluster instance is in, virtualization type (paravirtual or hvm), any tags assigned to the instance, hypervisor type, block device identifier for each EBS volume the instance is using, along with the device name, the volume ID, and the timestamp.
- `euca-describe-keypairs verbose`: Returns information about key pairs available to you, including keypair identifier, keypair name, and private key fingerprint.
- `euca-describe-snapshots verbose`: Returns information about EBS snapshots available to you, including snapshot identifier, ID of the snapshot, ID of the volume, snapshot state (pending, completed, error), timestamp when snapshot initiated, percentage of completion, ID of the owner, volume sized, description, and any tags assigned to the snapshot.
- `euca-describe-volumes verbose`: Describes your EBS volumes, including volume identifier, volume ID, size of the volume in GiBs, snapshot from which the volume was created, availability zone, volume state (creating, available, in-use, deleting, deleted, error), timestamp of the volume creation, and any tags assigned to the volume.

List Arbitrators

To see a list of Arbitrators running on your cloud:

- Enter the following command to display Arbitrators for the current CLC or Walrus:

```
/usr/sbin/euca-describe-services --system-internal
```

- Enter the following command to display Arbitrators on both primary and secondary CLCs or Walruses:

```
/usr/sbin/euca_conf --list-arbitrators
```

Add a Node Controller

Describes how to add a node to your system.

If you want to increase your system's capacity, you'll want to add more NC. To add an NC, perform the following tasks:



Note:

To add an ESXi host as a node controller, please see 'Re-generating VMWare Broker Configuration' in 'Configuring VMWare Support' section of the [Eucalyptus Installation Guide](#).



Caution: By default, the node controller uses the filesystem to perform key injection. This is potentially an unsafe practice. To disable key injection, set `DISABLE_KEY_INJECTION=1` in `eucalyptus.conf`.

1. Log in to the CLC and enter the following command:

```
/usr/sbin/euca_conf --register-nodes \ "[Node1_IP]; ...  
[NodeN_IP]; "
```

2. When prompted, enter the password to log into each node.

Eucalyptus requires this password to propagate the cryptographic keys.

Remove a Node Controller

Describes how to delete NCs in your system.

If you want to decrease your system's capacity, you'll need to decrease NC servers. To delete an NC, perform the following tasks.

Log in to the CLC and enter the following command:

```
/usr/sbin/euca_conf --deregister-nodes "<nodeName1> ... <nodeNameN>"
```

Restart Eucalyptus

Describes the recommended processes to restart Eucalyptus, including terminating instances and restarting Eucalyptus components.

You must restart Eucalyptus whenever you make a physical change (e.g., switch out routers), or edit the `eucalyptus.conf` file. To restart Eucalyptus, perform the following tasks in the order presented.



Note: Before you restart Eucalyptus, we recommend that you notify all users that you are terminating all instances.

1. *Shut Down All Instances*
2. *Restart the CLC*
3. *Restart Walrus*
4. *Restart the CC*
5. *Restart the SC*
6. *Restart a NC*

Shut Down All Instances

To terminate all instances on all NCs:

Enter the following command:

```
euca-terminate-instances <instance_id>
```

Restart the CLC

Log in to the CLC and enter the following command:

```
service eucalyptus-cloud restart
```

All Eucalyptus components on this server will restart.

Restart Walrus

Log in to Walrus and enter the following command:

```
service eucalyptus-cloud restart
```

Restart the CC

You can start the CC two different ways. A normal restart causes the CC to re-parse the `eucalyptus.conf` file without resetting the networking state of the cloud. This means that the CC maintains routes, firewall rules, IP assignments for running VMs. This is how you restart under normal circumstances.

A clean restart causes the CC to reset the networking configuration it maintains. The CC clears any existing network configuration regardless of whether it is in use. That is, a clean restart of a CC managing active VMs can cause the network connectivity they are using to be irreparably lost. We recommend that you only perform this type of restart when you change network modes.

Restart the CC (Normal)

To perform a normal restart:

Log in to the CC and enter the following command:

```
service eucalyptus-cc restart
```

Restart the CC (Clean)

To perform a clean restart:

Log in to the CC and enter the following command:

```
service eucalyptus-cc cleanrestart
```

Restart the SC

Log in to the SC and enter the following command:

```
service eucalyptus-cloud restart
```

Restart a NC

1. Log in to the NC and enter the following command:

```
service eucalyptus-nc restart
```

2. Repeat for each NC.

You can automate the restart command for all of your NCs. Store a list of your NCs in a file called `nc-hosts` that looks like:

```
nc-host-00
nc-host-01
...
nc-host-nn
```

To restart all of your NCs, run the following command:

```
cat nc-hosts | xargs -i ssh root@{ } service eucalyptus-nc restart
```

Shut Down Eucalyptus

Describes the recommended processes to shut down Eucalyptus.

There may be times when you need to shut down Eucalyptus. This might be because of a physical failure, topological change, backing up, or making an upgrade. We recommend that you shut down Eucalyptus components in the reverse order of how you started them. To stop the system, shut down the components in the order listed.



Note: Before you shut Eucalyptus down, we recommend that you notify all users that you are terminating all instances.

1. [Shut Down All Instances](#)
2. [Shut Down the NCs](#)
3. [Shut Down the CCs](#)
4. [Shut Down the Broker](#)
5. [Shut Down the SCs](#)
6. [Shut Down Walrus](#)

7. *Shut Down the CLC*

Shut Down All Instances

To terminate all instances on all NCs:

Enter the following command:

```
euca-terminate-instances <instance_id>
```

Shut Down the NCs

To shut down the NCs:

1. Log in as root to a machine hosting an NC.
2. Enter the following command:

```
service eucalyptus-nc stop
```

3. Repeat for each machine hosting an NC.

Shut Down the CCs

To shut down the CCs:

1. Log in as root to a machine hosting a CC.
2. Enter the following command:

```
service eucalyptus-cc cleanstop
```



Note: The cleanstop command causes the CC to reset and/or flush all of the existing networking state.

3. Repeat for each machine hosting a CC.

Shut Down the Broker

If your system uses the optional Broker component of Eucalyptus, shut it down by performing the following steps:

1. Log in as root to the machine running both the CC and the VMware Broker.
2. Enter the following command:

```
service eucalyptus-cloud stop
```



Tip: The eucalyptus-cloud stop command also shuts down a CLC, Walrus, and SC components co-located with the CC and VMware Broker to stop at the same time, in the correct order.

Shut Down the SCs

To shut down the SC:

1. Log in as root to the physical machine that hosts the SC.
2. Enter the following command:

```
service eucalyptus-cloud stop
```

3. Repeat for any other machine hosting an SC.

Shut Down Walrus

To shut down Walrus:

1. Log in as root to the physical machine that hosts Walrus.
2. Enter the following command:

```
service eucalyptus-cloud stop
```

Shut Down the CLC

To shut down the CLC:

1. Log in as root to the physical machine that hosts the CLC.
2. Enter the following command:

```
service eucalyptus-cloud stop
```

Back Up and Restore Eucalyptus

You can back up and restore Eucalyptus by completing the following tasks:

1. [Shut Down Eucalyptus](#)
2. [Back Up Eucalyptus](#)
3. [Restore Eucalyptus from a Backup](#)

Shut Down Eucalyptus

Describes the recommended processes to shut down Eucalyptus.

There may be times when you need to shut down Eucalyptus. This might be because of a physical failure, topological change, backing up, or making an upgrade. We recommend that you shut down Eucalyptus components in the reverse order of how you started them. To stop the system, shut down the components in the order listed.



Note: Before you shut Eucalyptus down, we recommend that you notify all users that you are terminating all instances.

1. [Shut Down All Instances](#)
2. [Shut Down the NCs](#)
3. [Shut Down the CCs](#)
4. [Shut Down the Broker](#)
5. [Shut Down the SCs](#)
6. [Shut Down Walrus](#)
7. [Shut Down the CLC](#)

Back Up Eucalyptus

Backing up Eucalyptus involves saving contents of the specific directories on each component. On the CLC host, these directories are:

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

On the Walrus host (which, depending on your set-up, might be the same as the CLC host):

- The Walrus buckets (the default location is `/var/lib/eucalyptus/bukkits`). You should only do this if you have enough spare disk space available. If the Walrus buckets location is an external storage location (like NFS), consult with your storage administrator to back up that location.

On the Storage Controller host (may be the same as the CLC host):

- the SC volumes (the default location is `/var/lib/eucalyptus/volumes`). You should only back this up if you are not using SAN support. If you are using a supported SAN device, consult with your storage administrator to back up volumes stored on the SAN.

To back up Eucalyptus:

1. 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). For example, on a single-cluster installation with default buckets and volumes paths:

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

2. Create a directory for storing these (\$BACKUP) in a location with enough disk space

Restore Eucalyptus from a Backup

In case your system experiences a failure, you can restore Eucalyptus from a backup by performing the following tasks:

1. Shut down any running Eucalyptus components. For further information about the shut down process, see [Shut Down Eucalyptus](#).
2. 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 (for example, `rpm -e` or `apt-get remove`).
 - Install the appropriate version. For more information, go to the Eucalyptus Installation Guide.



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

3. Replace the saved state.

- If you backed up with `cp`, enter the following:

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

- If you backed up with `tar`, enter the following:

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

Managing Access

Eucalyptus manages access control through an authentication, authorization, and accounting system. This system manages user identities, enforces access controls over resources, and provides reporting on resource usage as a basis for auditing and managing cloud activities. The user identity organizational model and the scheme of authorizations used to access resources are based on and compatible with the AWS Identity and Access Management (IAM) system, with some Eucalyptus extensions provided that support ease-of-use in a private cloud environment.

You can also perform user authentication by integrating Eucalyptus with an existing LDAP or Active Directory. In this case, the user, group and account information, and Eucalyptus Administrator Console login authenticate using the LDAP/AD service. This information cannot be changed from Eucalyptus side when LDAP/AD integration is turned on. However, other Eucalyptus-specific information about user, group and account is still stored within the local database of Eucalyptus, including certificates, secret keys and attached policies.

For more information about synchronizing an existing LDAP or Active Directory with Eucalyptus, see [LDAP/AD Integration](#).

Access Overview

The Eucalyptus design of user identity and access management provides layers in the hierarchical organization of user identities. This gives you refined control over resource access. The core of this design is compatible with the AWS Identity and Access Management (IAM) service. There are also a few Eucalyptus-specific extensions that meet the needs of enterprise customers.

Differences in Access from Eucalyptus 2

The concept of “user” has changed from when Amazon first introduced IAM. The original “user” has become the “account,” and the new “user” is an identity within the “account”. We made a similar change in the current Eucalyptus implementation of an IAM-compatible identity management system. If you upgrade of Eucalyptus 2 to Eucalyptus 3 is performed, the “users” in Eucalyptus 2 are converted into “accounts” in Eucalyptus 3. Please refer to the Eucalyptus upgrade section in the [Installation Guide](#) for the details about this identity conversion process.

In Eucalyptus 2, the access management ability was limited:

- The admin user had full access control of the system resources.
- Regular users controlled resources that they created.
- Certain resources could be shared among users, using mechanisms like image launch permissions and Walrus bucket ACLs.

Eucalyptus 3 introduces a two-tier hierarchy of user identities based on accounts. The access control, therefore, is provided at both tiers:

- Eucalyptus 2 style resource sharing is still available, but now implemented at account level.
- Within each account, fine-grained access control is provided by a policy that is fully compatible with IAM.

Access Concepts

This section describes what Eucalyptus access is and what you need to know about it so that you can configure access to your cloud.

User Identities

In Eucalyptus, user identities are organized into accounts. An account is the unit of resource usage accounting, and also a separate namespace for many resources (security groups, key pairs, users, etc.)

Accounts are identified by either a unique ID (UUID) or a unique name. The account name is equivalent to IAM’s account alias. It is used to manipulate accounts in most cases. However, to be compatible with AWS, the EC2 commands

often use account ID to display resource ownership. There are command line tools to discover the correspondence of account ID and account name. For example, `euare-accountlist` lists all the accounts with both their IDs and names.

An account can have multiple users, but a user can only be in one account. Within an account, users can be associated with Groups. Group is used to attach access permissions to multiple users. A user can be associated with multiple groups. Because an account is a separate name space, user names and group names have to be unique only within an account. Therefore, user X in account A and user X in account B are two different identities.

Both users and groups are identified by their names, which are unique within an account (they also have UUIDs, but rarely used).

Special Identities

Eucalyptus has two special identities for the convenience of administration and use of the system.

- The **eucalyptus** account: Each user in the eucalyptus account has unrestricted access to all of the cloud's resources, similar to the superuser on a typical Linux system. These users are often referred to as system administrators or cloud administrators. This account is automatically created when the system starts for the first time. You cannot remove the eucalyptus account from the system.
- The **admin** user of an account: Each account, including the eucalyptus account, has a user named admin. This user is created automatically by the system when an account is created. The admin of an account has full access to the resources owned by the account. You can not remove the admin user from an account. The admin can delegate resource access to other users in the account by using policies.

Credentials

Each user has a unique set of credentials. These credentials are used to authenticate access to resources. There are three types of credentials:

- An **X.509 certificate** is used to authenticate requests to the SOAP API service.
- A **secret access key** is used to authenticate requests to the REST API service.
- A **login password** is used to authenticate the Eucalyptus Administrator Console access.

You can manage credentials using the command line tools (the `euare-` commands) or the Eucalyptus Administrator Console. For more information about the command line tools, go to the [CLI Reference Guide](#).

In IAM, each account has its own credentials. In Eucalyptus, the equivalent of account credentials are the credentials of admin user of the account.

You can download the full set of credentials for a user or an account, including X509 certificate and secret access key, by:

```
/usr/sbin/euca_conf --get-credentials
```

or:

```
euca-get-credentials
```

or by using the **Download new credentials** in the Eucalyptus Administrator Console.

Whichever way you request the credentials, Eucalyptus returns the following:

- An arbitrary existing active secret access key
- A newly generated X509 certificate

Account Creation

You can create accounts two ways:

- Direct creation using command line tool or Eucalyptus dashboard by sys admin. The accounts created in this method will be available for accessing immediately.
- Registration process. One can apply for an account through the dashboard. The process is as follows:

1. The cloud user registers using the form on the dashboard interface.
2. An email will then be sent to the sys admin for review. Sys admin can approve or reject the application by invoking a URL included in the email. A notification email will be sent to the intended account owner.
3. If the application is approved, the account owner needs to invoke the URL included in the notification email to confirm the approval.
4. Once confirmed, the new account becomes available.

The account registration status can be found in Eucalyptus Administrator Console. The account registration status is actually associated with the account admin user. That means you can use the following command to obtain the same information:

```
euare-usergetattributes --delegate account -u admin --show-extra
```

Where the `--show-extra` option displays extra information of a user in the following order:

- Enabled status
- Registration status
- Password expires

The account registration status has the following values based on the status of registration process: `REGISTERED`, `APPROVED`, or `CONFIRMED`. An account that is not confirmed cannot be used or accessed. The system administrator can manipulate the account registration status in both dashboard and command line:

```
euare-usermod --delegate account -u admin --reg-status=status
```

The command line manipulation of the registration status does not send the notification emails. Unless you are experienced, we recommend that you use the Eucalyptus Administrator Console.

Special User Attributes

Eucalyptus extends the IAM model by providing the following extra attributes for a user.

- **Registration status:** This is only meaningful for the account administrator (that is, the account level).
- **Enabled status:** Use this attribute to temporarily disable a user.
- **Password expiration date**
- **Custom information:** Add any name-value pair to a user's custom information attribute. This is useful for attaching pure text information, like an address, phone number, or department. This is especially helpful with external LDAP or Active Directory services.

You can retrieve and modify the registration status, enabled status, and password expiration date using the `euare-usergetattributes` and `euare-usermod` commands. You can retrieve and modify custom information using `euare-usergetinfo` and `euare-userupdateinfo` commands. For more information, see the [CLI Reference Guide](#) for details about these commands.

Policy Overview

Eucalyptus uses the same policy language to specify user level permissions as AWS IAM. Policies are written in JSON. Each policy file can contain multiple statements, each specifying a permission. A permission statement specifies whether to allow or deny a list of actions to be performed on a list of resources, under specific conditions.

A permission statement has the following components:

- **Effect:** Begins the decision that applies to all following components. Either: "Allow" or "Deny"
- **Action or NotAction:** Indicates service-specific and case-sensitive commands. For example: "ec2:RunInstances"
- **Resource or NotResource:** Indicates selected resources, each specified as an Amazon resource name (ARN). For example: "arn:aws:s3:::acme_bucket/blob"
- **Condition:** Indicates additional constraints of the permission. For example: "DateGreaterThan"

The following policy example contains a statement that gives a user with full permission. This is the same access as the account administrator:

```
{
  "Version": "2011-04-01",
  "Statement": [ {
    "Sid": "1",
    "Effect": "Allow",
    "Action": "*",
    "Resource": "*"
  } ]
}
```

For more information about policy language, go to the [IAM User Guide](#).

Policy Notes

You can combine IAM policies with account level permissions. For example, the admin of account A can give users in account B permission to launch one of account A's images by changing the image attributes. Then the admin of account B can use IAM policy to designate the users who can actually use the shared images.

You can attach IAM policies to both users and groups. When attached to groups, a policy is equivalent to attaching the same policy to the users within that group. Therefore, a user might have multiple policies attached, both policies attached to the user, and policies attached to the group that the user belongs to.

Do not attach IAM policies (except quota policies, a Eucalyptus extension) to account admins. At this point, doing so won't result in a failure but may have unexpected consequences.

Policy Extensions

Eucalyptus extends the IAM policy in order to meet the needs of enterprise customers.

EC2 Resource

In IAM, you cannot specify EC2 resources in a policy statement except a wildcard, "*". So, it is not possible to constrain a permission on specific EC2 entities. For example, you can't restrict a user to run instances on a specific image or VM type. To solve that, Eucalyptus created the EC2 resource for the policy language. The following example shows the ARN of an EC2 resource.

```
arn:aws:ec2::<account_id>:<resource_type>/<resource_id>
```

Where account id is optional.

Eucalyptus supports the following resource types for EC2:

- image
- securitygroup
- address (either an address or address range: 192.168.7.1-192.168.7.255)
- availabilityzone
- instance
- keypair
- volume
- snapshot
- vmtypes

The following example specifies permission to launch instances with only an m1.small VM type:

```
{
  "Version": "2011-04-01",
```

```

    "Statement": [{
      "Sid": "2",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",

      "Resource": [
        "arn:aws:ec2::vmtype/ml.small",
        "arn:aws:ec2::image/*",
        "arn:aws:ec2::securitygroup/*",
        "arn:aws:ec2::keypair/*",
        "arn:aws:ec2::availabilityzone/*",
        "arn:aws:ec2::instance/*",
      ]
    }]
  }
}

```

Policy Key

Eucalyptus implements the following AWS policy keys:

- `aws:CurrentTime`
- `aws:SourceIp`

Eucalyptus extends the policy keys by adding the following to the lifetime of an instance:

- `ec2:KeepAlive`: specifies the length of time (in seconds) that an instance can run
- `ec2:ExpirationTime`: specifies the expiration time (in seconds) for an instance

The following example restricts an instance running time to 24 hours:

```

{
  "Version": "2011-04-01",
  "Statement": [{
    "Sid": "3",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {
      "NumericEquals": {
        "ec2:KeepAlive": "1440"
      }
    }
  }]
}

```

If there are multiple `ec2:KeepAlive` or `ec2:ExpirationTime` keys that match a request, Eucalyptus chooses the longer lifetime for the instance to run.

Default Permissions

Different identities have different default access permissions. When no policy is associated with them, these identities have the permission listed in the following table.

Identity	Permission
System admin	Access to all resources in the system
Account admin	Access to all account resources, including those shared resources from other accounts like public images and shared snapshots

Identity	Permission
Regular user	No access to any resource

For convenience, Eucalyptus grants the following default access to regular users:

- Users can list themselves (euare-userlistbypath)
- Users can get their own attributes (euare-usergetattributes)
- Users can get information about themselves (euare-usergetinfo)
- Users can list their own accounts (euare-accountlist)

Account administrators have the following default permissions:

- euare-accountlistpolicies
- euare-accountgetpolicy

Quotas

Eucalyptus adds quota enforcement to resource usage. To avoid introducing another configuration language into Eucalyptus, and simplify the management, we extend the IAM policy language to support quotas. The only addition added to the language is the new `limit` effect. If a policy statement's effect is `limit`, it is a quota statement.

A quota statement also has action and resource fields. You can use these fields to match specific requests, for example, quota only being checked on matched requests. The actual quota type and value are specified using special quota keys, and listed in the `condition` part of the statement. Only condition type `NumericLessThanEquals` can be used with quota keys.



Important: An account can only have a quota policy. If you attach an IAM policy to an account (where, for example, the Effect is "Allow" or "Deny"), there will be unexpected results.

The following quota policy statement limits the attached user to only launch a maximum of 16 instances in an account.

```
{
  "Version": "2011-04-01",
  "Statement": [{
    "Sid": "4",
    "Effect": "Limit",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {
      "NumericLessThanEquals": {
        "ec2:quota-vminstancenumber": "16"
      }
    }
  }]
}
```

You can attach quotas to both users and accounts, although some of the quotas only apply to accounts. Quota attached to groups will take no effect.

When a quota policy is attached to an account, it actually is attached to the account administrator user. Since only system administrator can specify account quotas, the account administrator can only inspect quotas but can't change the quotas attached to herself.

The following is all the quota keys implemented in Eucalyptus:

Quota Key	Description	Applies to
s3:quota-bucketnumber	Number of S3 buckets	account and user
s3:quota-bucketobjectnumber	Number of objects in each bucket,	account and user

Quota Key	Description	Applies to
s3:quota-bucketsize	Size of bucket, in MB	account and user
s3:quota-buckettotalsize	total size of all buckets, in MB	account and user
ec2:quota-addressnumber	Number of elastic IPs	account and user
ec2:quota-imagenumber	Number of EC2 images	account and user
ec2:quota-snapshotnumber	Number of EC2 snapshots	account and user
ec2:quota-vminstancenumber	Number of EC2 instances	account and user
ec2:quota-volumenumber	Number of EC2 volumes	account and user
ec2:quota-volumetotalsize	Number of total volume size, in GB	account and user
iam:quota-groupnumber	Number of IAM groups	account
iam:quota-usernumber	Number of IAM users	account

Default Quota

Contrary to IAM policies, by default, there is no quota limits (except the hard system limit) on any resource allocations for a user or an account. Also, system administrators are not constrained by any quota. Account administrators are only be constrained by account quota.

Algorithms

Eucalyptus uses two types of algorithms to determine access.

Policy Evaluation Algorithm

You can associated multiple policies and permission statements with a user. The way these are combined together to control the access to resources in an account is defined by the policy evaluation algorithm. Eucalyptus implements the same policy evaluation algorithm as IAM:

1. If the request user is account admin, access is allowed.
2. Otherwise, collect all the policy statements associated with the request user (attached to the user and all the groups the user belongs to), which match the incoming request (i.e. based on the API being invoked and the resources it is going to access).
 - a. If there is no matched policy statement, access is denied (default implicit deny).
 - b. Otherwise, evaluate each policy statement that matches.
 - If there is a statement that explicitly denies the access, the request is denied.
 - If there is no explicit deny, which means there is at least one explicit allow, access is allowed.

Access Evaluation Algorithm

Now we give the overall access evaluation combining both account level permissions and IAM permissions, which decides whether a request is accepted by Eucalyptus:

1. If the request user is sys admin, access is allowed.
2. Otherwise, check account level permissions, e.g. image launch permission, to see if the request user's account has access to the specific resources.
 - a. If not, the access is denied.
 - b. Otherwise, invoke the policy evaluation algorithm to check if the request user has access to the resources based on IAM policies.

Quota Evaluation Algorithm

Like the normal IAM policies, a user may be associated with multiple quota policies (and multiple quota statements). How all the quota policies are combined to take effect is defined by the quota evaluation algorithm:

1. If the request user is sys admin, there is no limit on resource usage.
2. Otherwise, collect all the quotas associated with the request user, including those attached to the request user's account and those attached to the request user himself/herself (for account admin, we only need collect account quotas).
3. Evaluate each quota one by one. Reject the request as long as there is one quota being exceeded by the request. Otherwise, accept the request.



Note: The hard limits on some resources override quota limits. For example, `walrus.storagemaxbucketsizeinmb` (system property) overrides the `s3:quota-bucketsize` (quota key).

Sample Policies

A few example use cases and associated policies.

Here are some example use cases and associated policies. You can edit these policies for your use, or use them as examples of JSON syntax and form.



Tip: For more information about JSON syntax used with AWS resources, go to [Using AWS Identity and Access Management](#).

Examples: Allowing Specific Actions

The following policy allows a user to only run instances and describe things.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": [ "ec2:*Describe*", "ec2:*Run*" ],
    "Resource": "*" ,
  } ]
}
```

The following policy allows a user to only list things:

```
{
  "Statement": [
    {
      "Sid": "Stmt1313686153864",
      "Action": [
        "iam:List*"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

The following policy grants a generic basic user permission for running instances and describing things.

```
{
  "Statement": [
    {
      "Sid": "Stmt1313605116084",
      "Action": [
```

```

        "ec2:AllocateAddress",
        "ec2:AssociateAddress",
        "ec2:AttachVolume",
        "ec2:Authorize*",
        "ec2:CreateKeyPair",
        "ec2:CreateSecurityGroup",
        "ec2:CreateSnapshot",
        "ec2:CreateVolume",
        "ec2>DeleteKeyPair",
        "ec2>DeleteSecurityGroup",
        "ec2>DeleteSnapshot",
        "ec2>DeleteVolume",
        "ec2:Describe*",
        "ec2:DetachVolume",
        "ec2:DisassociateAddress",
        "ec2:GetConsoleOutput",
        "ec2:RunInstances",
        "ec2:TerminateInstances",
        "ec2:ReleaseAddress"
    ],
    "Effect": "Allow",
    "Resource": "*"
}
]
}

```

Examples: Denying Specific Actions

The following policy allows a user to do anything but delete.

```

{
  "Statement": [
    {
      "Action": [
        "ec2>Delete*"
      ],
      "Effect": "Deny",
      "Resource": "*"
    }
  ]
}

```

The following policy denies a user from creating other users.

```

{
  "Statement": [
    {
      "Sid": "Stmt1313686153864",
      "Action": [
        "iam:CreateUser"
      ],
      "Effect": "Deny",
      "Resource": "*"
    }
  ]
}

```

Examples: Specifying Time Limits

The following policy allows a user to run instances within a specific time.

```
{
  "Statement": [
    {
      "Sid": "Stmt1313453084396",
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "DateLessThanEquals": {
          "aws:CurrentTime": "2011-08-16T00:00:00Z"
        }
      }
    }
  ]
}
```

The following policy blocks users from running instances at a specific time.

```
{
  "Statement": [
    {
      "Sid": "Stmt1313453084396",
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "DateLessThanEquals": {
          "aws:CurrentTime": "2011-08-16T00:00:00Z"
        }
      }
    }
  ]
}
```

The following policy keeps alive an instance for .

```
{
  "Statement": [
    {
      "Action": ["ec2:RunInstances" ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": { "NumericEquals":{"ec2:KeepAlive":"60000"}}
    }
  ]
}
```

The following policy sets an expiration date on running instances.

```
{
  "Statement": [
    {
      "Action": ["ec2:RunInstances" ],
```



```

    "Effect": "Allow",
    "Resource": "*",
    "Condition": { "DateEquals": {"ec2:ExpirationTime": "2011-08-16T00:00:00Z"} }
  }
}

```

Examples: Restricting Resources

The following policy allows users to only launch instances with a large image type.

```

{
  "Statement": [
    {
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2::vmtpe/ml.xlarge"
    }
  ]
}

```

The following policy restricts users from launching instances with a specific image ID.

```

{
  "Statement": [
    {
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:ec2::image/emi-0FFF1874"
    }
  ]
}

```

The following policy restricts users from allocating addresses to a specific elastic IP address.

```

{
  "Statement": [
    {
      "Sid": "Stmt1313626078249",
      "Action": "*",
      "Effect": "Deny",
      "Resource": "arn:aws:ec2::address/192.168.10.140"
    }
  ]
}

```

The following policy denies volume access.

```

{
  "Statement": [
    {
      "Action": [
        "ec2:*"
      ],

```

```

    "Effect": "Deny",
    "Resource": "arn:aws:ec2::volume/*"
  }
]
}

```

LDAP/AD Integration

You can use the Eucalyptus LDAP/Active Directory (AD) integration to synchronize existing LDAP/AD user and group information with Eucalyptus. When you enable LDAP/AD synchronization, Eucalyptus does the following:

- Imports specified user and group information from LDAP or AD and maps them into a predefined two-tier account/group/user structure
- Authenticates Eucalyptus Administrator Console users through the connected LDAP or AD service

Note that Eucalyptus only imports the identities and some related information. Any Eucalyptus-specific attributes are still managed from Eucalyptus. These include:

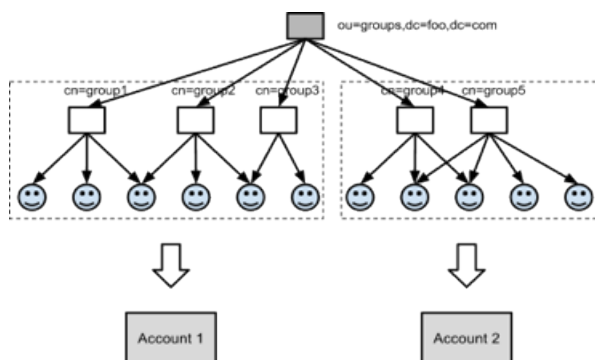
- User credentials: secret access keys and X.509 certificates. The Eucalyptus Administrator Console login password is an exception. Eucalyptus does not download passwords from LDAP/AD and does not save them either. Eucalyptus authenticates Eucalyptus Administrator Console logins directly through LDAP/AD, using LDAP/AD authentication (simple or SASL).
- Policies: IAM policies and quotas. Policies are associated with identities within Eucalyptus, and stored in internal database.

Also note that special identities, including system administrators and account administrators, are created in Eucalyptus and not imported from LDAP/AD. Only normal user identities are imported.

Identity Mapping

Identities in LDAP/AD are organized differently from the identity structure in Eucalyptus. So a transformation is required to map LDAP/AD identities into Eucalyptus.

The following image shows a simple scheme of how the mapping works. In this scheme, the user groups in LDAP tree are partitioned into two sets. Each set is mapped into one separate account. Group 1, 2 and 3 are mapped to Account 1 and Group 4 and 5 are mapped to Account 2. As the result, all users in Group 1, 2 and 3 will be in Account 1, and all users in Group 4 and 5 will be in Account 2.



To summarize the mapping method:

1. Pick user groups from LDAP/AD and combine them into different accounts. There are two ways of doing this:
 - Use something called accounting groups. Account groups are essentially groups of groups. Each accounting group contains multiple user groups in LDAP/AD. Then each accounting group maps to an account in Eucalyptus.
 - Manually partition groups into accounts. Each group partition maps to an account.

2. Once the accounts are defined (by accounting groups or group partitions), all the LDAP/AD user groups will be mapped into Eucalyptus groups within specific accounts; and LDAP/AD users will be mapped into Eucalyptus users. You also have options to filter the groups and users to be imported into Eucalyptus.

Note that each group can be mapped into multiple accounts. But understand that Eucalyptus accounts are separate name spaces. So for groups and users that are mapped into different accounts, their information (name, attributes, etc) will be duplicated in different accounts. And duplicated users will have separate credentials in different accounts. For example, Group 1 may map to both Account 1 and Account 2. Say user A belongs to Group 1. Then Account 1 will have user A and Account 2 will also have user A. User A in Account 1 and user A in Account 2 will have different credentials, policies, etc., but the same user information.



Note: Currently, there is not a way to map individual users into an account. The mapping unit is LDAP user group.

LDAP/AD Integration Configuration

The LDAP/AD Integration Configuration (LIC) is a JSON format file. This file specifies everything Eucalyptus needs to know about how to synchronize with an LDAP or AD service.

You can find a LIC template at `/usr/share/eucalyptus/lic_template`. This template shows all the fields of the LIC, and provides detailed documentation and example values for each field.

To start a LIC file, use the LIC command line tool.

```
/usr/sbin/euca-lictool --password <password> --out example.lic
```

The above command invokes the LIC tool to create a template LIC and fill in the encrypted password for authenticating to LDAP/AD service (i.e. the password of the administrative user for accessing the LDAP/AD during synchronization). The LIC tool's primary functions are to encrypt the LDAP/AD password and to generate the starting LIC template. The usage of the LIC tool shows different ways to invoke the command.

Once you have the LIC template, you can fill in the details by editing the “*.lic” file using your favorite editor as it is a simple text file. As we said above, the LIC file is in JSON format. Each top level entity specifies one aspect of the LDAP/AD synchronization. The following shows one possible example of a LIC file.

```
{
  "ldap-service": {
    "server-url": "ldap://localhost:7733",
    "auth-method": "simple",
    "user-auth-method": "simple",
    "auth-principal": "cn=ldapadmin,dc=foo,dc=com",
    "auth-credentials": "{RSA/ECB/PKCS1Padding}EAXRnvwnKtCZOxSrD/F3ng/yHH3J4jMxNUS
kJJf6oqNMsiUihjUerZ20e5iyXImPgjK1ELAPnppEfJvhCs7woS7jtFsedunsp5DJCNhgmOb2CR/MnH
11V3FNY7bBWoew5A8Wwy6x7YrPMS0j7dJkwM7yfp1Z6AbKOo2688I9uIvJUQwEKS4dOp7RVdA0izlJ
BDPAxiFZ2qa40VjFI/1mggbiWDNlgxiVtZXAEK7x9SRHJytLS8nrNPpIvPuTg3djKiWPVOLZ6vpSgP
cVeliP261qdUfnf3GDKi3jqbPpRRQ6n8yI6aHw0gAtq8/qPyqjkkDP8JsGBgmXMxiCNPogbWg==",
    "use-ssl": "false",
    "ignore-ssl-cert-validation": "false",
    "krb5-conf": "/path/to/krb5.conf",
  },
  "sync": {
    "enable": "true",
    "auto": "true",
    "interval": "900000",
    "clean-deletion": "false",
  }
}
```

```

},
"accounting-groups":{
  "base-dn":"ou=groups,dc=foo,dc=com",
  "id-attribute":"cn",
  "member-attribute":"member",
  "selection":{
    "filter":"objectClass=accountingGroup",
    "select":["cn=accountingToSelect,ou=Groups,dc=foo,dc=com"],
    "not-select":["cn=accountingToIgnore,ou=Groups,dc=foo,dc=com"],
  }
},
"groups":{
  "base-dn":"ou=groups,dc=foo,dc=com",
  "id-attribute":"cn",
  "member-attribute":"member",
  "selection":{
    "filter":"objectClass=groupOfNames",
    "select":["cn=groupToSelect,ou=Groups,dc=foo,dc=com"],
    "not-select":["cn=groupToIgnore,ou=Groups,dc=foo,dc=com"],
  }
},
"users":{
  "base-dn":"ou=people,dc=foo,dc=com",
  "id-attribute":"uid",
  "user-info-attributes":{
    "fullName":"Full Name",
    "email":"Email"
  },
  "selection":{
    "filter":"objectClass=inetOrgPerson",
    "select":["uid=john,ou=People,dc=foo,dc=com",
"uid=jack,ou=People,dc=foo,dc=com"],
    "not-select":["uid=tom,ou=People,dc=foo,dc=com"],
  }
},

```

In the following sections explain each field of LIC in detail.

ldap-service

The `ldap-service` element contains everything related to the LDAP/AD service.

Element	Description
<code>server-url</code>	The LDAP/AD server URL, starting with <code>ldap://</code> or <code>ldaps://</code> .
<code>auth-method</code>	The LDAP/AD authentication method to perform synchronization.
<code>auth-principal</code>	The ID of the administrative user for synchronization.
<code>auth-credentials</code>	The credentials for LDAP/AD authentication, like a password. We recommend that you encrypt this using <code>/usr/sbin/euca-lictool</code> .
<code>user-auth-method</code>	<p>The LDAP/AD authentication method for normal users to perform Eucalyptus Administrator Console login.</p> <ul style="list-style-type: none"> <i>simple</i>: for clear text user/password authentication. <i>DIGEST-MD5</i>: for SASL authentication using MD5

Element	Description
use-ssl	<ul style="list-style-type: none"> • <i>GSSAPI</i>: SASL authentication using Kerberos V5. Specifies whether to use SSL for connecting to LDAP/AD service.
ignore-ssl-cert-validation	Specifies whether to ignore self-signed SSL certs. This is useful when you only have self-signed SSL certs for your LDAP/AD services.
krb5-conf	The file path for krb5.conf, if you use GSSAPI authentication method.

sync

The `sync` element contains elements for controlling synchronization.

Element	Description
enable	States whether the Eucalyptus Administrator Console uses LDAP rather than the Eucalyptus database for log-ins. Set to true to turn on LDAP web logging. Set to false to use the Eucalyptus database for web logging. If set to false, you can ignore all other fields in this section. Default value: false
auto	Set to true to turn on automatic synchronization. Set to false to turn off synchronization.
interval	The length in milliseconds of the automatic synchronization interval.
clean-deletion	Parameter denoting whether to remove identity entities from Eucalyptus when they are deleted from LDAP. Set to true if you want Eucalyptus to remove any identities once their counterparts in LDAP are deleted. Set to false if you want these identities kept without being purged.

accounting-groups

This section uses a special group in LDAP/AD to designate accounts in the Eucalyptus “accounting group.” The accounting group takes normal LDAP/AD groups as members, i.e., they are groups of groups. The accounting group’s name becomes the account name in Eucalyptus. The member groups become Eucalyptus groups in that account. And the users of all those groups become Eucalyptus users within that account and corresponding Eucalyptus groups.



Important: If you use `accounting-groups`, remove the `groups-partition` section. These two sections are mutually exclusive.

Element	Description
base-dn	The base DN of accounting groups in the LDAP/AD tree.
id-attribute	The ID attribute name of the accounting group entry in LDAP/AD tree.
member-attribute	The LDAP/AD attribute name for members of the accounting group.

Element	Description
selection	<p>The accounting groups you want to map to. This contains the following elements:</p> <ul style="list-style-type: none"> <i>filter</i>: The LDAP/AD searching filter used for the LDAP/AD search to get the relevant LDAP/AD entities, e.g. the users to be synchronized. (Example: objectClass=groupOfNames) <i>select</i>: Explicitly gives the full DN of entities to be synchronized, in case they can not be specified by the search filter. (Example: cn=groupToSelect,ou=Groups,dc=foo,dc=com) <i>not-select</i>: Explicitly gives the full DN of entities NOT to be synchronized, in case this can not be specified by the search filter. (Example: cn=groupToIgnore,ou=Groups,dc=foo,dc=com)

groups-partition

Like accounting-groups, groups-partition specifies how to map LDAP/AD groups to Eucalyptus accounts. However, in this section you to manually specify which LDAP/AD groups you want to map to Eucalyptus accounts.



Important: If you use groups-partition, remove the accounting-groups section. These two sections are mutually exclusive.

The Eucalyptus accounts are created by partitioning LDAP/AD groups. Each partition composes an Eucalyptus account. So all the groups within the partition become Eucalyptus groups within that account. All the users of those groups will become Eucalyptus users within that account and the corresponding Eucalyptus groups.

This section requires that you specify one partition at a time, using a list of JSON key-value pairs. For each entry, the key is the account name to be mapped and the value is a list of names of LDAP/AD groups to be mapped into the account. For example:

```

"groups-partition": {
  "salesmarketing": ["sales", "marketing"],
  "devsupport": ["engineering", "support"],
}

```

Here salesmarketing and devsupport are names for the groups partition and are used as the corresponding Eucalyptus account names.



Tip: If you use groups-partition, remove the accounting-groups section. These two sections are mutually exclusive.

groups

This groups element specifies how to map LDAP/AD groups to Eucalyptus groups. It contains the elements listed in the following table. The meanings are similar to those in accounting-groups element.

Element	Description
base-dn	The base DN for searching groups.
id-attribute	The ID attribute name of the LDAP group.
member-attribute	The name of the attribute for group members. Usually, it is member in modern LDAP implementation, which lists full user DN.

Element	Description
selection	<p>The specific LDAP/AD groups you want to map to. This contains the following elements:</p> <ul style="list-style-type: none"> <i>filter</i>: The LDAP/AD searching filter used for the LDAP/AD search to get the relevant LDAP/AD entities, e.g. the users to be synchronized. (Example: objectClass=groupOfNames) <i>select</i>: The LDAP/AD searching filter used for the LDAP/AD search to get the relevant LDAP/AD entities, e.g. the users to be synchronized. (Example: objectClass=groupOfNames) <i>not-select</i>: Explicitly gives the full DN of entities NOT to be synchronized, in case this can not be specified by the search filter. (Example: cn=groupToIgnore,ou=Groups,dc=foo,dc=com)

users

Explicitly gives the full DN of entities NOT to be synchronized, in case this can not be specified by the search filter. (Example: cn=groupToIgnore,ou=Groups,dc=foo,dc=com)

Element	Description
base-dn	The base DN for searching users.
id-attribute	The attribute ID of the LDAP user.
selection	<p>The specific LDAP/AD users you want to map to. This contains the following elements:</p> <ul style="list-style-type: none"> <i>filter</i>: The LDAP/AD searching filter used for the LDAP/AD search to get the relevant LDAP/AD entities, e.g. the users to be synchronized. (Example: objectClass=groupOfNames) <i>select</i>: Explicitly gives the full DN of entities to be synchronized, in case they can not be specified by the search filter. (Example: cn=groupToSelect,ou=Groups,dc=foo,dc=com) <i>not-select</i>: Explicitly gives the full DN of entities NOT to be synchronized, in case this can not be specified by the search filter. (Example: cn=groupToIgnore,ou=Groups,dc=foo,dc=com)

Synchronization Process

The synchronization always starts when the following happens:

- You manually upload a LDAP/AD Integration Configuration (LIC) file. Every new or updated LIC upload triggers a new synchronization.
- If the automatic synchronization is enabled, a synchronization is started when the timer goes off.



Note: Eucalyptus does not allow concurrent synchronization. If you trigger synchronization more than once within a short time period, Eucalyptus only allows the first one.

During a synchronization, everything specified by an LIC in the LDAP/AD tree will be downloaded into Eucalyptus' internal database. Each synchronization is a merging process of the information already in the database and the information from LDAP/AD. There are three cases for each entity: user, group or account:

- If an entity from LDAP/AD is not in Eucalyptus, a new one is created in the database.
- If an entity from LDAP/AD is already in Eucalyptus, the Eucalyptus version is updated. For example, if a user's info attributes are changed, those changes are downloaded and updated.
- If an entity in Eucalyptus is missing from LDAP/AD, it will be removed from the database if the clean-deletion option in LIC is set to true. Otherwise, it will be left in the database.



Important: If clean-deletion is set to true, the removed entities in Eucalyptus will be lost forever, along with all its permissions and credentials. The resources associated with the entity will be left untouched. It is system administrator's job to recycle these resources.

Access Tasks

This section provides details about the tasks you perform using policies and identities. The tasks you can perform are divided up into tasks for users, tasks for groups, and tasks for policies.

The following use cases detail work flows for common processes:

- *Use Case: Creating an Administrator*
- *Use Case: Creating a User*

You can perform the following access-related tasks listed in the following sections:

- Accounts:
 - *Add an Account*
 - *Approve an Account*
 - *Reject an Account*
 - *Rename an Account*
 - *List Accounts*
 - *Delete an Account*
- Groups:
 - *Create a Group*
 - *Add a Group Policy*
 - *Modify a Group*
 - *Add a User to a Group*
 - *Remove a User from a Group*
 - *List Groups*
 - *List Policies for a Group*
 - *Delete a Group*
- Users:
 - *Add a User*
 - *Create a Login Profile*
 - *Modify a User*
 - *List Users*
 - *Delete a User*
- Credentials:
 - *Generating User Credentials*
 - *Retrieving Existing User Credentials*

- [Uploading a Certificate](#)
- [Working with Administrator Credentials](#)

Use Case: Creating an Administrator

This use case details tasks for creating an administrator. These tasks require that you have your account credentials for sending requests to Eucalyptus using the command line interface (CLI) tools.

Create an Admin Group

Eucalyptus recommends using account credentials as little as possible. You can avoid using account credentials by creating a group of users with administrative privileges.

1. Create a group called administrators.

```
euare-groupcreate -g administrators
```

2. Verify that the group was created.

```
euare-grouplistbypath
```

Eucalyptus returns a listing of the groups that have been created, as in the following example.

```
arn:aws:iam::123456789012:group/administrators
```

Add a Policy to the Group

Add a policy to the administrators group that allows its members to perform all actions in Eucalyptus.

Enter the following command to create a policy called `admin-root` that grants all actions on all resources to all users in the administrators group:

```
euare-groupaddpolicy -p admin-root -g administrators -e Allow -a "*" -r "*" -o
```

Create an Administrative User

Create a user for day-to-day administrative work and add that user to the administrators group.

1. Enter the following command to create an administrative user named `alice`:

```
euare-usercreate -u alice
```

2. Add the new administrative user to the administrators group.

```
euare-groupadduser -g administrators -u alice
```

Generate Administrative Credentials

To start running commands as the new administrative user, you must create an access key for that user.

1. Enter the following command to generate an access key for the administrative user:

```
euare-useraddkey -u alice
```

Eucalyptus returns the access key ID and the user's secret key.

2. Open the `~/.eucarc` file and replace your account credentials you just created, as in this example:

```
export EC2_ACCESS_KEY='WOKSEQRNM1LVIR702XVX1 '
export EC2_SECRET_KEY='0SmLCQ8DAZPKoac7oJYcRMfeDUgGbiSVv1ip5WaH'
```

3. Save and close the file.
4. Open the `~/.iamrc` file and replace your account credentials, as in this example:

```
AWSAccessKeyId=WOKSEQRNM1LVIR702XVX1
AWSecretKey=0SmLCQ8DAZPKoac7oJYcRMfeDUgGbiSVv1ip5WaH
```

5. Save and close the file.
6. Switch `euca2ools` over to using the new credentials.

```
source ~/.eucarc
```

Use Case: Creating a User

This use case details tasks needed to create a user with limited access.

Create a Group

We recommend that you apply permissions to groups, not users. In this example, we will create a group for users with limited access.

1. Enter the following command to create a group for users who will be allowed create snapshots of volumes in Eucalyptus.

```
euare-groupcreate -g ebs-backup
```

2. Open an editor and enter the following JSON policy:

```
{
  "Statement": [
    {
      "Action": [
        "ec2:CreateSnapshot"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

3. Save and close the file.
4. Enter the following to add the new policy name `allow-snapshot` and the JSON policy file to the `ebs-backup` group:

```
euare-groupuploadpolicy -g ebs-backup -p allow-snapshot -f allow-snapshot.json
```

Create the User

Create the user for the group with limited access.

Enter the following command to create the user `sam` in the group `ebs-backup` and generate a new key pair for the user:

```
euare-usercreate -u sam -g ebs-backup -k
```

Eucalyptus responds with the access key ID and the secret key, as in the following example:

```
AKIAJ25S6IJ5K53Y5GCA
QLKyICpfjWAvl09pWqWCbuGB9L3T61w7nYYF0571
```

Accounts

Accounts are the primary unit for resource usage accounting. Each account is a separate name space and is identified by its UUID (Universal Unique Identifier).

Tasks performed at the account level can only be done by the users in the **eucalyptus** account.

Add an Account

To add a new account:

Add an Account (CLI)

To add a new account using the CLI:

Enter the following command:

```
euare-accountcreate -a <account_name>
```

Eucalyptus returns the account name and its ID, as in this example:

```
account01 592459037010
```

Add an Account (Eucalyptus Administrator Console)

To add a new account using the Eucalyptus Administrator Console:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click **New account** in the **Accounts** page.
The **Create a new account** popup displays.
3. Enter an account name in the **Account name** field and click **OK**.

The new account displays in the list on the **Accounts** page.

Approve an Account

To approve an account:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to approve.
The account, name, and Registration status are highlighted.
3. Click **Approve**.
The **Approve selected accounts** popup displays.
4. Verify that the displayed account is the one you want, and click **OK**.

The account's registration status displays as **CONFIRMED** on the **Accounts** page.

Reject an Account

To reject an account:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to delete.
The account, name, and Registration status are highlighted.
3. Click **Reject**.
The **Reject selected accounts** popup displays.
4. Verify that the displayed account is the one you want, and click **OK**.

The account no longer displays on the **Accounts** page.

Rename an Account

This section explains steps to perform so that you can rename an account.

Using the CLI

To change an account's name using the CLI:

Enter the following command:

```
euare-accountaliascreate -a <new_name>
```

Using the Eucalyptus Administrator Console

To change an account's name using the Eucalyptus Administrator Console:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to rename.
The account's **Properties** area displays.
3. In the **Name** field, enter the new name of the account.
4. Click **Save**.

The new account name displays in the **Accounts** page.

List Accounts

This section explains steps to perform so that you can list accounts.

Using the CLI

Use the `euare-accountlist` command to list all the accounts in an account or to list all the users with a particular path prefix. The output lists the ARN for each resulting user.

```
euare-userlistbypath -p <path>
```

Using the Eucalyptus Administrator Console

To list accounts using the Eucalyptus Administrator Console:

- Click **Accounts** in the Quick Links section.
- The **Accounts** page displays.

The **Accounts** page displays all accounts in your cloud.

Delete an Account



Tip: If there are resources tied to the account that you delete, the resources remain. We recommend that you delete these resources first.

To delete an account:

Delete an Account (CLI)

To delete an account using the CLI:

Enter the following command:

```
euare-accountdel -a <account_name> -r true
```

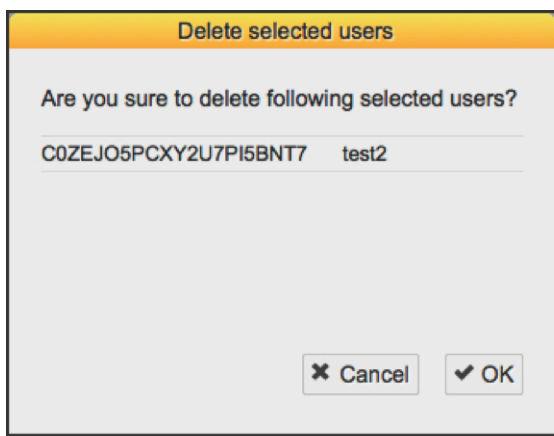
Use the `-r` option set to `true` to delete the account recursively. You don't have to use this option if have already deleted users, keys, and passwords in this account.

Eucalyptus does not return any message.

Delete an Account (Eucalyptus Administrator Console)

To delete an account:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to delete.
The account, name, and Registration status are highlighted.
3. Click **Delete accounts**.
The **Delete selected accounts** popup displays.
4. Verify that the displayed account is the one you want, and click **OK**.



The account no longer displays in the list on the **Accounts** page.

Groups

Groups are used to share resource access authorizations among a set of users within an account. Users can belong to multiple groups.



Important: A group in the context of access is not the same as a security group.

This section details tasks that can be performed on groups.

Create a Group

Using the CLI

To create a group using the CLI:

Enter the following command:

```
euare-groupcreate -g <group_name>
```

Eucalyptus does not return anything.

Using the Eucalyptus Administrator Console

To create a group using the Eucalyptus Administrator Console:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to add a group to.
The account, name, and Registration status are highlighted.
3. Click **New groups** in the **Accounts** page.
The **Create new groups** popup displays.
4. Enter the group name in the **Group name** field.



Tip: You can add more than one group at a time. Every group you add, however, will be in the same path.

5. Enter the group path in the **Group path** field.
6. Click **OK**.

The group is associated with the account you chose. You can see the information if you select the account in the **Accounts** page and click the **Member groups** link, located in the **Properties** section of the screen.

Add a Group Policy

Using the CLI

To add a policy to a group using the CLI:

Enter the following command:

```
euare-groupaddpolicy -g <group_name> -p <policy_name> -e <effect> -a  
  <actions> -o
```

The optional `-o` parameter tells Eucalyptus to return the JSON policy, as in this example:

```
{ "Version": "2008-10-17", "Statement": [ { "Effect": "Allow",  
  "Action": [ "ec2:RunInstances" ], "Resource": [ "*" ] } ] }
```

Using the Eucalyptus Administrator Console

To add a policy to a group using the Eucalyptus Administrator Console:

1. Click **Groups** in the Quick Links section.
The **Groups** page displays.
2. Click the **ID** of the group you want to add a policy to.

The ID, Name, Path, and Owner account line is highlighted.

3. Click **Add policy**.

The **Add new policy** popup displays.

4. Enter the policy name in the **Policy name** field.

5. Enter the policy content in the **Policy content** field.

6. Click **OK**.

The policy is now added to the group.

Modify a Group

Modifying a group is similar to a "move" operation. Whoever wants to modify the group must have permission to do it on both sides of the move. That is, you need permission to remove the group from its current path or name, and put that group in the new path or name.

For example, if a group changes from one area in a company to another, you can change the group's path from `/area_abc/` to `/area_efg/`. You need permission to remove the group from `/area_abc/`. You also need permission to put the group into `/area_efg/`. This means you need permission to call `UpdateGroup` on both `arn:aws:iam::123456789012:group/area_abc/*` and `arn:aws:iam::123456789012:group/area_efg/*`.

Using the CLI

To modify a group using the CLI:

1. Enter the following command to modify the group's name:

```
euare-groupmod -g <group_name> --new-group-name <new_name>
```

Eucalyptus does not return a message.

2. Enter the following command to modify a group's path:

```
euare-groupmod -g <group_name> -p <new_path>
```

Eucalyptus does not return a message.

Using the Eucalyptus Administrator Console

To modify a group using the Eucalyptus Administrator Console:

1. Click **Groups** in the Quick Links section.

The **Groups** page displays.

2. Click the **ID** of the group you want to rename.

The group's **Properties** area displays.

3. In the **Name** field, enter the new name of the group.

4. In the **Path** field, enter the new path for the group.

5. Click **Save**.

The new group name displays in the **Groups** page.

Remove a User from a Group

Using the CLI

To remove a user from a group using the CLI:

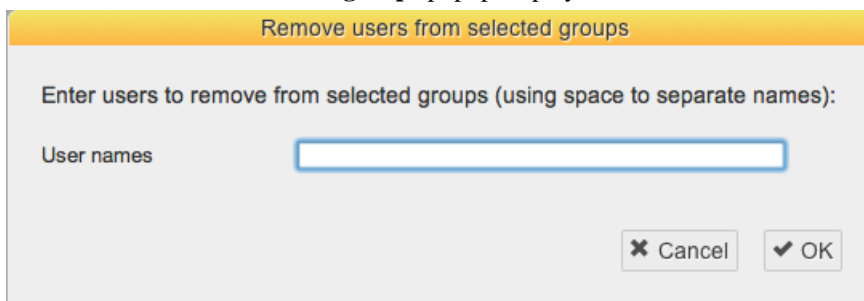
Enter the following command:

```
euare-groupremoveuser -g <group_name> -u <user-name>
```

Using the Eucalyptus Administrator Console

To remove a user from a group using the Eucalyptus Administrator Console:

1. Click **Groups** in the Quick Links section.
The **Groups** page displays.
2. Click the **ID** of the group you want to remove the user from.
The ID, Name, Path, and Owner account line is highlighted.
3. Click **Remove users**.
The **Remove users to selected groups** popup displays.



4. Enter the name of the user you want to remove and click **OK**.

The user is now removed from the group.

List Groups Using the CLI

To list all the groups a specific user is in:

Enter the following command:

```
euare-grouplistbypath
```

Eucalyptus returns a list of paths followed by the ARNs for the groups in each path. For example:

```
arn:aws:iam::eucalyptus:group/groupa
```

Using the Eucalyptus Administrator Console

To list groups using the Eucalyptus Administrator Console:

Click **Groups** in the Quick Links section.
The **Groups** page displays.

The **Groups** page displays all groups in your cloud.

List Policies for a Group

Using the CLI

To list policies associated with a group using the CLI:

Enter the following command:

```
euare-grouplistpolicies -g <group_name>
```

Eucalyptus returns a listing of all policies associated with the group.

Using the Eucalyptus Administrator Console

To list the policies associated with a group using the Eucalyptus Administrator Console:

1. Click **Groups** in the Quick Links section.
The **Groups** page displays.
2. Click the **ID** of the group you want to list policies for.
The **Properties** section displays.
3. In the **Properties** section, click **Policies**.
The **Access Policies** page displays.

Delete a Group

Using the CLI

When you delete a group, you have to remove users from the group and delete any policies from the group. You can do this with one command, using the `euare-groupdel` command with the `-r` option. Or you can follow the following steps to specify who and what you want to delete.

1. Individually remove all users from the group.

```
euare-groupremoveuser -g <group_name> -u <user_name>
```

2. Delete the policies attached to the group.

```
euare-groupdelpolicy -g <group_name> -p <policy_name>
```

3. Delete the group.

```
euare-groupdel -g <group_name>
```

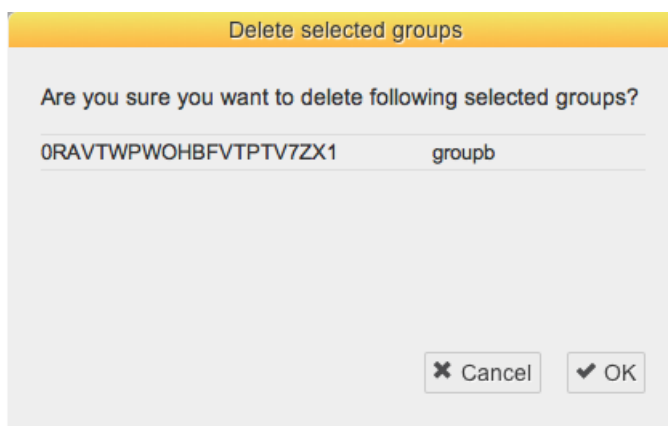
The group is now deleted.

Using the Eucalyptus Administrator Console

To delete a group using the Eucalyptus Administrator Console:

1. Click **Groups** in the Quick Links section.
The **Groups** page displays.
2. Click the **ID** of the group you want to delete.
The ID, Name, Path, and Owner account line is highlighted.
3. Click **Delete groups**.

The **Delete selected groups** popup displays.



4. Click **OK**.

The group is now deleted.

Users

Users are subsets of accounts and are added to accounts by an appropriately credentialed administrator. While the term **user** typically refers to a specific person, in Eucalyptus, a **user** is defined by a specific set of credentials generated to enable access to a given account. Each set of user credentials is valid for accessing only the account for which they were created. Thus a user only has access to one account within a Eucalyptus system. If an individual person wishes to have access to more than one account within a Eucalyptus system, a separate set of credentials must be generated (in effect a new 'user') for each account (though the same username and password can be used for different accounts).

When you need to add a new user to your Eucalyptus cloud, you'll go through the following process:

- | | |
|---|----------------------------------|
| 1 | <i>Create a user</i> |
| 2 | <i>Add user to a group</i> |
| 3 | <i>Give user a login profile</i> |

Add a User

Using the CLI

To add a user using the CLI:

Enter the following command

```
euare-usercreate -u <user_name> -g <group_name> -k
```

Eucalyptus does not return a response.



Tip: If you include the `-v` parameter, Eucalyptus returns a response that includes the user's ARN and GUID.

Using the Eucalyptus Administrator Console

To add a user using the Eucalyptus Administrator Console:

1. Click **Accounts** in the Quick Links section.
The **Accounts** page displays.
2. Click the **ID** of the account you want to rename.
The account's **Properties** area displays.
3. Click **New Users**.
The **Create new users** popup window displays.
4. Enter a name in the **User names** field.



Tip: You can add more than one user at a time. Every user you add, however, will be in the same path.

5. Enter a path in the **User path** field.
6. Click **OK**.

The user is added to the account.

Add a User to a Group

Using the CLI

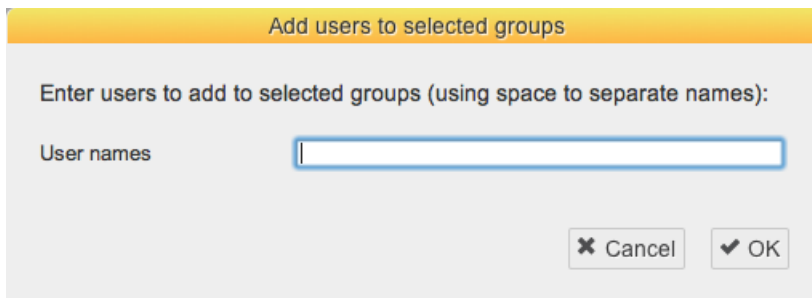
To add a user to a group using the CLI:

Enter the following command:

```
euare-groupadduser -g <group_name> -u <user-name>
```

Using the Eucalyptus Administrator Console

1. Click **Groups** in the Quick Links section.
The **Groups** page displays.
2. Click the **ID** of the group you want to add the user to.
The ID, Name, Path, and Owner account line is highlighted.
3. Click **Add users**.
The **Add users to selected groups** popup displays.



4. Enter the name of the user you want to add and click **OK**.

The user is now added to the group.

Create a Login Profile

Once you create a user, you must generate a password for the user to use the Eucalyptus Administrator Console.

Using the CLI

To create a login profile using the CLI:

Enter the following command:

```
euare-useraddloginprofile -u <user_name> -p <password>
```

Eucalyptus does not return a response.

Using the Eucalyptus Administrator Console

To create a login profile using the Eucalyptus Administrator Console:

1. Click **Users** in the Quick Links section.
The **Users** page displays.
2. Click the **ID** of the user whose path you want to change.
The user's **Properties** area displays.
3. Click **Password**.
The **Change password** popup window displays.
4. Enter the new password in the **New user password** field, and repeat in the **New password again** field.
5. Click **OK**.

The login profile is now complete. If you are generating the password for a different user, let the user know the password and the URL to the Eucalyptus Administrator Console.

Modify a User

Modifying a user is similar to a "move" operation. Whoever wants to modify a user must have permission to do it on both sides of the move. That is, you need permission to remove the user from the current path or name, and put that user in the new path or name.

For example, if a user changes from one team in a company to another, you can change the user's path from `/team_abc/` to `/team_efg/`. You need permission to remove the user from `/team_abc/`. You also need permission to put the user into `/team_efg/`. This means you need permission to call `UpdateUser` on both

```
arn:aws:iam::123456789012:user/team_abc/* and
arn:aws:iam::123456789012:user/team_efg/*.
```

Using the CLI

To rename a user using the CLI:

1. Enter the following command to rename a user:

```
euare-usermod -u <user_name> --new-user-name <new_name>
```

Eucalyptus does not return a message.

2. Enter the following command:

```
euare-groupmod -u <user_name> -p <new_path>
```

Eucalyptus does not return a message.

Using the Eucalyptus Administrator Console

To rename a user using the Eucalyptus Administrator Console:

1. Click **Users** in the Quick Links section.
The **Users** page displays.
2. Click the **ID** of the user you want to rename.
The user's **Properties** area displays.
3. In the **Name** field, enter the new name of the user.
4. Click **Save**.

The new user name displays in the **Users** page.

List Users

You can list users within a path.

Using the CLI

Use the `euare-userlistbypath` command to list all the users in an account or to list all the users with a particular path prefix. The output lists the ARN for each resulting user.

```
euare-userlistbypath -p <path>
```

Using the Eucalyptus Administrator Console

To list users in the same path using the Eucalyptus Administrator Console:

1. Click **Users** in the Quick Links section.
The **Users** page displays.
2. Click the **Path** column to sort all users by path.

Delete a User

Using the CLI

To delete a user using the CLI:

Enter the following command

```
euare-userdel -u <user_name>
```

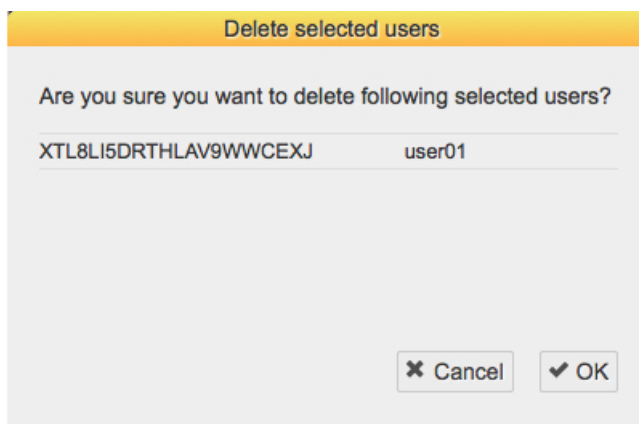
Eucalyptus does not return a response.

Using the Eucalyptus Administrator Console

To delete a user using the Eucalyptus Administrator Console:

1. Click **Users** in the Quick Links section.
The **Users** page displays.
2. Click the **ID** of the user you want to delete.
The user's information is highlighted.
3. Click **Delete Users**.

The **Delete selected users** popup window displays.



4. Click **OK**.

The user is deleted.

Credentials

Eucalyptus uses different types of credentials for both user and administrative functions. Besides the login and password used for accessing the Eucalyptus Administrator Console, Eucalyptus uses an SSH keypair and an X.509 certificate to control access to instances and to Eucalyptus system functions using the command line tools. This section discusses the various types of credentials and how to use them.

- [Working with User Credentials](#)
- [Working with Administrator Credentials](#)

Working with User Credentials

This section describes how to create new user credentials and retrieve existing user credentials.

- [Generating User Credentials](#)
- [Retrieving Existing User Credentials](#)
- [Uploading a Certificate](#)

Generating User Credentials

You can generate new credentials a number of ways. The first time you get credentials using either the Eucalyptus Administrator Console or the `euca_conf` command, a new secret access key is generated. On each subsequent request to get credentials, an existing active secret key is returned. You can also generate new keys using the `euca-useraddkey` command.



Tip: Each request to get a user's credentials either via the download link in the Eucalyptus Administrator Console or using `euca_conf`, a new pair of a private key and X.509 certificate

- To generate a new key for a user by an account administrator, enter the following

```
euca-useraddkey -u <user_name>
```

- To generate a private key and an X.509 certificate pair, enter the following:

```
euare-usercreatecert -u <user_name>
```

Retrieving Existing User Credentials

Eucalyptus provides two main ways of getting user credentials. In both cases, Eucalyptus returns a zip file that contains keys, certificates, a bash script, and several other required files. To use these credentials with such CLI tools as euca2ools or ec2-tools, unzip your credentials zip file to a directory of your choice.

- An administrator with a root access to the machine on which CLC is installed can get credentials using euca_conf CLI tool on that machine.

```
/usr/sbin/euca_conf --cred-account <account> --cred-user <user_name>
--get-credentials <filename>.zip
```

Where <account> and <user_name> are the names of the account and the user whose credentials are retrieved.



Tip: You can omit the `--cred-account` and `--cred-user` options when you get credentials for the **admin** user of the **eucalyptus** account.

- A user can get his or her credentials by logging in into the Eucalyptus Administrator Console and clicking **Download new credentials** in the drop-down menu at the top of the screen. This will result in a download of a zip file.

In the following example we download the credentials zip file to `~/ .euca`, then change access permissions, as shown:

```
mkdir ~/.euca
cd ~/.euca
unzip <filepath>/<creds_zipfile>.zip
chmod 0700 ~/.euca
chmod 0600 *
```



Important: The zip file with credentials contains security-sensitive information. We recommend that you remove or read- and write-protect the file from other users after unzipping.

Alternatively, you can view and copy your access keys and X.509 certificates from the Eucalyptus Administrator Console after logging in, using the Navigation menu.

Uploading a Certificate

To upload a certificate provided by a user:

Enter the following command:

```
euare-useraddcert -u <user_name> -f <cert_file>
```

Working with Administrator Credentials



Important: When you run the following command, you are requesting a new X.509 and a corresponding private key. You cannot retrieve an existing private key.

To generate a set of credentials:

1. Log in to the CLC.

2. Get administrator credentials and source eucarc:

```
/usr/sbin/euca_conf --get-credentials admin.zip
unzip admin.zip
chmod 0600 *
source eucarc
```

Synchronize LDAP/AD

To start an LDAP/AD synchronization:

1. Create an LDAP/AD Integration Configuration (LIC) file to specify all the details about the LDAP/AD synchronization.
2. Upload the LIC file to Eucalyptus using `euca-modify-property`.

Start a LIC File

The LIC is a file in JSON format, specifying everything Eucalyptus needs to know about how to synchronize with an LDAP or AD service. Eucalyptus provides a LIC template at `${EUCALYPTUS}/usr/share/eucalyptus/lic_template`. This template shows all the fields of the LIC, and provides detailed documentation and example values for each field.

To start a LIC file:

Enter the following command:

```
/usr/sbin/euca-lictool --password secret --out example.lic
```

The above command invokes the LIC tool to create a template LIC and fill in the encrypted password for authenticating to LDAP/AD service (i.e. the password of the administrative user for accessing the LDAP/AD during synchronization).

The LIC tool's primary functions are to encrypt the LDAP/AD password and to generate the starting LIC template. The usage of the LIC tool shows different ways to invoke the command.

Once you have the LIC template, you can fill in the details by editing the `*.lic` file using a text editor. Each top level entity specifies one aspect of the LDAP/AD synchronization.

Upload a New LIC File

To upload a new LIC file:

Enter the following:

```
/usr/sbin/euca-modify-property -f
authentication.ldap_integration_configuration=<lic_filename.lic>
```

This triggers a new synchronization using the uploaded LIC file.

Managing Images

A Eucalyptus Machine Image (EMI) is a special type of pre-packaged operating system and application software that Eucalyptus uses to create a virtual machine instance. An EMI contains all necessary information to boot instances of your software. An EMI can contain whatever software you want to meet your needs. For example, you can create an EMI to act as an application server, or a web server, or a custom web service. The following sections contain information to help you create, configure, and modify EMIs.



Important: All users can upload and register images, depending on their access. However, only the admin user can upload and register kernels or ramdisks.

Distro	Version	Arch	Hypervisor
CentOS	5	32	Xen
CentOS	5	32	esx-4.0
CentOS	5	32	esx-4.1
CentOS	5	32	esxi-4.0
CentOS	5	32	esxi-4.1
CentOS	5	64	Xen
CentOS	5	64	esx-4.0
CentOS	5	64	esx-4.1
CentOS	5	64	esxi-4.0
CentOS	5	64	esxi-4.1
RHEL	5	32	Xen
RHEL	5	64	Xen
RHEL	5	64	esx-4.0
RHEL	5	64	esx-4.1
RHEL	5	64	esxi-4.0
RHEL	5	64	esxi-4.1
RHEL	6	64	KVM
RHEL	6	64	esx-4.0
RHEL	6	64	esx-4.1
RHEL	6	64	esxi-4.0
RHEL	6	64	esxi-4.1
Ubuntu	Lucid	32	KVM
Ubuntu	Lucid	64	KVM
Ubuntu	Lucid	64	esx-4.0
Ubuntu	Lucid	64	esx-4.1
Ubuntu	Lucid	64	esxi-4.0
Ubuntu	Lucid	64	esxi-4.1

Image Overview

An image defines what will run on a guest instance with your Eucalyptus cloud. Typically, an image contains one of the Linux distributions like CentOS, Fedora, Ubuntu, Debian or others. It could also contain one of the supported Windows server versions. The format for these is identical.

Normally when we use the term "image" we mean the root file system. Once bundled, uploaded, and registered with Eucalyptus, such an image is known as a Eucalyptus machine image (EMI).

There are, however, other types of images that support the EMI. They are the kernel (EKI) and ramdisk (ERI). They contain kernel modules necessary for proper functioning of the image. Often, one set of these ERIs and EKIs are used by multiple EMIs. Once loaded into the Eucalyptus cloud, the EKI and ERI are referred to by the image and you don't have much interaction with them directly.



Tip: When you run an image, you can override the image's associated kernel and ramdisk if necessary (for example, if you want to try out another kernel. For more information, see [Associate a Kernel and Ramdisk](#).

To help get you started, Eucalyptus provides pre-packaged virtual machines that are ready to run in your cloud. You can get them at the [Eucalyptus Machine Images](#) page. Each Eucalyptus image from this site comes bundled with a corresponding EKI and ERI. You can manually download these images from the web page, or you can use the Eucalyptus Image Store commands to list and describe these images, as well as to install an image in your cloud. For more information see the Eucalyptus Image Store section in the [Command Line Reference Guide](#).

If you find that the pre-packaged images don't meet your needs, you can add an image from an existing, non-registered image, or create your own image.

Once you've selected and downloaded the image(s) you plan to use, read the details in the following section for directions about how to bundle, upload and register the images with your Eucalyptus cloud.

To view the status of your newly created and registered EKI, ERI, and EMI images, use the `euca-describe-images` command. You can also view image status using the Eucalyptus Administrator Console's **Images** page.

Once you have added your image to Eucalyptus, see the [User Guide](#) for information about launching and using instances based on the image.

Image Tasks

This section explains the tasks that you can perform on images in Eucalyptus.

You can perform the following image-related tasks listed in the following sections:

- [Add an Image](#)
- [Browse and Install Images from EuStore](#)
- [Associate a Kernel and Ramdisk](#)
- [Modify an Image](#)
- [Creating an Image](#)
- [Bundle an Image for Amazon EC2](#)
- [Bundle a Windows Instance](#)

Add an Image

An image is the basis for instances that you spin up for your computing needs. If you have access to images that meet your needs, you can skip this section. This section explains how to add an image to your Eucalyptus cloud that is customized for your needs.

There are a few ways you can add an image to Eucalyptus:

- **Add an image based on an existing image.** If you have an image already, read the rest of this section. Eucalyptus has stock images available to help you get started right away. You can find links to these images in the Eucalyptus Administrator Console's start page. You can also get images from the [Eucalyptus Machine Images](#) page.
- Add an image that you modify from an existing image. If you have an image that you want to modify, see [Modify an Image](#).
- Add an image that you create. If you want to create a new image, see [Creating an Image](#).

Each of these three ways has a different first step, but the way you get image to Eucalyptus is the same. To enable an image as an executable entity, you do the following:

1. Bundle a root disk image and kernel/ramdisk pair
2. Upload the bundled image to Walrus bucket storage
3. Register the data with Eucalyptus



Important: Note that while all users can bundle, upload and register images, only the administrator has the required permissions to upload and register kernels and ramdisks.

Once you have an image that meets your needs, perform the tasks listed in this section to add the image to your cloud.

Add a Kernel

When you add a kernel to Walrus, you bundle the kernel file, upload the file to a bucket in Walrus that you name, and then register the kernel with Eucalyptus.

To add a kernel to Walrus:

Use the following three commands:

```
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
```

For example:

```
euca-bundle-image -i
euca-fedora-10-x86_64/xen-kernel/vmlinuz-2.6.27.21-0.1-xen --kernel
true
...
Generating manifest /tmp/vmlinuz-2.6.27.21-0.1-xen.manifest.xml

euca-upload-bundle -b example_kernel_bucket -m
/tmp/vmlinuz-2.6.27.21-0.1-xen.manifest.xml
...
Uploaded image as
example_kernel_bucket/vmlinuz-2.6.27.21-0.1-xen.manifest.xml

euca-register
example_kernel_bucket/vmlinuz-2.6.27.21-0.1-xen.manifest.xml
IMAGE eki-XXXXXXXXX
```

Where the returned value eki-XXXXXXXXX is the unique ID of the registered kernel image.

Add a Ramdisk

When you add a ramdisk to Walrus, you bundle the ramdisk file, upload the file to a bucket in Walrus that you name, and then register the ramdisk with Eucalyptus.

To add a ramdisk to Walrus:

Use the following three commands:

```
euca-bundle-image -i <ramdisk_file> --ramdisk true
euca-upload-bundle -b <ramdisk_bucket> -m /tmp/<ramdisk_file>.manifest.xml
euca-register <ramdisk_bucket>/<ramdisk_file>.manifest.xml
```

For example:

```
euca-bundle-image -i
euca-fedora-10-x86_64/xen-kernel/initrd-2.6.27.21-0.1-xen --ramdisk
true
...
Generating manifest /tmp/initrd-2.6.27.21-0.1-xen.manifest.xml

euca-upload-bundle -b example_rd_bucket -m
/tmp/initrd-2.6.27.21-0.1-xen.manifest.xml
...
Uploaded image as
example_rd_bucket/initrd-2.6.27.21-0.1-xen.manifest.xml

euca-register
example_rd_bucket/initrd-2.6.27.21-0.1-xen.manifest.xml
IMAGE eri-XXXXXXX
```

Where the returned value `eri-XXXXXXX` is the unique ID of the registered ramdisk image.

Add a Root Filesystem

When you add a root filesystem to Walrus, you bundle the root filesystem file, upload the file to a bucket in Walrus that you name, and then register the root filesystem with Eucalyptus. The bundle operation can include a registered ramdisk (ERI ID) and a registered kernel (EKI ID). The resulting image will associate the three images.

You can also bundle the root file system independently and associate the ramdisk and kernel with the resulting EMI at run time. For more information, see [Associate a Kernel and Ramdisk](#).

To add a root filesystem to Walrus:

Use the following three commands:

```
euca-bundle-image -i <root_filesystem_file>
euca-upload-bundle -b <root_filesystem_file_bucket> -m
/tmp/<root_filesystem_file>.manifest.xml
euca-register <root_filesystem_file_bucket>/<root_filesystem_file>.manifest.xml
```

For example:

```
euca-bundle-image -i euca-fedora-10-x86_64/fedora.10.x86-64.img
--ramdisk eri-722B3CBA --kernel eki-5B3D3859
...
Generating manifest /tmp/fedora.10.x86-64.img.manifest.xml

euca-upload-bundle -b example_rf_bucket -m
/tmp/fedora.10.x86-64.img.manifest.xml
...
Generating manifest /tmp/fedora.10.x86-64.img.manifest.xml
```

```
euca-register example_rf_bucket/fedora.10.x86-64.img.manifest.xml
IMAGE emi-XXXXXXX
```

Where the returned value emi-XXXXXXX is the unique ID of the registered machine image.

Browse and Install Images from EuStore

Eucalyptus provides a resource - called EuStore - that contains images that you can download and install. This task explains how to browse and install images from EuStore.

To browse and install an image from EuStore:

1. Find an image on EuStore:

```
eustore-describe-images
```

This command returns a list of images available from the EuStore. For example:

```
0400376721 fedora      x86_64  starter      kvm          Fedora 16
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
2425352071 fedora      x86_64  starter      kvm          Fedora 17
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
1107385945 centos      x86_64  starter      xen, kvm, vmware CentOS 5 1.3GB
root, Hypervisor-Specific Kernels
3868652036 centos      x86_64  starter      kvm          CentOS 6.3
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
1347115203 opensuse     x86_64  starter      kvm          OpenSUSE 12.2
x86_64 - KVM image. SUSE Firewall off. Root disk of 2.5G. Root user enabled.
Working with kexec kernel and ramdisk. OpenSUSE minimal base package set..
.
.
.
```

For additional information regarding the images on eustore (for example, who is the maintainer of the image), use the -v option:

```
# eustore-describe-images -v
0400376721 fedora      x86_64  starter      kvm          Fedora 16
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
20121107181713 d13e-1e35 fedora-based
olivier.renault@eucalyptus.com
2425352071 fedora      x86_64  starter      kvm          Fedora 17
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
20121107181713 6369-6e28 fedora-based
olivier.renault@eucalyptus.com
1107385945 centos      x86_64  starter      xen, kvm, vmware CentOS 5 1.3GB
root, Hypervisor-Specific Kernels
20120517102326 84ae-59db centos-based
images@lists.eucalyptus.com
3868652036 centos      x86_64  starter      kvm          CentOS 6.3
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
20121107181713 48df-52d4 centos-based
olivier.renault@eucalyptus.com
1347115203 opensuse     x86_64  starter      kvm          OpenSUSE 12.2
x86_64 - KVM image. SUSE Firewall off. Root disk of 2.5G. Root user enabled.
Working with kexec kernel and ramdisk. OpenSUSE minimal base package set..
20121120130646 a981-db13 opensuse-based
lester.wade@eucalyptus.com
```

- Pick an available image from the returned list and note the image ID. For this example, we will choose:

```
3868652036 centos      x86_64  starter      kvm          CentOS 6.3
x86_64 - SELinux / iptables disabled. Root disk of 4.5G. Root user enabled.
```

- Install the image from EuStore using the `eustore-install-image` command. For this example, we only need to specify the image ID and the name of a bucket (the bucket will be created if it doesn't already exist):



Note: Some images may require additional parameters (for example, that you specify a kernel type with the `-k` option). Please see the `eustore-install-image` topic in the Eucalyptus Command Line Reference for more information.

```
eustore-install-image -b centos-testbucket -i 3868652036 -k kvm
```

This command performs a number of tasks for you, including downloading the image from the central Eucalyptus image store and installing the image on your own Eucalyptus private cloud. The output from this command will look similar to the following example:

```
Downloading Image : CentOS 6.3 x86_64 - SELinux / iptables disabled. Root
disk of 4.5G. Root user enabled.
0-----1-----2-----3-----4-----5-----6-----7-----8-----9-----10
#####

Checking image bundle
Unbundling image
going to look for kernel dir : kvm-kernel
Bundling/uploading ramdisk
Checking image
Compressing image
Encrypting image
Splitting image...
Part: initrd-2.6.32-279.14.1.el6.x86_64.img.part.00
Generating manifest
/tmp/olEuG_/initrd-2.6.32-279.14.1.el6.x86_64.img.manifest.xml
Checking bucket: centos-testbucket
Creating bucket: centos-testbucket
Uploading manifest file
Uploading part: initrd-2.6.32-279.14.1.el6.x86_64.img.part.00
Uploaded image as
centos-testbucket/initrd-2.6.32-279.14.1.el6.x86_64.img.manifest.xml
centos-testbucket/initrd-2.6.32-279.14.1.el6.x86_64.img.manifest.xml
eri-064B387A
Bundling/uploading kernel
Checking image
Compressing image
Encrypting image
Splitting image...
Part: vmlinuz-2.6.32-279.14.1.el6.x86_64.part.00
Generating manifest /tmp/olEuG_/vmlinuz-2.6.32-279.14.1.el6.x86_64.manifest.xml
Checking bucket: centos-testbucket
Uploading manifest file
Uploading part: vmlinuz-2.6.32-279.14.1.el6.x86_64.part.00
Uploaded image as
centos-testbucket/vmlinuz-2.6.32-279.14.1.el6.x86_64.manifest.xml
centos-testbucket/vmlinuz-2.6.32-279.14.1.el6.x86_64.manifest.xml
eki-A4D6398A
Bundling/uploading image
Checking image
```

```

Compressing image
Encrypting image
Splitting image...
Part: centos-6.3-x86_64.part.00
Part: centos-6.3-x86_64.part.01
Part: centos-6.3-x86_64.part.02
Part: centos-6.3-x86_64.part.03
Part: centos-6.3-x86_64.part.04
Part: centos-6.3-x86_64.part.05
Part: centos-6.3-x86_64.part.06
Part: centos-6.3-x86_64.part.07
Part: centos-6.3-x86_64.part.08
Part: centos-6.3-x86_64.part.09
Part: centos-6.3-x86_64.part.10
Part: centos-6.3-x86_64.part.11
Part: centos-6.3-x86_64.part.12
Part: centos-6.3-x86_64.part.13
Part: centos-6.3-x86_64.part.14
Part: centos-6.3-x86_64.part.15
Part: centos-6.3-x86_64.part.16
Part: centos-6.3-x86_64.part.17
Part: centos-6.3-x86_64.part.18
Part: centos-6.3-x86_64.part.19
Generating manifest /tmp/olEuG_/centos-6.3-x86_64.manifest.xml
Checking bucket: centos-testbucket
Uploading manifest file
Uploading part: centos-6.3-x86_64.part.00
Uploading part: centos-6.3-x86_64.part.01
Uploading part: centos-6.3-x86_64.part.02
Uploading part: centos-6.3-x86_64.part.03
Uploading part: centos-6.3-x86_64.part.04
Uploading part: centos-6.3-x86_64.part.05
Uploading part: centos-6.3-x86_64.part.06
Uploading part: centos-6.3-x86_64.part.07
Uploading part: centos-6.3-x86_64.part.08
Uploading part: centos-6.3-x86_64.part.09
Uploading part: centos-6.3-x86_64.part.10
Uploading part: centos-6.3-x86_64.part.11
Uploading part: centos-6.3-x86_64.part.12
Uploading part: centos-6.3-x86_64.part.13
Uploading part: centos-6.3-x86_64.part.14
Uploading part: centos-6.3-x86_64.part.15
Uploading part: centos-6.3-x86_64.part.16
Uploading part: centos-6.3-x86_64.part.17
Uploading part: centos-6.3-x86_64.part.18
Uploading part: centos-6.3-x86_64.part.19
Uploaded image as centos-testbucket/centos-6.3-x86_64.manifest.xml
centos-testbucket/centos-6.3-x86_64.manifest.xml
Installed image: emi-233637E1

```

Note the last line in the output, which provides the image ID for the image you just installed from the euca store. In this example, the image ID is emi-233637E1.

4. Verify the image was installed on your Eucalyptus cloud. To do this, use the `euca-describe-images` command, which returns a list of the available images on your Eucalyptus cloud:

```
euca-describe-images | grep centos-testbucket
```

This command will return output similar to the following example:

```
IMAGE     eki-A4D6398A
centos-testbucket/vmlinuz-2.6.32-279.14.1.el6.x86_64.manifest.xml
345590850920    available    public      x86_64    kernel
instance-store
IMAGE     eri-064B387A
centos-testbucket/initrd-2.6.32-279.14.1.el6.x86_64.img.manifest.xml
345590850920    available    public      x86_64    ramdisk
instance-store
IMAGE     emi-233637E1    centos-testbucket/centos-6.3-x86_64.manifest.xml
345590850920    available    public      x86_64    machine eki-A4D6398A
eri-064B387A    instance-store
```

Note the ID of the last image in the output -

```
emi-233637E1
```

- matches that of the image we installed from EuStore.

The image has been successfully downloaded from EuStore and installed on your Eucalyptus cloud.

You can now run an instance from this image and connect to it using SSH.

Modify an Image

You might find that existing images in your cloud don't meet your needs.

To modify an existing image:

1. Create a mount point for your image.

```
mkdir temp-mnt
```

2. Associate a loop block device to the image.

```
losetup /dev/loop5 <image_name>
```

where loop5 is a free device

3. Mount the image.

```
mount /dev/loop5 temp-mnt
```

4. Make procfs, dev and sysfs available in your chroot environment.

```
mkdir -p temp-mnt/proc
mkdir -p temp-mnt/sys
mkdir -p temp-mnt/dev
mount -o bind /proc temp-mnt/proc
mount -o bind /sys temp-mnt/sys
mount -o bind /dev temp-mnt/dev
```

5. You now have the image under temp-mnt and you can copy over what you want into it. If you want to install packages into it, you have few options:

- chroot temp-mnt and use apt-get, yum, or zypper to install what you want.

- Instruct the package manager program to use a different root (for example both `dpkg` and `rpm` uses the `--root` option)

6. Unmount the drive.

```
umount /dev/loop5
losetup -d /dev/loop5
```

You now have an image with your modifications. You are ready to [add the image](#) to Eucalyptus.

Creating an Image

You can create your own image if you to deploy an instance that is customized for your specific use. For example, you can create an EMI to deploy a LAMP server, MySQL database server, or a Postgres database server. All you need is some familiarity with the creation process.

About Linux Images

A Linux image is a root partition of a Linux installation. We use the convention followed by EC2 images:

- The first partition is the root partition (where the EMI is attached)
- The second is the ephemeral partition (for additional storage)
- The third is the swap partition

Any image you create must conform to these conditions (for example, `/etc/fstab` must follow this convention). Typically, the disk provided by Eucalyptus (as well as EC2) is the first SCSI disk (`sda`). However, you can customize it to be, for example, the first IDE disk (`hda`) or the first virtual disk on a paravirtual machine (`xvda`).

About Windows Images

Eucalyptus supports a licensed version of Windows images. Before you bundle a Windows image, you must have a valid version of Windows OS installed on your hypervisor and the Eucalyptus Windows Integration Service installed in the created Windows VM. The following sections detail how to perform these tasks.

Eucalyptus is compatible with images created from licensed versions of Windows Server 2003 R2, Windows Server 2008, Windows Server 2008 R2, and Windows 7. Windows OS is sensitive to physical and virtual hardware changes made after installation. We recommend that you create any Windows image on the specific hypervisor (Xen, KVM, or VMware) where you plan to run instances of that Windows image. Eucalyptus does not support running a Windows image across different hypervisors.

Before you begin, you will need the following room for a blank disk file:

- 8GB minimum, for Windows Server 2003 R2 (32-bit and 64-bit)
- 15GB minimum for Windows Server 2008 (32-bit and 64-bit), Windows Server 2008 R2 (64-bit), and Windows 7 (32-bit and 64-bit)



Tip: Eucalyptus implements ephemeral disks by which a running Windows instances are allocated an additional disk space based on the instance type. For example, if you assign an instance type 15 GB of disk and the registered Windows EMI is 8 GB, 7 GB of disk spaces are ephemeral disks accessible under `D:\`. If your Windows application uses scratch space most of time, we recommend keeping the Windows EMI small because it takes longer to launch bigger EMIs.

The windows image is a root file system that has no kernel and ramdisk associated with it. After creating your own Windows images, you can bundle, upload, and register it with Eucalyptus.

Hypervisor and OS Information

The following sections detail how to create an image based on the hypervisor and image OS you wish to use. To create an image using Windows, see [Create Windows Image](#). To create an image using Linux with either KVM or Xen, see [Create Linux Image \(Xen/KVM\)](#).

Create Windows Image

For KVM and Xen hypervisors, we recommend that you perform this task on a node controller (NC), or a host running the same Linux distributions and hypervisors as your NCs. If you are creating the Windows image on a machine currently running as a NC, terminate all running instances and stop the NC. To stop the NC, enter:

```
service eucalyptus-nc stop
```

Template files that closely match those that Eucalyptus generates at VM instantiation time exist for KVM and Xen. These files are located at

`/usr/share/eucalyptus/doc/libvirt-[hypervisor]-windows-example.xml`. We recommend that you review the appropriate file to acquaint yourself with its contents, noting required files, bridges, and resources. For more information about configuring the `libvirt.xml` file, go to the [Domain XML format](#) page in the libvirt documentation.

To create an image from a Windows OS in VMware you will need one network interface and one disk.



Note: If you are using VMware, make sure that the Windows VM uses the LSI Logic Parallel driver as the SCSI controller. For some Windows versions, this is not the default SCSI controller in the VM setting.

Install Base Windows OS

The first task for creating a Windows image is installing a base Windows operating system (OS). To install a base Windows OS using KVM or Xen:

1. Log in to the stopped NC server or a host that runs the same hypervisor as the NCs.
2. Create a blank disk file. Specify your Windows VM image name using the parameter `of`.

```
dd if=/dev/zero of=windows.<image_name>.img bs=1M count=1 seek=16999
```



Important: Your image name must start with the word, `windows` (all lower-case).

3. Create a floppy and secondary blank disk to be attached to the image later, in order to test paravirtualization drivers

```
dd if=/dev/zero of=floppy.img \
bs=1k count=1474
dd if=/dev/zero of=secondary.img \
bs=1M count=1 seek=1000
```

4. Copy all of the `.img` and `.iso` files to the `/var/lib/libvirt/images/` directory.
5. Copy the `libvirt-[hypervisor]-windows-example.xml` file to your working directory and rename it to `libvirt-[hypervisor]-windows.xml`, where `[hypervisor]` is one of `xen` or `kvm`.

```
cp /usr/share/eucalyptus/doc/libvirt-kvm-windows-example.xml
/var/lib/libvirt/images/libvirt-kvm-windows.xml
```

or

```
cp /usr/share/eucalyptus/doc/libvirt-xen-windows-example.xml
/var/lib/libvirt/images/libvirt-xen-windows.xml
```

6. Open the new `libvirt-[hypervisor]-windows.xml` file and provide fully qualified paths to the VM image file and iso. Make sure that the name of the bridge is the same as the one used by the hypervisor on which you are creating the Windows image.

Your file should look similar to one of the following examples.

For Xen:

```
<domain type='xen'>
  <name>eucalyptus-windows</name>
  <os>
    <type>hvm</type>
    <loader>/usr/lib/xen/boot/hvmlloader</loader>
    <boot dev='cdrom' />
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <clock offset='localtime' />
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <memory>524288</memory>
  <vcpu>1</vcpu>
  <devices>
    <emulator>/usr/lib64/xen/bin/qemu-dm</emulator>
    <disk type='file'>
      <source file='/root/windows_2003.img' />
      <target dev='hda' bus='ide' />
    </disk>
    <!--<disk type='file' device='disk'>
      <driver name='tap' type='aio' />
      <source file='fully_qualified_path_to_secondary_disk' />
      <target dev='xvda' bus='xen' />
    </disk>
    <disk type='file' device='floppy'>
      <source file='fully_qualified_path_to_floppy_disk' />
      <target dev='fda' />
    </disk> -->
    <disk type='file' device='cdrom'>
      <source
file='/root/en_win_srv_2003_r2_enterprise_with_sp2_cd1_x13-05460.iso' />
      <target dev='hdc' />
      <readonly />
    </disk>
    <interface type='bridge'>
      <source bridge='xenbr0' />
      <script path='/etc/xen/scripts/vif-bridge' />
    </interface>
    <serial type='pty'>
      <source path='/dev/pts/3' />
      <target port='0' />
    </serial>
    <input type='tablet' bus='usb' />
    <input type='mouse' bus='ps2' />
```

```

        <graphics type='vnc' port='-1' autoport='yes' keymap='en-us'
listen='0.0.0.0' />
    </devices>
</domain>

```

For KVM:

```

<domain type='kvm'>
  <name>eucalyptus-windows</name>
  <os>
    <type>hvm</type>
    <boot dev='cdrom' />
  </os>
  <features>
    <acpi />
  </features>
  <memory>524288</memory>
  <vcpu>1</vcpu>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file'>
      <source file='/var/lib/libvirt/images/windows_2003.img' />
      <target dev='hda' />
    </disk>
    <!-- <disk type='file' device='disk'>
      <source file='fully_qualified_path_to_secondary_disk' />
      <target dev='vda' bus='virtio' />
    </disk>
    <disk type='file' device='floppy'>
      <source file='fully_qualified_path_to_floppy_disk' />
      <target dev='fda' />
    </disk> -->
    <disk type='file' device='cdrom'>
      <source
file='/var/lib/libvirt/images/en_win_srv_2003_r2_enterprise_with_sp2_cd1_xl3-05460.iso' />
      <target dev='hdc' />
      <readonly />
    </disk>
    <interface type='bridge'>
      <source bridge='br0' />
      <model type='rtl8139' />
    </interface>
    <!--<interface type='bridge'>
      <source bridge='br0' />
      <model type='virtio' />
    </interface> -->
    <graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0' />
  </devices>
</domain>

```

7. Start the VM.

```

cd /usr/share/eucalyptus/doc/
virsh create libvirt-[hypervisor]-windows.xml

```

8. Connect to the virtual console using the VNC client of your choice. On the NC, check the display number that has been allocated by looking at the process table (`ps axw | grep vnc`). For example, if the display number is 0, then connect to the NC using the VNC client:

```
vinagre <machine-hosting-vm>:0
```

9. Follow the standard Windows installation procedure until the VM has completed installing Windows.



Tip: On some hosts, the VNC's display number will change when an image restarts. Use `ps` to find the current number.

10. Run `virsh list` to display the domain name.

11. Shut down the Windows VM you have just created. The easiest way to shutdown your VM is to use the `virsh destroy` command, as shown:

```
virsh destroy <domain_name>
```

To install the base Windows operating system using VMware, create a new VM using the VMware vSphere Client. Install Windows on the VM following standard VMware procedures, and install VMware Tools.

Install Eucalyptus Windows Integration

To install the Eucalyptus Windows Integration Service:

1. Download the most recent version of the Windows Image Preparation Tool from <http://downloads.eucalyptus.com/software/tools/windows-prep/> to the working directory of your NC or the host running the vSphere client.
2. For KVM and Xen, open the `libvirt-[hypervisor]-windows-example.xml` file you used in the previous section and make the following edits:
 - Comment out the lines of XML code directing the hypervisor to boot the Windows image from the CDROM.
 - Change the text so that `windows-prep-tools-3.1.0.iso` replaces the Windows `.iso` image and is mounted as `cdrom`.
 - Enter the path to the secondary disk file you created in the previous task.
 - Uncomment the lines that direct attachment of a floppy disk, secondary disk, and secondary network interface.



Tip: If you plan on using virtio networking for instances (via `USE_VIRTIO_NET` option on node controllers), uncommenting the virtio interface in the xml is mandatory

Your finished file should look similar to one of the following examples.

For Xen:

```
<domain type='xen'>
  <name>eucalyptus-windows</name>
  <os>
    <type>hvm</type>
    <loader>/usr/lib/xen/boot/hvmlloader</loader>
    <!-- <boot dev='cdrom' /> -->
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <clock offset='localtime' />
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
```

```

<on_crash>destroy</on_crash>
  <memory>524288</memory>
  <vcpu>1</vcpu>
  <devices>
    <emulator>/usr/lib64/xen/bin/qemu-dm</emulator>
    <disk type='file'>
      <source file='/root/windows_2003.img' />
      <target dev='hda' bus='ide' />
    </disk>
    <disk type='file' device='disk'>
      <driver name='tap' type='aio' />
      <source file='/root/secondary.img' />
      <target dev='xvda' bus='xen' />
    </disk>
    <disk type='file' device='floppy'>
      <source file='/root/floppy.img' />
      <target dev='fda' />
    </disk>
    <disk type='file' device='cdrom'>
      <source file='windows-prep-tools-3.1.0.iso' />
      <target dev='hdc' />
      <readonly />
    </disk>
    <interface type='bridge'>
      <source bridge='xenbr0' />
      <script path='/etc/xen/scripts/vif-bridge' />
    </interface>
    <serial type='pty'>
      <source path='/dev/pts/3' />
      <target port='0' />
    </serial>
    <input type='tablet' bus='usb' />
    <input type='mouse' bus='ps2' />
    <graphics type='vnc' port='-1' autoport='yes' keymap='en-us'
listen='0.0.0.0' />
  </devices>
</domain>

```

For KVM:

```

<domain type="kvm">
  <name>eucalyptus-windows</name>
  <os>
    <type>hvm</type>
    <!-- <boot dev='cdrom' /> -->
  </os>
  <features>
    <acpi />
  </features>
  <memory>524288</memory>
  <vcpu>1</vcpu>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file'>
      <source file='/root/windows_2003.img' />
      <target dev='hda' />
    </disk>
    <disk type='file' device='disk'>
      <source file='/root/secondary.img' />
      <target dev='vda' bus='virtio' />
    </disk>
    <disk type='file' device='floppy'>

```

```

        <source file='/root/floppy.img' />
        <target dev='fda' />
    </disk>
    <disk type='file' device='cdrom'>
        <source file='windows-preps-tools-3.1.0.iso' />
        <target dev='hdc' />
        <readonly />
    </disk>
    <interface type='bridge'>
        <source bridge='br0' />
        <model type='rtl8139' />
    </interface>
    <interface type='bridge'>
        <source bridge='br0' />
        <model type='virtio' />
    </interface>
    <graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0' />
</devices>
</domain>

```

3. There is an issue with Xen paravirtualization drivers on some Windows versions. 64-bit Windows Server 2008, Windows Server 2008R2, and Windows 7 have driver signing requirements that prevent unsigned drivers from being installed. The supplied drivers in the package are not signed. To resolve this issue:

- a) Log in to the Windows VM.
- b) Launch `cmd.exe` as administrator, and execute the following commands:

```

bcdedit.exe -set loadoptions DISABLE_INTEGRITY_CHECKS
bcdedit.exe -set TESTSIGNING ON

```

- c) Reboot the VM.

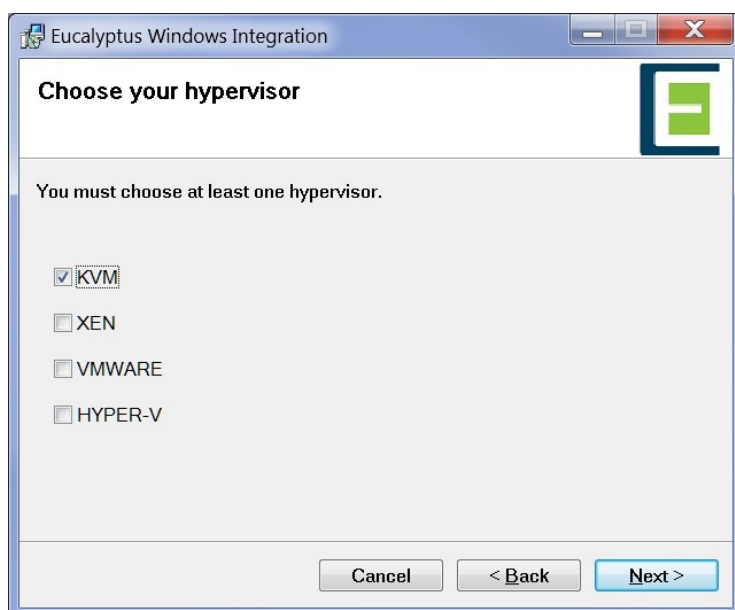
4. Start the VM with the newly modified libvirt xml file:

```

virsh create libvirt-[hypervisor]-windows.xml

```

5. For VMware, use the VMware vSphere client to upload the ISO file to the VSphere datastore. Attach the `windows-prep-tools-3.1.0.iso` to the Windows VM.
6. Log in to Windows and find the Eucalyptus installation files in the CDROM drive.
 - For Windows Server 2003 R2, run `setup.exe`. This automatically installs the .NET framework 2.0, which is not bundled in Server 2003 R2.
 - For all other versions, run `EucalyptusWindowsIntegration.msi`. (`setup.exe` will automatically install .NET framework 2.0, which is not bundled in Server 2003 R2).
7. In the **Choose your hypervisor** step, select your hypervisor and then click **Next**.



Click **Next** and continue until the end of installation.

8. Reboot the Windows VM
9. For KVM and Xen, open the Windows device manager and check that the following drivers are found for each device.

For KVM:

- Floppy disk drive
- Disk drivers: Red Hat VirtIO SCSI Disk Device
- SCSI and RAID controllers: Red Hat VirtIO SCSI controller
- Network adapters: Red Hat VirtIO Ethernet Adapter

For Xen:

- Floppy disk drive
- Disk drivers: XEN PV SCSI Disk Device
- SCSI and RAID controllers: Xen Block Device Driver
- Network adapters: Xen Net Device Driver

If the correct drivers are not found, question marks display on the devices. To install the devices, do the following:

- Right-click on the devices in question and select **Update Drivers** to open the New Hardware Wizard.
- When the new hardware wizard asks if Windows update is to be connected, click **No, not this time**.
- Choose **Install software automatically (recommended)**.
- If a confirmation popup message displays, click **Continue**.

You are now ready to [Configure Active Directory](#).

Configure Active Directory

The Eucalyptus Integration service lets an enterprise with existing Active Directory domains attach Windows instances to the domains and control access to these instances using the existing AD user database. Users can log into the instance either using their domain credentials or the Administrator's password generated with the `euca-get-password` command.

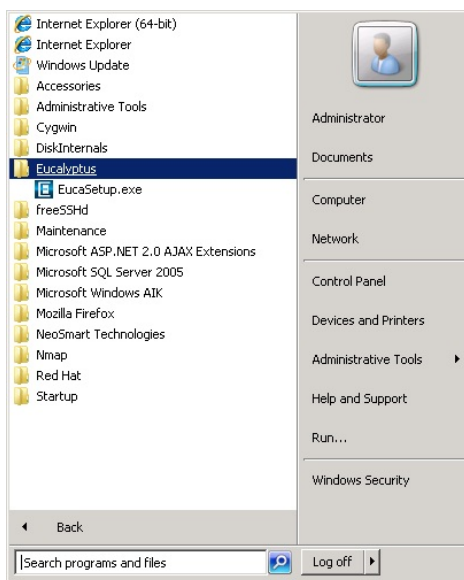
Because AD technology is tightly integrated with domain name service (DNS), the default name server contacted by the instance must be able to resolve the AD address as a proper domain controller. You can do this for all networking modes except System, by configuring the following line the CC's `eucalyptus.conf` file:

```
VNET_DNS=<domain_controller_IP_address>
```

If there is no such pre-existing DNS set-up or your networking mode is System, you might need to change the VM's network interface so that the preferred DNS server points to the domain controller.

To set up Active Directory:

1. Click **Windows Programs > Eucalyptus > Eucalyptus Setup**.



The **Eucalyptus Windows Integration** popup displays.

2. Click the **Active Directory** tab in the Eucalyptus Windows Integration window and enter the following information:

Eucalyptus Windows Integration

General ActiveDirectory RemoteDesktop

AD Address:

Admin Username:

Admin Password: Confirm:

Organizational Unit (optional):

Status: not a member of a domain

Buttons: Apply, Close, Clear

- Enter the address of the existing Active Directory domain controller in the **AD Address** field.
- Enter the administrator username in the **Admin Username** field. We recommend using a generic user account that has permission to join a computer to a domain or a specific organizational unit.
- Enter and confirm the password in the **Admin Password** field. Note that the Admin username and password are required to join an instance to an Active Directory. When launched in Eucalyptus, these properties will be deleted as soon as the instance joins (or fails to join) the domain.

- Optionally, enter an organizational unit in the **Organizational Unit** field. This specifies a container that the instances launched from this image will be attach to.



Tip: If the values entered in this section are incorrect, the launched instances will fail to join the domain. We recommend that you verify the information by manually joining a computer to a domain using the same information that you entered in this step. You may first log in the launched instance using the administrator password (`euca-get-password`) and manually join the domain for verification.

3. Click **Apply**.

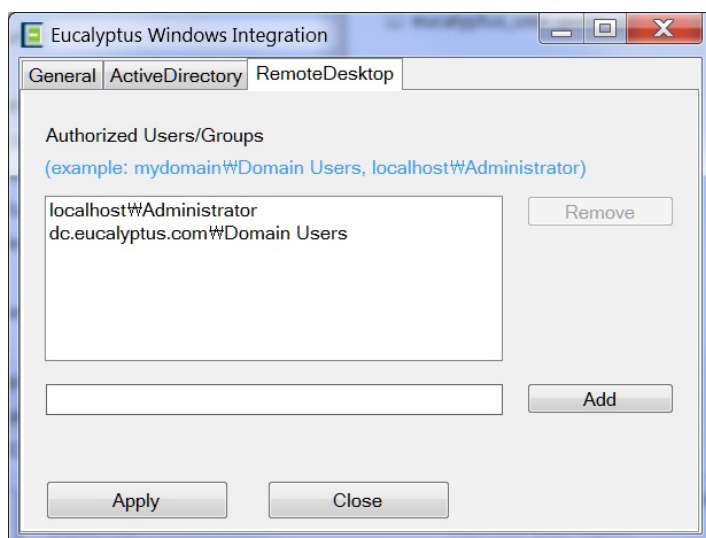
You are now ready to [Configure Remote Desktop](#).

Configure Remote Desktop

Domain users or groups require remote desktop permission to log into an instance. By default, only the local administrator has the remote desktop permission. The Eucalyptus Integration Service provides a way to grant remote desktop permission to additional domain users or groups.

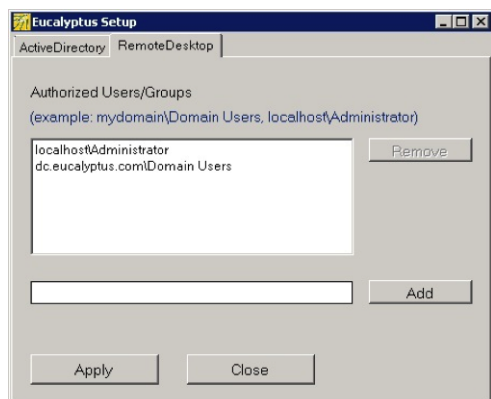
To configure remote desktop permission:

- Open the **Eucalyptus Windows Integration** popup (**Windows Programs > Eucalyptus > Eucalyptus Setup**).
- Click the **RemoteDesktop** tab



The names of authorized domain users and groups display in the **Authorized User/Groups** field. By default, only the local administrator is listed as authorized.

- In the text field below the list, enter a user and group account name in the format `[DOMAIN]\[USER or GROUP]`. If you add a new local user or local group, prepend the account name `localhost\` instead of the domain name.



4. Click **Add**.
5. Repeat for all user/groups that you want to add.
6. Click **Apply**.

When the instance launches, the members of the groups you added can log in to the instance through remote desktop.

You are now ready to [Run Sysprep](#).

Run Sysprep

Sysprep is a Microsoft tool for deploying multiple Windows operating systems in an enterprise. Running Sysprep removes system-specific information such as security ID (SID) from the Windows OS before you clone an image. Sysprep then re-initializes the OS after the image is cloned and started on multiple computers. Use Sysprep to prepare images when you use Microsoft Key Management Service to activate license keys. Also, use Sysprep when your Windows systems are attached to Active Directory.

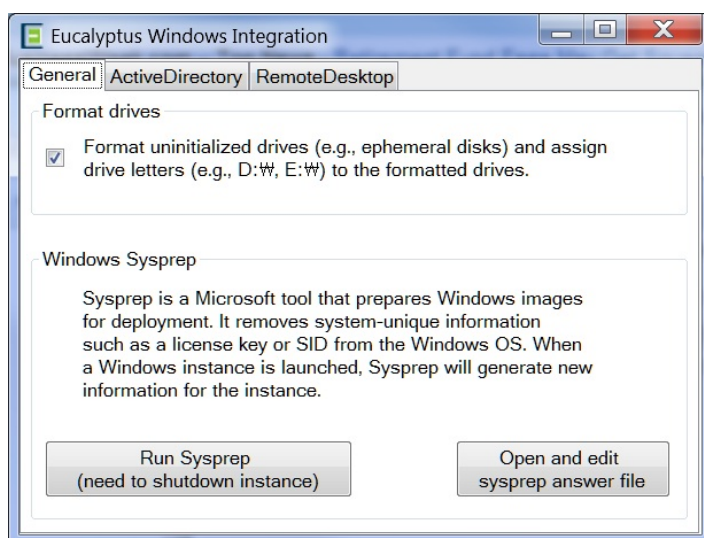
In Eucalyptus, you can run Sysprep before you bundle images with the `euca-bundle-image` or `euca-bundle-instance` commands.



Note: The Eucalyptus Integration Service supports Sysprep for Windows Server 2008, Windows Server 2008 R2, and Windows 7.

To configure and run Sysprep:

1. Open the **Eucalyptus Windows Integration** popup (**Windows Programs** > **Eucalyptus** > **Eucalyptus Setup**).



2. Click the **General** tab.
3. Ensure that **Format uninitialized drives** is checked
4. If you want to edit the Sysprep answer file, click the **Open and change** button in the General tab. Otherwise skip this step.
5. Click the **Run Sysprep** button.
Sysprep starts.
6. After Sysprep is complete, close the application and shutdown the Windows VM using the Windows Programs menu.

You are now ready to [Add Image to Eucalyptus](#).

Convert the VMDK to an Image

After your Windows VM image is ready on ESX/ESX-i/VCenter, use the `euca_imager` command to download the image and convert it to the format Eucalyptus uses. The `euca_imager` command converts a remote VMDK located on a datastore into a disk image on the local disk on the CC/VMware Broker machine.

1. Log into the CC/VMware Broker machine and, if necessary, set the appropriate environment variables.

```
# export VDDK_HOME=/opt/packages/vddk/
# export EUCALYPTUS=/
# export LD_LIBRARY_PATH=/opt/packages/vddk/lib/vmware-vix-disklib/lib64/
# PATH=/usr/lib/eucalyptus/:$PATH
# cd ~/.euca (or path to .eucarc)
# source .eucarc
```

2. Enter the euca_imager command with appropriate parameters to extract and convert the VMDK to a raw disk image. Parameters are described in the following table:

Parameter	Description
debug	Enables euca_imager debugging output
in	https URL for the vCenter Server or ESX/ESXi machine
out	User-defined name of disk
login	Name used to access the VMware machine
password	Password used to access the VMware machine
vsphere-vmdk	Go to vSphere client, find the virtual machine that you have installed (make sure it is in powered off state), and select the Hard disk 1 resource. There will be a Disk File field for the VMDK, and a similar field on the Option tab for the VMX file.
vsphere-vmx	Same description as vsphere-vmdk parameter. This is optional if you're contacting an ESX node directly, but may be required when contacting vCenter.
vsphere-datacenter	Specified name of the datacenter. This is optional if you're contacting an ESX node directly, but may be required when contacting vCenter.

The following example shows euca_imager with the appropriate parameters as described above converting a VMDK to a raw disk image. Note that the prefix of the out parameter must begin with the word windows.

```
/usr/lib/eucalyptus/euca_imager debug=yes convert in=
https://192.168.7.198 out=windows.disk login=Administrator
password=password
vsphere-vmdk="[datastore1]WindowsVM/WindowsVM.vmdk"
vsphere-datacenter=DataCenter1
vsphere-vmx="[datastore1]WindowsVM/WindowsVM.vmx"
```

You are now ready to [Add Image to Eucalyptus](#).

Add Image to Eucalyptus

To enable an image as an executable entity, you must bundle and upload the Windows disk image to Walrus, and then register the uploaded image with Eucalyptus.

Run the following command to bundle, upload, and register your Windows disk image:

```
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
```

Your Windows image is now ready to run as an instance.

After you register the image, Walrus decrypts the image bundle. This process might take a few minutes for a large Windows image to be decrypted. For example, a 10G image requires that you wait about 10 minutes before you launch the instance.

Create Linux Image (Xen/KVM)



Important: Before you begin, make sure that your hypervisor is installed and properly functioning for the account you are going to use.

To create a root filesystem to be used with Eucalyptus using Xen or KVM and an ISO image of the guest OS you want to install:

1. Download the ISO image of the distro (guest OS) you want to install.
2. Create a virtual disk with the desired size. For example, to create a 4 GiB disk:

```
dd if=/dev/zero of=<image_name> bs=1M count=4096
```

3. If you are using Xen, complete the following steps:

- a) Create a configuration file (for example, `xen.cfg`) to boot off the ISO image. The file should look like the following:

```
name = "bootFromISO"
kernel = "/usr/lib/xen/boot/hvmloader"
memory = 1024
builder = "hvm"
device_model = "/usr/lib64/xen/bin/qemu-dm"
boot = "d"
disk = [ 'file:PATH_TO_ISO,hdc:cdrom,r',
'file:PATH_TO/<image_name>,sda,w' ]
vif = [ 'mac=00:01:01:00:00:03, bridge=xenbr0' ]
dhcp="on"
vnc = 1
vncdisplay = 7
pae = 1
```



Important: You must modify `device_model` and `kernel` since they depend on where the distribution puts such files. Look into the Xen package file list. Also be sure that all the options are correct for your needs.

- b) Start the domU.

```
xen create xen.cfg
```

- c) Connect with a VNC viewer. For example:

```
xvncviewer localhost:7
```

- If you are using KVM, start KVM using the CDROM as the boot device and new image as the disk.

```
/usr/libexec/qemu-kvm -cdrom iso_image -drive
if=scsi,file=<image_name>,boot=off
```



Note: You need to have a SCSI as a bus because Eucalyptus expects the disk to be in `/dev/sda`.

- Install the distro as you would normally would. Do not use `lvm` or `md`. Install everything on one partition. Remember the partition on which the distro is installed.



Important: Most distributions use the MAC address to associate each network interface with an interface name. To ensure that each instance started from the image sees the appropriate network interfaces as `eth0` etc, you must remove this association. Refer to your distribution's documentation for information on how to do this. On Debian and Ubuntu, `/etc/udev/rules.d/*net*` may be relevant. On CentOS and RHEL, `/etc/network-scripts/ifcfg-eth0` may be relevant. Failure to remove this association may leave your instance unable to connect to the network.

- If you are using Xen, stop the domU.
- Run `parted` to find out the starting block and the block size of the root file system, in bytes.

```
parted <image_name>
```

- Change the units shown by `parted` to blocks.

```
(parted) u
Unit? [compact]? b
```

- Print the current partition table using `p`.

Note the start block and block size for the partition to which you will extract the file system. This example uses a start block of 32256 and a block size of 1024000.

- Extract the file system.

```
dd if=<image_name> of=rootfs.img bs=1 skip=32256 count=1024000
```



Tip: If this process is running slowly, you can increase the block size. Set `bs` to a power of two, and divide `skip` and `count` by the same number. For example, using a block size of 512, the above becomes `dd if=<image_name> of=rootfs.img bs=512 skip=63 count=2000`.

You now have a root filesystem. Make sure that it is compatible with the kernel/initrd you are using in your cloud environment. In particular you may want to be sure you have the modules of the kernel you are going to use. Then you are ready to [add the image](#) to Eucalyptus.

Associate a Kernel and Ramdisk

There are three ways that you can associate a kernel and ramdisk with an image.



Tip: This only applies to Linux images. Windows images do not use kernels or ramdisks.

- You can associate a specific kernel and ramdisk identifier with an image at the `euca-bundle-image` step.

```
euca-bundle-image -i <vm_image_file> --kernel <eki-XXXXXXX> --ramdisk
<eri-XXXXXXX>
```

- You can specify kernel and ramdisk at instance run time as an option to `euca-run-instances` command.

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

- An administrator can set default registered kernel and ramdisk identifiers that will be used if a kernel and ramdisk are unspecified by either of the other options.

Bundle an Image for Amazon EC2

EMIs can be uploaded unchanged to Amazon EC2 and run as AMIs in the public cloud. To upload an EMI image file to Amazon, do the following:

1. Locate the Amazon ec2 cert file that is provided as part of the EC2 AMI tools. This file is generally located in `$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2.pem`.
2. Enter the following command:

```
euca-bundle-image -i <image_name> -r <architecture>\
-c <cert_filename> -k <private_key_filename> \
--ec2cert <path_to_cert_file>
```

For example:

```
euca-bundle-image -i euca-centos-5.3 -r x86_64\
x86_64/centos.5-3.x86-64.img -u 123456789111 -c cert- \
abc.pem -k pk-abc.pem --ec2cert \
$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2.pem
```

Bundle a Windows Instance

The `euca-bundle-instance` command lets you bundle a new Windows image from a running Windows instance directly to Walrus storage. Using `euca-bundle-instance` is an efficient way to generate modified Windows VMs. You can spin up a Windows VM instance from an existing Windows VM image, modify it as needed, then save the modified image to Walrus storage, where it is immediately available for registering and running with the modifications already in place.

To bundle a Windows instance:

Enter the following command:

```
euca-bundle-instance -b <bucket_name> -p
<prefix_starting_with_windows> -o $EC2_ACCESS_KEY -w
$EC2_SECRET_KEY <instance_ID>
```



Tip: You can query the progress of a bundling task using the `euca-describe-bundle-task` command. You can cancel the active bundle task using `euca-cancel-bundle-task` command.

The following example bundles and registers a new Windows image from an existing Windows instance with ID `i-12c4af6a`, to the bucket `mybucket`, with image prefix `windows2003_bundle`:

```
euca-bundle-instance -b mybucket -p windows2003_bundle -o  
$EC2_ACCESS_KEY -w $EC2_SECRET_KEY i-12c4af6a  
  
euca-register mybucket/windows2003_bundle.manifest.xml
```

Managing Reporting

Eucalyptus provides two ways for getting metrics for your cloud: you can get a report directly from the Cloud Controller (CLC), or you can get a report from data exported from the CLC and imported to a data warehouse.

When you install Eucalyptus, you automatically get the reporting system in place to generate reports from the CLC. However, the down side to using the CLC for reports is latency. Because of this, Eucalyptus also supports a data warehouse that resides outside the Eucalyptus system to store report data.

This section describes the concepts and best practices for Eucalyptus reporting, and how to generate reports.

Reporting Overview

Eucalyptus lets you generate reports to monitor cloud resource use. Each type of report is for a specified time range. Eucalyptus supports the following report types:

- **Instance:** The instance report provides information about the amount, duration, and utilization of all running instances. Use this report to understand how many instances each user is running, whether your instance types are large enough, etc.
- **S3:** The S3 report provides information about the number of buckets and objects stored in Walrus. Empty buckets are not reported. Use this report to understand the storage needs of each user and your cloud's storage needs.
- **Volume:** The volume report provides information about the amount, duration, and size of all volumes in use. Use this report to understand how many volumes are running, and what the storage size of each volume is.
- **Snapshot:** The snapshot report provides information about the amount of your cloud's snapshots. Use this report to understand how many snapshots there are and from which volumes, and what the size of each snapshot is.
- **Elastic IP:** The elastic IP report provides information about the lifecycle of elastic IPs in your cloud, including which user is using which IPs, which IPs are currently in use, and how often and for how long does IP get allocated. Use this report to understand how many IPs each user is assigned and to which instance the IP is assigned to, and the running time of each IP.
- **Capacity:** The capacity report provides overall information about your cloud's resources, including instance types and storage. Use this report to determine if your resources are being used adequately, and whether you need to scale up or down.

You can generate reports in either CSV or HTML formats for use with external tools.

If you want to use the CLC for your reports, see [Reporting Tasks: CLC](#).

If you want to use the data warehouse for your reports, see [Set Up the Data Warehouse](#).

Understanding the Report Format

All Eucalyptus reports contain a usage section. The instance report also contains a running time section.

The usage section shows cumulative (**cumul.**) metrics for each zone, account, and user. Then the report lists metrics for each resource. The column for each resource type (for example, **Instance Id** or **Volume Id** displays **cumul.** for all cumulative metrics. When individual resources are reported, the individual resource's name or identifier displays in that column.

Reporting Best Practices

- Eucalyptus recommends that you run reports from the data warehouse. The Cloud Controller (CLC) generates the data. The data warehouse is a store of the stale data exported from the CLC.
- Monitor the rate of information collected and written to the CLC database. The database expands through usage and event-driven records. More report information stored in the CLC database lessens the effectiveness of the CLC to

perform its cloud duties. If the database gets too large, export the data to the data warehouse then delete the data from the CLC.

- Be careful about deleting data in the CLC. If you delete data in the CLC after you export it, you should use the data warehouse to generate all future reports. This ensures that you have a comprehensive picture of your cloud data.
- You can't import data from different clouds into the same data warehouse.

Reporting Tasks

This section explains the tasks associated with the Eucalyptus reporting feature. These tasks are divided into where you will run reports from, either the Cloud Controller (CLC) or the data warehouse. Follow the steps listed below to configure and then find the tasks associated with reports.

When you install Eucalyptus, you automatically get the ability to run reports against the CLC. This reporting functionality is done in the Eucalyptus Administrator Console. For further information, see [Reporting Tasks: CLC](#).

Many environments choose to focus the CLC function on the cloud processes, rather than on reporting processes. For these needs, Eucalyptus supports exporting data from the CLC to a data warehouse and running reports against the data in that data warehouse. For more information, see [Reporting Tasks: Data Warehouse](#).

Reporting Tasks: CLC

A Eucalyptus installation gives you the ability to run reports against your Cloud Controller (CLC). For information about each report type, see the following sections.

Create a Report: CLC

To create a report using data on the Cloud Controller (CLC):

1. Open the Eucalyptus Administrator Console.
2. Click the **Usage Report** link in the **Resource Management** section.
The Usage Report page displays.
3. Choose the following fields:
 - Enter the starting date in the **From** field.
 - Enter the end date in the **Through** field.
 - Enter the report type you want to generate in the **Report type** field.
4. Click **Generate**.
The report displays on the screen.

To download the report, click the **CSV** or **HTML** icon at the bottom of the screen.

Reporting Tasks: Data Warehouse

Eucalyptus recommends that you run your reports against the data warehouse. Setting up a data warehouse allows you to remove data from the Cloud Controller (CLC). This ensures that you have enough disk space to operate the CLC. This section contains information needed to install the data warehouse and run those reports.

Once the data warehouse is installed, the workflow for running reports against the data warehouse is:

1. Export the data from the CLC. For more information, see [Export Data](#).
2. Import the data to the data warehouse. For more information, see [Import Data](#).
3. Create the report from the data in the data warehouse. For more information, see [Create a Report: Data Warehouse](#).

Set Up the Data Warehouse

This section explains how to set up the data warehouse and how to generate reports using data in the data warehouse.

Install the Data Warehouse

To install the Data Warehouse on hosts running RHEL 6 or CentOS 6:



Important: Do not install the Data Warehouse on a machine running Eucalyptus services.

1. Configure the Eucalyptus package repository on the Data Warehouse host:

```
yum --nogpgcheck install
http://downloads.eucalyptus.com/software/eucalyptus/3.2/centos/6/x86_64/
eucalyptus-release-3.2.noarch.rpm
```

2. Install the Data Warehouse packages:

```
yum install eucadw
```

3. Install the PostgreSQL server:

```
yum install postgresql91-server
```

You are now ready to [Configure the Database](#).

Configure the Database

To configure the database in your data warehouse:

1. Initialize the PostgreSQL database.

```
service postgresql-9.1 initdb
```

2. Start the PostgreSQL service.

```
service postgresql-9.1 start
```

3. Log in to the PostgreSQL server.

```
su - postgres
```

4. Start the PostgreSQL terminal.

```
psql
```

5. At the psql prompt run:

```
create database eucalyptus_reporting;
create user eucalyptus with password 'mypassword';
grant all on database eucalyptus_reporting to eucalyptus;
\q
```

6. Log out.

```
exit
```

7. Edit the `/var/lib/pgsql/9.1/data/pg_hba.conf` file to contain the following content:

```
local    all             all                                     password
host     all             all             127.0.0.1/32          password
host     all             all             ::1/128               password
```

8. Reload the PostgreSQL service.

```
service postgresql-9.1 reload
```

Your machine is now configured as a data warehouse.

Check the Data Warehouse Status

To check the status of the data warehouse:

Enter the following command:

```
eucadw-status -p <your_password>
```

For more information about `eucadw-status`, go to the [Command Line Interface Reference Guide](#).

Export Data

To export data from the Cloud Controller (CLC):

Run the following command:

```
eureport-export-data [filename] -s [start_date] -e [end_date]
-d
```

For more information about the `eureport-export-data` command, go to the [Command Line Interface Reference Guide](#).

Import Data

To import data into the data warehouse:

Run the following command:

```
eucadw-import-data -e [filename] -p [your_password]
```

where `filename` is the name of the imported file that you want to get data from.

For more information about `eucadw-import-data`, go to the [Command Line Interface Reference Guide](#).

Create a Report: Data Warehouse

To create a report from data in the data warehouse:

Run the following command:

```
eucadw-generate-report -s <start_date> -e <end_date> -t <report_type> -p
<your_password>
```

where:

- `start_date` is the date you want data from. For example, 2012-11-05.
- `end_date` is the date you want data to.
- `report_type` is the type of report you want to run: instance, S3, volume, snapshot, IP, or capacity.
- `your_password` is the administrator password you configured in the data warehouse installation.

For more information about `eucadw-generate-report`, go to the [Command Line Interface Reference Guide](#).

Appendix

This appendix for the Eucalyptus Administration Guide provides information about administrative commands and general troubleshooting help.

Troubleshooting Eucalyptus

This section covers common management or configuration problems and solutions for fixing these problems.

To troubleshoot Eucalyptus, the administrator must know the location of the Eucalyptus components, that is, on which machine each component is installed. The administrator must have root access to each machine hosting the components and must understand the network configuration connecting the components.

Log Files

Usually when an issue arises in Eucalyptus, you can find information that points to the nature of the problem either in the Eucalyptus log files or in the system log files. By default, the Eucalyptus log files are stored in `/var/log/eucalyptus/` on each machine that hosts a Eucalyptus component. If Eucalyptus is installed somewhere other than the filesystem root (`/`), log files are stored in `$EUCALYPTUS/var/log/eucalyptus/`.

Here are the relevant logs for each component:

Cloud controller (CLC), Walrus, Storage controller (SC), and VMware Broker

- `cloud-output.log`
- `euca_imager.log`

These components also include specialized developer log files. These are not relevant to troubleshooting a production system, and are not affected by any log level settings. These logs include the following:

- `cloud-debug.log`
- `cloud-error.log`
- `cloud-exhaust.log`
- `cloud-extreme.log`

Cluster controller (CC)

- `cc.log`
- `axis2c.log`
- `httpd-cc_error_log`

Node controller (NC)

- `nc.log`
- `axis2c.log`
- `httpd-nc_error_log`
- `euca_test_nc.log`

System Logs

You might also find helpful information about the nature of an issue in the system logs. In particular, the following logs may be relevant:

- `/var/log/messages`
- `/var/log/libvirt/`

- `/var/log/xen/`

Configuring Logging

For the cluster controller and the node controller, log level is configured using the `LOGLEVEL` parameter in `eucalyptus.conf`. This parameter will be picked up dynamically when the value is changed in the config file, without requiring a restart of the component.

For all other components, the log level can be configured by passing an appropriate `--log-level` argument in the init script. It can also be dynamically changed using `euca-modify-property` to set an appropriate value for `cloud.euca_log_level`. This takes precedence over the value specified in the init script.

Valid log levels are as follows, from most to least verbose:

- ALL
- EXTREME
- TRACE
- DEBUG
- INFO
- WARN
- ERROR
- FATAL
- OFF

If no value is specified, the default `INFO` is used.

Log Format

Eucalyptus logs now have a standard format, which varies slightly per log level.

For log levels `FATAL`, `ERROR`, `WARN` and `INFO`:

```
YYYY-MM-DD HH:MM:SS LEVEL | message
```

For log levels `DEBUG` and `TRACE`:

```
YYYY-MM-DD HH:MM:SS LEVEL PROCESS:THREAD loggingMethodOrClass | message
```

For log level `EXTREME` and `ALL`:

```
YYYY-MM-DD HH:MM:SS LEVEL PROCESS:THREAD loggingMethodOrClass  
FILENAME:LineNumber | message
```

Fault Logs

Eucalyptus includes fault logs for easy identification of conditions outside of Eucalyptus's control that may cause it to fail. These messages are logged per component, and each fault is logged only once per component, in `/var/log/eucalyptus/[component]-fault.log`. The messages include a suggested resolution, and can be customized. Where they have been translated, Eucalyptus will use the system-configured `LOCALE` variable to serve appropriate messages.

Fault messages are based on XML-formatted templates, stored in a per-locale directory structure, with one file per fault message, and one file storing common strings. Default templates are shipped with Eucalyptus. These are stored in `/usr/share/eucalyptus/faults/` as follows:

```
/usr/share/eucalyptus/faults/en_US/0001.xml  
...
```

```
/usr/share/eucalyptus/faults/en_US/1234.xml
/usr/share/eucalyptus/faults/en_US/common.xml
```

Using Localized Fault Logs

Localized messages are located in a per-locale directory under `/usr/share/eucalyptus/faults/`. If localized messages are available matching the system `LOCALE`, Eucalyptus will use those messages. If no `LOCALE` is set, Eucalyptus defaults to `en_US`.

Set the system `LOCALE` in `/etc/sysconfig/i18n` as follows:

```
LOCALE=ru_RU
```

Using Customized Fault Logs

To use your own customized messages, copy the message files to the appropriate directory under `/etc/eucalyptus/faults/` and edit them. Do not change the filenames. To test the fault template, run `euca-generate-fault`, providing the component name, fault ID, and any relevant parameters along with their values.

```
euca-generate-fault -c component_name fault_id [param] [value]
```

For example

```
euca-generate-fault -c broker 1008 daemon ntpd
```

The test fault should be logged in the appropriate component fault log (in this case, `/var/log/eucalyptus/broker-fault.log`)

Eucalyptus uses customized messages where they are available, preferring a non-localized custom message over a localized default message. Localized messages should be in a per-locale directory under `/etc/eucalyptus/faults/`, with a directory name that matches the system `LOCALE`. If no `LOCALE` is set, Eucalyptus defaults to `en_US`.

Network Information

When you have to troubleshoot, it's important to understand the elements of the network on your system. Here are some ideas for finding out information about your network:

- It is also important to understand the elements of the network on your system. For example, you might want to list bridges to see which devices are enslaved by the bridge. To do this, use the `brctl` command.
- You might also want to list network devices and evaluate existing configurations. To do this, use these commands: `ip`, `ifconfig`, and `route`.
- If you are running Eucalyptus in Managed networking mode, you can also use `vconfig` to evaluate VLAN configuration.
- You can get further information if you use the `euca-describe` commands with the verbose options. For example, `euca-describe-instances verbose` returns all instances running by all users on the system. Other describe commands are:
 - `euca-describe-volumes verbose`
 - `euca-describe-snapshots verbose`
 - `euca-describe-groups verbose`
 - `euca-describe-keypairs verbose`

Walrus and Storage

Walrus decryption failed. On Ubuntu 10.04 LTS, kernel version 2.6.32-31 includes a bug that prevents Walrus from decrypting images. This can be determined from the following line in cloud-output.log

```
java.lang.IllegalArgumentException: pad block corrupted
```

If you are running this kernel:

1. Update to kernel version 2.6.32-33 or higher.
2. De-register the failed image (euca-deregister).
3. Re-register the bundle that you uploaded (euca-register <bucket>/<manifest>).

Walrus physical disk is not large enough.

1. Stop the CLC.
2. Add a disk.
3. Migrate your data.

Make sure you use LVM with your new disk drive(s).

Access and Identities

Need to verify an existing LIC file.

1. Enter the following command:

```
/usr/sbin/euca-describe-properties | grep ldap
```

The output from the example above shows the name of the LIC file and status of the synchronization (set to false).

```
PROPERTY authentication.ldap_integration_configuration
{ 'sync': { 'enable': 'false' } }
```

Windows Images

Properties

A typical size of Windows images is large and Eucalyptus has a set of properties that limit the size of various storage components. The first step in troubleshooting is to make sure that the values are large enough to store your Windows images. You can modify a property using

```
/usr/sbin/euca-modify-property -p <property>=<value>
```

The properties that might affect registering Windows images are:

- `walrus.storage.maxbucketsizeinmb`: max bucket size enforced by Walrus; should be larger than a Windows image
- `walrus.storage.maxcachesizeinmb`: total size of all images that is cached in Walrus; should be larger than the sum of all images (Windows/Linux) in Walrus
- `walrus.storage.maxtotalsnapshotsizingb`: if a Windows image is a type of EBS-backed EMI, this should be large enough to store all EBS backed images
- `{PARTITION}.storage.maxvolumesizingb`: if a Windows image is a type of EBS-backed EMI, this should be large enough to store the image

In addition, during the `euca-run-instances`, the CLC may time-out an instance while a large windows image (images in both Walrus and EBS) is being launched. We recommend that you raise the values of the following properties.

- `cloud.vmstate.instance_timeout`: maximum wait time, in minutes, before the instance becomes running. An instance cannot stay in pending longer than this. Default: 60
- `cloud.vmstate.ebs_volume_creation_timeout`: maximum wait time, in minutes, before a volume backing a boot from EBS image is created. Default: 30
- `cloud.addresses.maxkillorphans`: The public IP assigned to an instance will be expired after the time limit. The exact time-out is `{maxkillorphans*5}` seconds (by default it's 50 seconds). If the volume backing an EBS image is not created in time, the public IP will be released from the instance.

Image Preparation

- `euca-bundle-image` hangs** The time to bundle an image is proportional to the image size. Because the typical size of Windows image is big, give enough time until bundling is complete. As a rule of thumb, it may take up to 20 min. for bundling a 10 GB Windows image.
- `euca-upload-bundle` fails** Make sure `'walrus.storage.maxbucketsizeinmb'` is large enough. If not, ask your administrator.

Instance Launch and Login

- Instance stays in pending** Typically, it takes longer to launch Windows images than Linux images as the delay is proportional to the image size. This can be especially long when the image is seeded on NCs the first time (images are cached in NCs and run within few seconds thereafter). As a rule of thumb, 10 GB Windows images may take up to 10 minutes to become 'running' when it is not cached in NCs.
- Instance stay in pending and goes to shutdown** An instance may time-out if the Windows image is too big. Review and adjust the relevant properties.
- Instance is running, but not accessible using Remote Desktop.** after instances become running, you should wait until Windows is fully booted. If the image is sysprepped, the booting time may take up to 10 min. Also you should make sure the followings are cleared:
- The port 3389 is authorized in the security group
 - If the instance is attached to your active directory domain, the domain GPO shouldn't block the RDP port (3389)
 - The username should be authorized to log-in using Remote Desktop (refer to User guide: Windows integration service)
- Finding the login username and password** Use Administrator and the password retrieved by `euca-get-password`. If the instance is attached to a domain, you may use your domain username and password (make sure the username is prepended with domain name, such as `YOUR_DOMAIN\Alice`).
- Can't retrieve windows password using `euca-get-password`** Make sure the platform field of your windows EMI is set to 'windows', not 'linux' (use `euca-describe-images`). If not, the most likely reason is that the image name does not begin with 'windows'. You should bundle/upload/register the image with a proper name.
- Instance is not attached to an Active Directory domain**
- Make sure the parameters set in Windows integration service are correct. One way to verify them is to log in the instance using Administrator password and manually attach the instance to the domain (System Properties -> Computer Name) using the same parameters.
 - Make sure `VNET_DNS` in `eucalyptus.conf` is set to the domain controller (refer to User Guide: Configure Active Directory).

Disk and Volume

Ephemeral disks are not visible in the Windows	Open Disk Management console (All Programs->Administrative Tools->Server Manager->Storage) and find the uninitialized disks. You should create a partition on the disk and format it.
EBS volume is attached, but not visible in the Windows	Open Disk Management console (All Programs->Administrative Tools->Server Manager->Storage) and find the uninitialized disks. You should create a partition on the disk and format it. You don't have to repeat it when the volume is reattached later.
EBS volume is detached, but the disk drive (for example, E:\) is still visible in the Windows	For KVM hypervisor, you should perform "remove hardware safely" before detaching the volume.
euca-bundle-instance fails	Make sure the bucket specified with '-b' option doesn't already exist and the property 'walrus.storagemaxbucketsizeinmb' is large enough to store the image.

Instances

Inaccurate IP addresses display in the output of euca-describe-addresses.	<p>This can occur if you add IPs from the wrong subnet into your public IP pool, do a cleanrestart on the CC, swap out the wrong ones for the right ones, and do another clean restart on the CC. To resolve this issue, run the following:</p> <pre> /etc/init.d/eucalyptus-cloud stop /etc/init.d/eucalyptus-cc stop iptables -F /etc/init.d/eucalyptus-cc cleanrestart /etc/init.d/eucalyptus-cloud start </pre>
NC does not recalculate disk size correctly	<p>This can occur when trying to add extra disk space for instance ephemeral storage. To resolve this, you need to delete the instance cache and restart the NC.</p> <p>For example:</p> <pre> rm -rf /var/lib/eucalyptus/instances/* service eucalyptus-nc restart </pre>

High Availability

In the event that incorrect keys for a secondary CLC are used, Eucalyptus behaves as if that CLC no longer exists. The current primary CLC continues to operate as expected. In order to bring back the secondary CLC, perform the following tasks.

1. Stop the secondary CLC.

```
service eucalyptus-cloud stop
```

2. On the secondary CLC, delete all files from /var/lib/eucalyptus/db.
3. On the secondary CLC, delete all .pem and vtunpass files from var/lib/eucalyptus/keys.
4. Start the secondary CLC.

```
service eucalyptus-cloud start
```

5. Re-register the secondary CLC with the primary CLC.

```
/usr/sbin/euca_conf --register-cloud --partition eucalyptus
--host [Secondary_CLC_IP] --component [CLC_Name]
```

Recovering from a Failure: Walrus

In these examples, we will assume that Walrus WS00 is the primary and WS01 is the secondary Walrus server.

Software Failure Example

In this scenario, WS01 refuses to go to DISABLED state. DRBD complains that it is in split brain mode. `drbdadm cstate r0` shows that DRBD is in WFCnection state.

If you are sure that data on WS01 is out of date and can be discarded, execute the following commands to restore HA mode.

1. Shut down the eucalyptus-cloud process on WS01.
2. Ensure that the DRBD connection is down by typing "drbdadm disconnect r0" on any of the two Walrus hosts.
3. On the primary Walrus, WS00, set drbd as the primary by executing "drbdadm primary r0"
4. On the secondary Walrus, WS01, execute the following command:

```
drbdadm -- --discard-my-data connect
```



Warning: This command will discard all data on WS01 and synchronize data from WS00.

5. Monitor the state of DRBD by running:

```
watch -n 2 cat /proc/drbd
```

6. When the data on WS01 is synced, start the eucalyptus-cloud process on WS01.

Hardware Failure Example

In this example, the primary WS00 needs to be taken out of service due to a hardware failure, such as a failed disk.

1. Shut down the eucalyptus-cloud process on WS00 if it is still running.
2. Monitor service status by running `euca-describe-services` on WS01 and ensure that WS01 has taken over as the new primary (state: ENABLED).
3. Shut down the host running WS00.
4. If the host running WS00 is to be replaced entirely or the OS reinstalled:
 - On the primary CLC, enter the following to deregister WS00:

```
euca_conf --deregister-walrus --component WS00 partition <name of partition>
--host <WS00 host>
```

- After Linux has been installed on the new WS00 host and it is ready for use, please reinstall the "eucalyptus-walrus" package.
- Synchronize the DRBD configuration (`/etc/drbd.conf` and `/etc/eucalyptus/drbd*`) from the WS01 host.
- On WS00, re-configure DRBD by following the Configure DRBD section of the Installation Guide and performing the steps that are relevant to the secondary Walrus server (WS00 is the new secondary Walrus server, in this example).
- Re-register WS00 with a new host name if necessary. This will synchronize keys.

- On WS00, execute the following command:

```
drbdadm -- --discard-my-data connect
```



Warning: This command will discard all data on WS00 and synchronize data from WS01.

- Monitor the state of DRBD by entering:

```
watch -n 2 cat /proc/drbd
```

WS01 should be marked as the primary and WS00 is the new secondary. Wait until data is synchronized.

- When the data on WS00 is synced from WS01, start the eucalyptus-cloud process on WS00.
- Monitor service status by running "euca-describe-services" on the primary CLC and ensure that WS00 is DISABLED and WS01 is ENABLED.

At this point, the Walrus service is back in HA mode.

Configuring SSL for the Admin Console UI

In order to configure SSL for the Admin Console UI, you will need a signed certificate.

- Create a p12 keystore that includes the signed certificate and private key:

```
openssl pkcs12 -export -out signedcert.p12 -inkey [key_file] -in  
[certificate_file] -name jetty -certfile gd_bundle.crt
```

Ensure that the keystore is readable by the Eucalyptus user.

- Verify that the certificate was converted correctly:

```
keytool -exportcert -v -alias jetty -keystore signedcert.p12 -storetype pkcs12  
> certificate.crt  
keytool -printcert -file certificate.crt
```

- Move the new certificate store into place:

```
mv signedcert.p12 /var/lib/eucalyptus/keys/signedcert.p12  
chown eucalyptus:eucalyptus /var/lib/eucalyptus/keys/signedcert.p12  
chmod 600 /var/lib/eucalyptus/keys/signedcert.p12
```

- Create a temporary directory, and extract the eucalyptus-www jar:

```
mkdir /tmp/eucalyptus-www  
cd /tmp/eucalyptus-www  
jar -xf /usr/share/eucalyptus/eucalyptus-www-3.1.2.jar
```

- Edit eucalyptus-jetty.xml to point to the new keystore. The following example assumes that your new keystore is stored in /var/lib/eucalyptus/keys/:

```
<Set name="keystore">/var/lib/eucalyptus/keys/signedcert.p12</Set>  
<Set name="truststore">/var/lib/etc/eucalyptus/keys/signedcert.p12</Set>  
<Set name="password">[yourkeystorepassword]</Set>  
<Set name="keyPassword">[yourkeypassword]</Set>  
<Set name="trustPassword">[yourkeystorepassword]</Set>
```

- Copy the eucalyptus-jetty.xml file to /etc/eucalyptus/cloud.d/.



Important: This file contains the keys for your keystore. Ensure that it is protected, and can only be read or written to by the eucalyptus user.

7. Restart Eucalyptus services:

```
service eucalyptus-cloud restart
service eucalyptus-cc restart
```

After restarting Eucalyptus, verify that the system is up using `euca-describe-services`. You should now be able to access the Admin UI over SSL at `http://[CLC-IP]:8443/`.

Advanced Storage Configuration

This section covers advanced storage provider configuration options.

EMC VNX Advanced Configuration

This section contains advanced configuration, best practices, and troubleshooting tips for the EMC VNX SAN provider.

Configure EMC VNX Synchronous Snapshots

Setting the `<partition>.storage.enablesyncsnaps` property to `true` will cause snapshots to be set synchronously during a `euca-create-snapshot` operation. In this mode, the snapshot is created synchronously before the `euca-create-snapshot` command returns, while the copy and upload to Walrus still takes place asynchronously. This helps ensure that the `euca-create-snapshot` command returns quickly.

If the CLC loses the connection with the SC or if the connection times out (the default timeout is 60 seconds), the SC will detect the connection has been closed and will mark the snapshot as failed and will clean up. This detection occurs after the VNX snapshot has been created, but before it initiates the thread that performs the asynchronous migration and transfer of the snapshot LUN to Walrus. When using synchronous snapshot mode, if the CLC returns an error to the user on the `euca-create-snapshot` command then the snapshot will be marked as failed when listing snapshots using the `euca-describe-snapshots` command.

To configure synchronous snapshots for an EMC VNX SAN:

Set the `<partition>.storage.enablesyncsnaps` property to `true`:

```
euca-modify-property -p mypartition.storage.enablesyncsnaps=true
```

You have now successfully configured synchronous snapshots for your EMC VNX SAN installation.

Best Practices for Multipathing with EMC VNX



Note: FEATURE PREVIEW: The multipathing feature is not yet complete, and may change or be removed from future releases. It is included in this release so that users can try it out and provide feedback.

The primary goals for multipathing with EMC VNX as a Eucalyptus EBS backend are to:

- Avoid single points of failure
- Maximize bandwidth for data access
- Isolate control traffic from data traffic to avoid performance problems

To achieve these goals, some best practice suggestions for multipathing are:

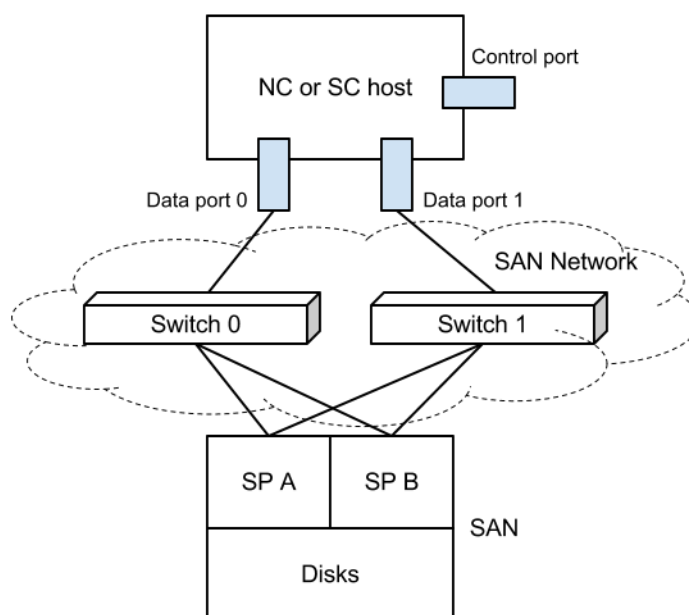
- Have at least two distinct networks for the data paths between NC/SC hosts and the SAN, so that there is no single point failure on the data path.

- Have separate network interfaces for NC and SC data and control traffic, to minimize the traffic interferences and maximize data bandwidth. Data access interfaces can use larger pipes, like 10GB Ethernet.
- Connect both SPs on the SAN to all of the data access networks.

The following diagram shows a typical multipathing configuration with EMC VNX. In this diagram, NC/SC hosts have 3 network interfaces: data port 0 and data port 1 for iSCSI data access, and the control port, which is used for control messages for Eucalyptus internal traffic. Each of the data port connects to a separate switch: switch 0 and switch 1. Each of the SAN storage processors, SP A and SP B, connects to both switches. In this diagram, we have 4 distinct iSCSI paths for each storage volume:

1. Data port 0 Switch 0 SP A
2. Data port 0 Switch 0 SP B
3. Data port 1 Switch 1 SP A
4. Data port 1 Switch 1 SP B

In this scenario, failure of any of the paths will not affect the storage access to the volumes:



Troubleshooting EMC VNX Multipathing



Note: FEATURE PREVIEW: The multipathing feature is not yet complete, and may change or be removed from future releases. It is included in this release so that users can try it out and provide feedback.

The Eucalyptus EMC VNX Multipathing feature requires the following to function properly:

- Properly installed and configured Linux Device Mapper Multipathing software on both the Storage Controller and Node Controller hosts.

- Correctly configured iSCSI path system property and related `STORAGE_INTERFACES` parameters in the “eucalyptus.conf” configuration file for both SC and NC.

Prerequisites for Troubleshooting Typical Multipathing Failures

Before you start diagnosing the problems with multipathing, make sure you set the proper logging level on both SC and NC machines, so that you can get detailed failure logs. To do that:

- Set the “cloud.euca_log_level” system property to “DEBUG”
- Uncomment the “LOGLEVEL=DEBUG” entry in the “eucalyptus.conf” file on the NC, and then restart the NC service

General Troubleshooting Techniques for Multipathing Failures

The following are general tips to help diagnose multipathing problems:

- Make sure you turn on the DEBUG log level for both SC and NC so that you can get detailed information from the logs.
- Eucalyptus calls some external Perl scripts to perform the actual iSCSI connect/disconnect operations. These scripts are:
 - `/usr/share/eucalyptus/connect_iscsitarget.pl`
 - `/usr/share/eucalyptus/disconnect_iscsitarget.pl`
 - `/usr/share/eucalyptus/get_iscsitarget.pl`

The `STDERR` output of these scripts is logged; you can add debug code to print information to `STDERR` to see what happens during connection or disconnection operations.

- The `iscsiadm open-iscsi` initiator command line tool can help you get the current status of all the iSCSI connections in the system. For example:

```
iscsiadm -m session -P 3
```

- Use the `multipath` command line tool to see multipathing status. For example:

```
multipath -ll -v 3
```

Cannot attach volumes

This can occur for a number of reasons. To diagnose this, try some of the following:

- Make sure you can attach a volume without using multipathing.
- Check your SAN-related system properties to see if you have set the correct values.
- Use a single path for the NC; for example, set “`PARTITION.storage.ncpaths`” to something like “192.168.25.182”. If you specify an iface in your path, like “`iface0:192.168.25.182`”, also make sure you have “`iface0`” defined with “`STORAGE_INTERFACES`” in “eucalyptus.conf” configuration file on the NC.
- If you have no problem attaching a volume with a single path, the failure may be due to the incorrect state of the Linux device mapper multipathing tool. Check if the “`multipathd`” service is running on the NC hosts and if “`/etc/multipath.conf`” is installed and configured properly (for example, copy the example configuration provided by Eucalyptus). Remember to set “`user_friendly_names`” to “yes” in “`/etc/multipath.conf`”. You can try restarting “`multipathd`” and/or reloading “`/etc/multipath.conf`” if you changed it previously. Run “`multipath -ll`” on NC host and see if it returns reasonable output without any error.
- Check that the “`PARTITION.storage.ncpaths`” configuration file entries are correct. A typo can cause volume attach failures.
- Make sure that the networking configuration is correct for the NC hosts. If you set the paths without specific ifaces, check to see if you can connect to each IP in the path using default network interface; otherwise, check each path’s connectivity using a specific network interface.
- Check network connectivity with all of the configured paths.
- Check the “`nc.log`” log file for the string “`connect_iscsitarget`”. Examine the return results, especially the “`stderr`” output.

Not all paths are connected

Sometimes when you run “`multipath -ll`” on NC hosts after attaching a volume, you find that the `multipath` device does not have all of the paths connected. In this case, the problem could be due to one of the following:

- There is a mistake in the paths in one of the “PARTITION.storage.ncpaths” entries. If one of the paths specified in the system property is wrong, then it is possible that the specific path can not be connected. Make sure you have all the paths specified correctly.
- The missing paths are not valid networking paths, or have networking issues. For example, when you ignore the iface part of a path, are you sure that the destination of the path (the IP part of the path) can be connected via the default network interface? Or if you specified the iface, are you sure you defined the iface in the “eucalyptus.conf” configuration file, and that the destination can be connected with the specified network interface?
- If the paths specified are all valid, but some of them do not have connectivity, try to ping each of the specified paths on the NC hosts to check for connectivity. If there are connectivity issues, contact your network administrator.

Snapshotting failed

The Eucalyptus Storage Controller needs to attach a volume on the machine it runs so it can upload to Walrus during an EC2 snapshot call. To help ensure maximum reliability for snapshotting, you should use multipathing for the SC host; this is configured with the “PARTITION.storage.scpaths” system property. When multipathing is enabled for the SC, if you see a snapshot failure, it may be caused by multipathing. Techniques for diagnosing SC multipathing failures is similar to those used for NC multipathing failures. In the case of SC multipathing failures, the logs are in “/var/log/eucalyptus/cloud-*.log”, not “nc.log”, since the iSCSI connect scripts are invoked by Java code.

NetApp Advanced Configuration

This section contains advanced configuration, best practices, and troubleshooting tips for the NetApp SAN provider.

Configurable NetApp SAN Properties

The following table lists the NetApp SAN-specific properties you can configure using `euca-modify-property`, along with their valid values and Eucalyptus default values.



Note: The following configuration options are a subset of the Netapp SAN configuration parameters. Changing these default values may cause storage operations to fail. Please proceed at your own risk. For more information on NetApp configuration, please refer to the [NetApp Data ONTAP 7G documentation](#) (this link requires you to register and login).

Eucalyptus Property	Description	Valid Values
<region>.storage.convertunicode	Setting this option to "on" forces conversion of all directories to UNICODE format when accessed from both NFS and CIFS.	"on" (default) or "off"
<region>.storage.createunicode	Setting this option to "on" forces UNICODE format directories to be created by default from NFS and CIFS.	"on" (default) or "off"
<region>.storage.fractionalreserve	The percentage of space reserved for overwrites of reserved objects (LUNs or files) in a volume.	0-100; default is 0
<region>.storage.noatimeupdate	Prevents the update of inode access times when a file is read.	"on" (default) or "off"
<region>.storage.tryfirst	Determines if the volume size is increased before deleting snapshots if enableautosize property is "true".	"volume_grow" (default) or "snap_delete"

Eucalyptus Property	Description	Valid Values
<region>.storage.guarantee	Controls space reservation for flexible volumes. See the NetApp SDK documentation for more information.	"none", "file", or "volume" (default)
<region>.storage.enableautosize	Toggles the flex volume autosize feature.	"true" (default) or "false"
<region>.storage.volautosizemaxmultiplier	Flex volume's maximum size allowed, specified as a multiple of the original size	Integer >= 1; default is 3
<region>.storage.volautosizeincrementinmb	Flex volume's increment size in megabytes.	Integer >= 1; default is 256
<region>.storage.snapschedweeks	Number of weekly snapshots to keep online.	Integer >= 0; default is 0
<region>.storage.snapscheddaily	Number of daily snapshots to keep online.	Integer >= 0; default is 0
<region>.storage.snapschedhourly	Number of hourly snapshots to keep online.	Integer >= 0; default is 0
<region>.storage.snappercent	Additional space reserved on the flex volume to store automatic and manual snapshots created outside of Eucalyptus. The amount of space to be reserved is specified as a percentage of the flex volume.	Integer >= 0; default is 0

Glossary

cluster

A group of resources that contains a CC, an SC and, optionally, a VMware Broker.

availability zone

An availability zone for AWS denotes a large subset of their cloud environment. Eucalyptus refines this definition to denote a subset of the cloud that shares a local area network. Each availability zone has its own cluster controller and storage controller.

AWS

Amazon Web Services

bucket storage

A storage container that accepts objects via PUT and GET commands.

bundling

A virtual machine image splits the image into multiple image parts to facilitate ease of uploading. It also generates an XML manifest file containing metadata referencing the image, including image parts and kernel, which is used to assemble instances of the image.

Cloud Controller

The Cloud Controller (CLC) is the entry-point into the cloud for administrators, developers, project managers, and end-users. The CLC queries the node managers [SM1] for information about resources, makes high-level scheduling decisions, and makes requests to the Cluster Controllers (CCs). As the interface to the management platform, the CLC is responsible for exposing and managing the underlying virtualized resources (servers, network, and storage). You can access the CLC through Amazon's Elastic Compute Cloud (EC2) and through a web-based Eucalyptus Administrator Console.

Cluster Controller

The Cluster Controller (CC) generally executes on a machine that has network connectivity to both the machines running the Node Controller (NC) and to the machine running the CLC. CCs gather information about a set of node machines and schedules virtual machine (VM) execution on specific nodes. The CC also manages the virtual machine networks and participates in the enforcement of SLAs[SM3] as directed by the CLC. All nodes associated with a single CC must be in the same broadcast domain (Ethernet).

kernel/ramdisk pair

A ramdisk contains drivers that direct the kernel to launch appropriate system files when instantiating a virtual machine.

Node Controller

The Node Controller (NC) executes on any machine that hosts VM instances. The NC controls VM activities, including the execution, inspection, and termination of VM instances. It also fetches and maintains a local cache of instance images, and it queries and controls the system software (host OS and the hypervisor) in response to queries and control requests from the CC. The NC is also responsible for the management of the virtual network endpoint.

Storage Controller

The Storage Controller (SC) provides functionality similar to the [5] Amazon Elastic Block Storage (EBS) and is capable of interfacing with various storage systems (NFS, iSCSI, SAN devices, etc.). Elastic block storage exports storage volumes that can be attached by a VM and mounted or accessed as a raw block device. EBS volumes persist past VM termination and are commonly used to store persistent data. An EBS volume cannot be shared between VMs and can only be accessed within the same availability zone in which the VM is running. Users can create snapshots from EBS volumes. Snapshots are stored in Walrus and made available across availability zones. Eucalyptus with SAN support lets you use your enterprise-grade SAN devices to host EBS storage within a Eucalyptus cloud.

VMware Broker

VMware Broker (Broker or VB) is an optional Eucalyptus component activated only in versions of Eucalyptus with VMware support. VMware Broker enables Eucalyptus to deploy VMs on VMware infrastructure elements and mediates all interactions between the Cluster Controller (CC) and VMware hypervisors (ESX/ESXi) either directly or through VMware vCenter.

Walrus

Walrus allows users to store persistent data, organized as buckets and objects. You can use Walrus to create, delete, and list buckets, or to put, get, and delete objects, or to set access control policies. Walrus is interface compatible with Amazon's Simple Storage Service (S3), providing a mechanism for storing and accessing virtual machine images and

user data. Note that Walrus access is global to the entire Eucalyptus cloud. This means that it can be accessed by end-users, whether the user is running a client from outside the cloud or from a virtual machine instance running inside the cloud.

Index

A

access

[23, 41–42](#)

IAM [23](#)

types of [23](#)

use case [41–42](#)

access tasks

[40, 43–55](#)

accounts

[43–45](#)

add an account [43](#)

approve an account [43](#)

delete an account [45](#)

list all [44](#)

reject an account [44](#)

rename an account [44](#)

credentials

[53–54](#)

generate [53](#)

get administrator credentials [54](#)

retrieve existing [54](#)

upload a certificate [54](#)

groups

[46–49](#)

add a policy [46](#)

create a group [46](#)

delete a group [49](#)

list all [48](#)

list policies [48](#)

modify a group [47](#)

remove user [47](#)

LDAP/AD

[55](#)

synchronize [55](#)

LIC file

[55](#)

start [55](#)

upload [55](#)

list of [40](#)

users

[50–52](#)

add a user [50](#)

add a user to group [50](#)

create login profile [51](#)

delete a user [52](#)

list all [52](#)

modify a user [51](#)

Active Directory [71](#)

Adding Windows Image [75](#)

C

cloud

[8–9, 11, 13](#)

best practices [9, 11](#)

high availability in [11](#)

overview [8](#)

cloud (*continued*)

securing [9](#)

storage volumes

[13](#)

best practices [13](#)

synchronizing clocks [9](#)

timestamp expiration [9](#)

vSphere

[9](#)

working with [9](#)

cloud tasks

[15–19, 21](#)

add a Node Controller [17](#)

back up Eucalyptus [21](#)

inspect system health [15](#)

list arbitrators [17](#)

list of [15](#)

remove a Node Controller [17](#)

restart Eucalyptus [18](#)

restore Eucalyptus [21](#)

shut down Eucalyptus [19, 21](#)

view user resources [16](#)

configuring [92](#)

E

Eucalyptus

[4](#)

accessing

[4](#)

CLI [4](#)

Eucalyptus Administrator Console [4](#)

overview [4](#)

F

fail

[90](#)

recovering from [90](#)

I

image

[57](#)

defined [57](#)

pre-packaged [57](#)

types of [57](#)

image tasks

[14, 57, 60, 63–65, 76–78](#)

add an image [57](#)

associate kernel and ramdisk [77](#)

bundle for Amazon EC2 [78](#)

bundle Windows instance [78](#)

caching [14](#)

create an image

[64–65, 76](#)

Linux [76](#)

- image tasks *(continued)*
 - create an image *(continued)*
 - Windows [65](#)
 - install from EuStore [60](#)
 - list of [57](#)
 - modify an image [63](#)

Images

- [65, 74–75](#)

Windows

- [65, 74–75](#)

- Adding [75](#)

- Convert VMDK [74](#)

R

- Remote Desktop [73](#)

S

SSL

- [91](#)

- Admin Console [91](#)

- Sysprep [74](#)

T

troubleshooting

- [84, 86–87, 89, 93](#)

- access and identities [87](#)

- high availability [89](#)

- instances [89](#)

- log files [84](#)

- multipathing [93](#)

- network information [86](#)

- Walrus and storage [87](#)

- Windows images [87](#)