Setting Up Euca2ools and FireFox ElasticFox on Windows

## Caveat

The following was done on a standard IT Windows 7 laptop. I would hope they work in other Windows environment (e.g. XP etc.). Also, I have not “system tested” these instructions. This just documents what I had to do to get ElasticFox and Euca2ools working on my Windows 7 environment – hopefully I did not forget something.

Also, I would recommend reading these complete instructions before starting. It may avoid troubles for you if you know what is coming.

Lastly, the euca2ools do not have any log files I can find for this installation, so I if you use the debug flags on the euca2ools they will not provide any debugging data. I posted this question on the forum – hopefully I will get an answer.

Good Luck!

## Prerequisites

* You need a tool capable of decrypting *“\*.tar.gz*” files. I could not find a good tool on the internet for doing this in Windows so one option is to go the IT Self Service web site at <https://itoselfservice.it.att.com/SelfService/homeMenu.jsf> and place an order for WinZip.
* If not already on your PC, you will need to contact IT and have them install Windows Powershell 2.0 for you.
* If you do not have the *putty.exe* and *puttygen.exe* applications installed, you will need IT to install both of these for you. These are SSH utilities. Note the directory where these are installed, you will need this later to set PATH names in your Powershell profile.

## Installing Euca2ools

Go to the following web site <http://open.eucalyptus.com/participate/wiki/euca2ools-windows> and follow the instructions for steps 1-7. **Do not** follow the instructions for step 8 “Setup Powershell”, there are specific instructions below in this memo for accomplishing Powershell setup. The instructions in the web site are incorrect because the Powershell script at that web site does not work. Please use the one I included in this document.

Notes:

* Please note the directory you install Euca2tools into. You will need this later for inclusion in Powershell PATH variables.
* Step 1: If you use PowerShell you do not need to include the Python 2.7 in your PATH, I will include that as part of the “Setting Up Powershell” below.
* Step 3: It asks for you to install Win32\_OpenSSLv0.9.8p but when going to the web site there was no such version, so I installed Win32\_OpenSSLv0.9.8r.
* Step 5: You need to run the python setup commands from ....\euca2ools-1.3.1\euca2ools directory.

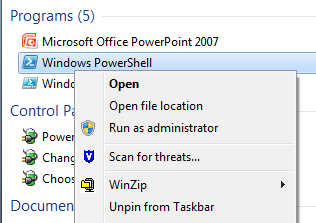
## Setting up Powershell

If you do not have Windows Powershell on your PC you will need to contact IT and get Windows Powershell 2.0 installed (I had a Windows 7 image and it was included, but I do not think this is the case for XP).

Under the MyDocuments in Windows 7 there should be a sub-directory “WindowsPowerShell”. Change directory to the WindowsPowerShell directory and copy the Powershell profile in the *Appendix: Windows PowerShell Profile* section in this document into a file named Microsoft.PowerShell\_profile.ps1 (use notepad to do this). Please note the highlighted items in the profile because you may need to edit these depending on where you installed Euca2ools, Python and the Putty utilities.

Once this is done you will need to bring up a Windows Powershell and execute the following command: *Set-executionpolicy unrestricted*

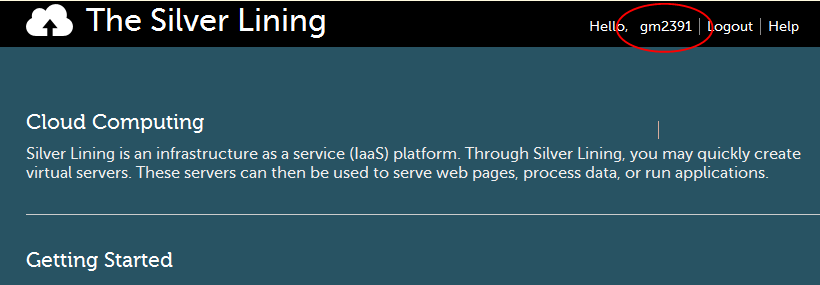
If you get an error executing this within Powershell because you do not have permissions you may need to do run Windows PowerShell as an administrator. On Windows 7 you do this by right clicking the Windows Powershell icon and “Run as administrator”. If this does not work call IT, you may need administrator privileges. It worked for me. On XP I have no idea how to do this.



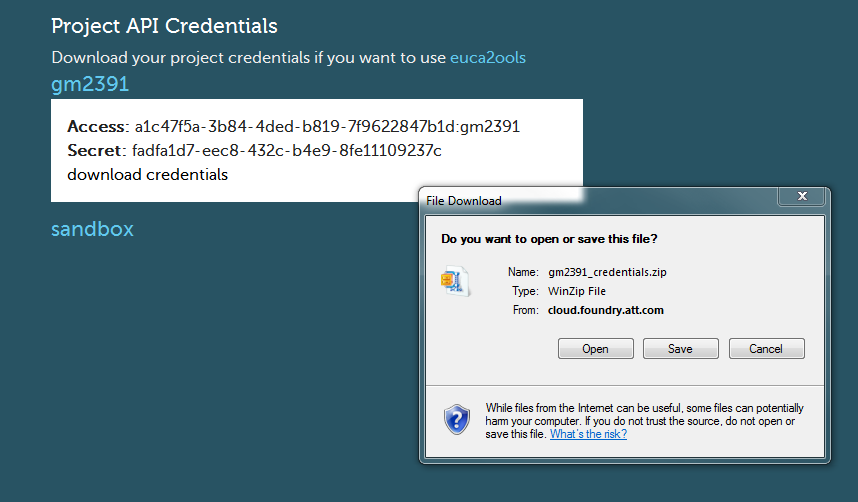
## Setting Up Euca2tools “eucarc” File

Before you can execute the euca2ools primitives, you need to get an account in Silver Lining. Once you do that you will need to get the Silver Lining credential file and unzip it into your euca2ools directory (e.g. C:\Euca2ools\euca2ools-1.3.1). Once you do that you should have the following files: novarc, pk.pem, cert.pem and cacert.pem. Because the *novarc* file format is not compatible with the Powershell profile scripts and environment you will need to create another file that is compatible. Since the Poweshell script in the appendix references a file named *eucarc* please copy/paste the *eucarc* file in the appendix and modify as appropriate using the data from the *novarc* file you got from the Silver Lining site. The appendix shows a color coded mapping between the data in the *novarc* file and the new *eucarc* file. When done place the *eucarc* file in the same euca2ools directory were you unzipped the *novarc* file.

You can get the Silver Lining credential file (e.g. novarc, pk.pem, etc.) by going to the Silver Lining portal and click on your login name in the upper right corner.



Once you do that you can down load and unzip the credentials files by clicking on your id and then select download credentials.



## Install ElasticFox

Open a firefox browser and go to: <http://www.eaglegenomics.com/wp-content/uploads/2011/06/elasticfox.xpi> and install the ElasticFox plugin. Once installed you need to follow the ElasticFox setup as described in the instructions in *Appendix: Getting Started With The Foundry Cloud*. Be careful to only follow the instructions for ElasticFox set up – not euca2tools. Also, note that when entering in the foundry site URL in the regions setup you may need to change the URL to use port 80 instead of 8773. This is what I had to do because 8773 could not get through the corporate firewalls etc.

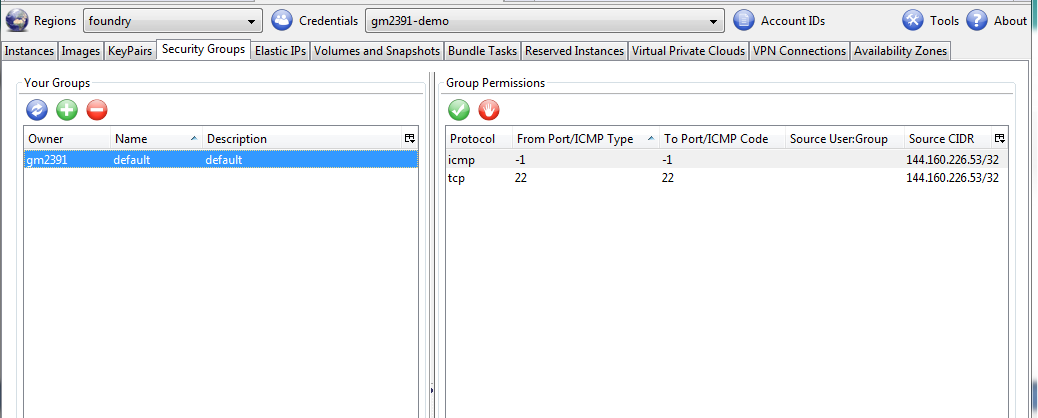
## Using Silver Lining

Now that you have installed the euca2tools you should be able to perform the following the *euca-describe-images* command as a test. The output should look like:



Now this works you will need to be aware of a few more things.

Make sure you have the right security group permissions set up. You can do this via ElasticFox or the euca2ools CLI (but ElasticFox is probably easier). Below is the permissions I used including allowing ICMP so I can ping the instance (a good test to ensure you have connectivity).



Also you will need to create keypairs for accessing your running instances. Once done you will need to convert the keypair file to one that is be compatible with the Windows Putty utilities by using the *puttygen.exe* utility.

To do this you must use ElasticFox – for some reason I could not get this to work when I generated the keypair from euca2tools. So, create a keypair using ElasticFox. Once this is done go to the directory where you saved the keypair file on your laptop and convert it to a Putty compatible keyfile using the *puttygen.exe*. You will need this to access the running instance.

Once this is done go ahead and use Elastic Fox and/or euca2ools to start an instance. You may have to try multiple images, most seem to not work because they immediately shutdown. The only one that seems to work is *ami-000000a5*. Remember you will have to associate an external IP with it to access the running instance.

Now that the instance is running, it is highly recommended that you do not use ElasticFox to access your running image. The utility does not fill in the “host” correctly and will drive you crazy. So to access your instance execute the following command:

*putty –i converted\_keypair.ppk root@ip\_address\_of\_instance*

Given time, I may go back and try to debug why I can’t create a keypair with eua2ools that works with Windows Putty utilities and debug what is wrong with the ElasticFox ${host} macro that does not fill in the correct host IP when accessing the running instance.

This should be it, I hope this all works for you – GOOD LUCK – it is complicated, but once you get it working it is great.

## Appendix: Windows PowerShell Profile

########## Euca2ools powershell wrapper ###############

$env:Path = $env:Path + ";C:\Python27;C:\Putty"

$env:EUCA\_HOME = "C:\Euca2ools\euca2ools-1.3.1"

if($env:EUCA\_HOME){

$EUCA\_HOME = $env:EUCA\_HOME #Get-item -path env:EUCA\_HOME

}else{

write-warning "EUCA\_HOME variable is not set"

$EUCA\_HOME = "C:\Eucalyptus"

}

#write-warning "EUCA\_HOME is set to $EUCA\_HOME"

$EUCA\_RC="$EUCA\_HOME\\eucarc"

[string]$ec2url=""

[string]$access\_key=""

[string]$secret\_key=""

[string]$walrusurl=""

[string]$ec2privatekey=""

[string]$ec2cert=""

[string]$eucalyptuscert=""

[string]$ec2userid=""

if(![System.IO.Directory]::Exists("$EUCA\_HOME")){

throw "The directory '$EUCA\_HOME' doesn't exist"

}

if(![System.IO.File]::Exists($EUCA\_RC)){

throw "File '$EUCA\_RC' doesn't exist"

}

#parse eucarc file and set environment variable properly

$rcfile = get-content "$EUCA\_RC"

foreach ($line in $rcfile)

{

if($line.contains("EC2\_URL="))

{

$ec2url = $line.Substring($line.IndexOf("EC2\_URL=")+8);

}elseif($line.contains("EC2\_ACCESS\_KEY="))

{

$access\_key = $line.Substring($line.IndexOf("EC2\_ACCESS\_KEY=")+15);

$access\_key = $access\_key.Replace("'","")

}elseif($line.contains("EC2\_SECRET\_KEY="))

{

$secret\_key = $line.substring($line.IndexOf("EC2\_SECRET\_KEY=")+15);

$secret\_key = $secret\_key.Replace("'","")

}elseif($line.contains("S3\_URL="))

{

$walrusurl=$line.substring($line.IndexOf("S3\_URL=")+7);

}elseif($line.contains("EC2\_CERT="))

{

$ec2cert=$line.substring($line.IndexOf("EC2\_CERT=")+9);

$ec2cert=$ec2cert.replace("`${EUCA\_KEY\_DIR}/","$EUCA\_HOME\");

}elseif($line.contains("EC2\_PRIVATE\_KEY=")){

$ec2privatekey=$line.substring($line.IndexOf("EC2\_PRIVATE\_KEY=")+16);

$ec2privatekey=$ec2privatekey.replace("`${EUCA\_KEY\_DIR}/","$EUCA\_HOME\");

}elseif($line.contains("EUCALYPTUS\_CERT="))

{

$eucalyptuscert=$line.substring($line.IndexOf("EUCALYPTUS\_CERT=")+16);

$eucalyptuscert=$eucalyptuscert.replace("`${EUCA\_KEY\_DIR}/","$EUCA\_HOME\");

}elseif($line.contains("EC2\_USER\_ID=")){

$ec2userid=$line.substring($line.IndexOf("EC2\_USER\_ID=")+12);

$ec2userid=$ec2userid.Replace("'","")

}

}

if($ec2url -eq ""){

write-output "URL: $ec2url"

throw "EC2\_URL variable is not set"

}

if($access\_key -eq ""){

throw "EC2\_ACCESS\_KEY variable is not set"

}

if($secret\_key -eq ""){

throw "EC2\_SECRET\_KEY variable is not set"

}

if($walrusurl -eq ""){

throw "WALRUS\_URL variable is not set"

}

Set-Item -path env:EC2\_URL -value "$ec2url"

Set-Item -path env:EC2\_ACCESS\_KEY -value "$access\_key"

Set-Item -path env:EC2\_SECRET\_KEY -value "$secret\_key"

Set-Item -path env:WALRUS\_URL -value "$walrusurl"

Set-Item -path env:S3\_URL -value "$walrusurl"

if($ec2cert -ne ""){

Set-Item -path env:EC2\_CERT -value "$ec2cert";

}

if($ec2privatekey -ne ""){

Set-Item -path env:EC2\_PRIVATE\_KEY -value "$ec2privatekey";

}

if($eucalyptuscert -ne ""){

Set-Item -path env:EUCALYPTUS\_CERT -value "$eucalyptuscert";

}

if($ec2userid -ne ""){

Set-Item -path env:EC2\_USER\_ID -value "$ec2userid";

}

#write-output "EUCA\_KEY\_DIR set to $env:EUCA\_KEY\_DIR"

#write-output "EC2\_URL set to $env:EC2\_URL"

#write-output "EC2\_ACESS\_Ket set to $env:EC2\_ACCESS\_KEY"

#write-output "EC2\_SECRET\_KEY set to $env:EC2\_SECRET\_KEY"

#write-output "WALRUS\_URL set to $env:WALRUS\_URL"

#write-output "EC2\_USER\_ID set to $env:EC2\_USER\_ID"

#write-output "EC2\_PRIVATE\_KEY set to $env:EC2\_PRIVATE\_KEY"

#write-output "EC2\_CERT set to $env:EC2\_CERT"

#write-output "EUCALYPTUS\_CERT set to $env:EUCALYPTUS\_CERT"

#write-output "S3\_URL set to $env:S3\_URL"

function euca-add-group{

python "$EUCA\_HOME\bin\euca-add-group" $args

}

function euca-add-keypair{

python "$EUCA\_HOME\bin\euca-add-keypair" $args

}

function euca-allocate-address{

python "$EUCA\_HOME\bin\euca-allocate-address" $args

}

function euca-associate-address{

python "$EUCA\_HOME\bin\euca-associate-address" $args

}

function euca-attach-volume{

python "$EUCA\_HOME\bin\euca-attach-volume" $args

}

function euca-authorize{

#write-output "$args"

python "$EUCA\_HOME\bin\euca-authorize" $args

}

function euca-bundle-image{

python "$EUCA\_HOME\bin\euca-bundle-image" $args

}

function euca-bundle-instance{

python "$EUCA\_HOME\bin\euca-bundle-instance" $args

}

function euca-bundle-upload{

python "$EUCA\_HOME\bin\euca-bundle-upload" $args

}

function euca-bundle-vol{

python "$EUCA\_HOME\bin\euca-bundle-vol" $args

}

function euca-cancel-bundle-task{

python "$EUCA\_HOME\bin\euca-cancel-bundle-task" $args

}

function euca-check-bucket{

python "$EUCA\_HOME\bin\euca-check-bucket" $args

}

function euca-confirm-product-instance{

python "$EUCA\_HOME\bin\euca-confirm-product-instance" $args

}

function euca-create-snapshot{

python "$EUCA\_HOME\bin\euca-create-snapshot" $args

}

function euca-create-volume{

python "$EUCA\_HOME\bin\euca-create-volume" $args

}

function euca-delete-bundle{

python "$EUCA\_HOME\bin\euca-delete-bundle" $args

}

function euca-delete-group{

python "$EUCA\_HOME\bin\euca-delete-group" $args

}

function euca-delete-keypair{

python "$EUCA\_HOME\bin\euca-delete-keypair" $args

}

function euca-delete-snapshot{

python "$EUCA\_HOME\bin\euca-delete-snapshot" $args

}

function euca-delete-volume{

python "$EUCA\_HOME\bin\euca-delete-volume" $args

}

function euca-deregister{

python "$EUCA\_HOME\bin\euca-deregister" $args

}

function euca-describe-addresses{

python "$EUCA\_HOME\bin\euca-describe-addresses" $args

}

function euca-describe-availability-zones{

python "$EUCA\_HOME\bin\euca-describe-availability-zones" $args

}

function euca-describe-bundle-tasks{

python "$EUCA\_HOME\bin\euca-describe-bundle-tasks" $args

}

function euca-describe-groups{

python "$EUCA\_HOME\bin\euca-describe-groups" $args

}

function euca-describe-image-attribute{

python "$EUCA\_HOME\bin\euca-describe-image-attribute" $args

}

function euca-describe-images{

#write-output "PARAM: $args"

python "$EUCA\_HOME\bin\euca-describe-images" $args

}

function euca-describe-instances{

python "$EUCA\_HOME\bin\euca-describe-instances" $args

}

function euca-describe-keypairs{

python "$EUCA\_HOME\bin\euca-describe-keypairs" $args

}

function euca-describe-regions{

python "$EUCA\_HOME\bin\euca-describe-regions" $args

}

function euca-describe-snapshots{

python "$EUCA\_HOME\bin\euca-describe-snapshots" $args

}

function euca-describe-volumes{

python "$EUCA\_HOME\bin\euca-describe-volumes" $args

}

function euca-detach-volume{

python "$EUCA\_HOME\bin\euca-detach-volume" $args

}

function euca-disassociate-address{

python "$EUCA\_HOME\bin\euca-disassociate-address" $args

}

function euca-download-bundle{

python "$EUCA\_HOME\bin\euca-download-bundle" $args

}

function euca-get-console-output{

python "$EUCA\_HOME\bin\euca-get-console-output" $args

}

function euca-get-password{

python "$EUCA\_HOME\bin\euca-get-password" $args

}

function euca-get-password-data{

python "$EUCA\_HOME\bin\euca-get-password-data" $args

}

function euca-modify-image-attribute{

python "$EUCA\_HOME\bin\euca-modify-image-attribute" $args

}

function euca-reboot-instances{

python "$EUCA\_HOME\bin\euca-reboot-instances" $args

}

function euca-register{

python "$EUCA\_HOME\bin\euca-register" $argseuca-describe

}

function euca-release-address{

python "$EUCA\_HOME\bin\euca-release-address" $args

}

function euca-reset-image-attribute{

python "$EUCA\_HOME\bin\euca-reset-image-attribute" $args

}

function euca-revoke{

python "$EUCA\_HOME\bin\euca-revoke" $args

}

function euca-run-instances{

python "$EUCA\_HOME\bin\euca-run-instances" $args

}

function euca-terminate-instances{

python "$EUCA\_HOME\bin\euca-terminate-instances" $args

}

function euca-unbundle{

python "$EUCA\_HOME\bin\euca-unbundle" $args

}

function euca-upload-bundle{

python "$EUCA\_HOME\bin\euca-upload-bundle" $args

}

function euca-version{

python "$EUCA\_HOME\bin\euca-version" $args

}

#########

## Appendix: novarc and eucarc file comparison

You need to transfer the color coded items from the *novarc* file to a corresponding *eucarc* file. Please be very care to do this exactly as I have specified and please beware of the lack of (“) in the *eucarc* file that are present in the equivalent field in the *novarc* file. Please ensure that the only (“) in the *eucarc* file are those that I have in the example file. Also be aware that in my eucarc file the the EC2\_URL is on port 80 and not port 8773. This is because of firewall issues. You will likely have to do the same. Also it should be noted that I did not change anyother URL port numbers. I suspect that it may be likely these may also have to changed depending on if you use services that depend on them. But for getting this to work, I did not have to change these because I was not using S3 etc.

novarc file:

NOVARC=$(readlink -f "${BASH\_SOURCE:-${0}}" 2>/dev/null) ||

NOVARC=$(python -c 'import os,sys; print os.path.abspath(os.path.realpath(sys.argv[1]))' "${BASH\_SOURCE:-${0}}")

NOVA\_KEY\_DIR=${NOVARC%/\*}

export EC2\_ACCESS\_KEY="**a1c47f5a-3b84-4ded-b819-7f9622847b1d:gm2391**"

export EC2\_SECRET\_KEY="**fadfa1d7-eec8-432c-b4e9-8fe11109237c**"

export EC2\_URL="**http://12.208.178.2:80/services/Cloud**"

export S3\_URL="**http://12.208.178.2:3333**"

export EC2\_USER\_ID=**42** # nova does not use user id, but bundling requires it

export EC2\_PRIVATE\_KEY=${NOVA\_KEY\_DIR}/pk.pem

export EC2\_CERT=${NOVA\_KEY\_DIR}/cert.pem

export NOVA\_CERT=${NOVA\_KEY\_DIR}/cacert.pem

export EUCALYPTUS\_CERT=${NOVA\_CERT} # euca-bundle-image seems to require this set

alias ec2-bundle-image="ec2-bundle-image --cert ${EC2\_CERT} --privatekey ${EC2\_PRIVATE\_KEY} --user **42** --ec2cert ${NOVA\_CERT}"

alias ec2-upload-bundle="ec2-upload-bundle -a ${EC2\_ACCESS\_KEY} -s ${EC2\_SECRET\_KEY} --url ${S3\_URL} --ec2cert ${NOVA\_CERT}"

export NOVA\_API\_KEY="**a1c47f5a-3b84-4ded-b819-7f9622847b1d**"

export NOVA\_USERNAME="**gm2391**"

export NOVA\_PROJECT\_ID="**gm2391**"

export NOVA\_URL="**http://12.208.178.2:8774/v1.0/**"

eucarc file:

EUCARC=$(readlink -f "${BASH\_SOURCE:-${0}}" 2>/dev/null) ||

EUCARC=$(python -c 'import os,sys; print os.path.abspath(os.path.realpath(sys.argv[1]))' "${BASH\_SOURCE:-${0}}")

EUCA\_KEY\_DIR=${EUARC%/\*}

export EC2\_ACCESS\_KEY=**a1c47f5a-3b84-4ded-b819-7f9622847b1d:gm2391**

export EC2\_SECRET\_KEY=**fadfa1d7-eec8-432c-b4e9-8fe11109237c**

export EC2\_URL=**http://12.208.178.2:80/services/Cloud**

export S3\_URL=**http://12.208.178.2:3333**

export EC2\_USER\_ID=**42**

export EC2\_PRIVATE\_KEY=${EUCA\_KEY\_DIR}/pk.pem

export EC2\_CERT=${EUCA\_KEY\_DIR}/cert.pem

export EUCALYPTUS\_CERT=${EUCA\_KEY\_DIR}/cacert.pem

alias ec2-bundle-image="ec2-bundle-image --cert ${EC2\_CERT} --privatekey ${EC2\_PRIVATE\_KEY} --user **42** --ec2cert ${EUCALYPTUS\_CERT}"

alias ec2-upload-bundle="ec2-upload-bundle -a ${EC2\_ACCESS\_KEY} -s ${EC2\_SECRET\_KEY} --url ${S3\_URL} --ec2cert ${EUCALYPTUS\_CERT}"

export EUCA\_API\_KEY=**a1c47f5a-3b84-4ded-b819-7f9622847b1d**

export EUCA\_USERNAME=**gm2391**

export EUCA\_PROJECT\_ID=**gm2391**

export EUCA\_URL=**http://12.208.178.2:8774/v1.0/**

## Appendix: Getting Started With The Foundry Cloud

Getting Started With The Foundry Cloud

9/15/2011

Overview:

The Foundry Cloud is similar to Amazon EC2 -- you create VM instances based on predefined bootable images, and manage those instances through a variety of tools or a REST API.

Credentials are issued through the Cloud Portal and must be injected into the CLI environment or provided as part of API invocations. The Portal provides basic operational control, but use of a EC2-compatible CLI is the preferred means of full administrative control. The Elasticfox extension for Firefox provides very good functionality and covers most, if not all, common use cases.

VMs initially come up on a completely private network. It is possible to assign an externally routable address to any VM. A suggested architectural pattern is to assign an external IP to just one VM initially and use it as ajump server, since it can route to the rest of the VMs on the private network.Developers can access their VMs securely and then choose to assign public IPs only to those which actually require open network access.

Prerequisites:

For CLI management of instances

. eucatools

http://open.eucalyptus.com/wiki/Euca2oolsGuide

http://open.eucalyptus.com/wiki/Euca2oolsSourceInstall (Mac?)

For browser-based management

. elasticfox

http://www.eaglegenomics.com/wp-content/uploads/2011/06/elasticfox.xpi

It is also possible to use any EC2-compatible CLI or GUI;configuration should be similar to that described for elasticfox below.

Tasks:

1. Obtain cloud credentials

. Connect to the Foundry Cloud portal

https://cloud.foundry.att.com

. Click your user id at top of page

. Select a project from the list of available ones. Note that   
 **"**sandbox**"**instances will be purged without warning after 7 days,   
 so if you don't see any other projects available to you please   
 ask an administrator.

. Click "download credentials file" link.

. Expand downloaded file into a temp directory.

. Move novarc to a working directory (you’ll need to source it   
 later in order to use CLI tools).

2a. Configure CLI environment

. Source novarc (might want to do this in your .profile or other

startup files)

2b. Configure Elasticfox

You will need information from novarc, so have it handy.

. Bring up elasticfox

Tools -> Elasticfox

. Click "Regions"; set up a new region named "foundry" using the

value for EC2\_URL as specified in novarc.

. Click "Credentials"; set up new account named the same as your   
 current project using the values for EC2\_ACCESS\_KEY and   
 EC2\_SECRET\_KEY as specified in novarc. Note that you can set up

distinct accounts for each project you want to use.

. Select the "foundry" Region and appropriate Credentials, you   
 should now be able to manage your instances.

3. Generate a keypair to access your instances via ssh

CLI:

$ euca-add-keypair unique-name > unique-name.pk

Elasticfox:

Click KeyPairs, then the green "key" icon. Download your new

private key.

. Set proper attributes on the saved private key

$ chmod 600 unique-name.pk

Note: Save the generated/downloaded file someplace you won’t lose   
 it, this is the private key you’ll need to access your instances   
 and cannot be recovered.

4. Select an available launch image

CLI:

$ euca-describe-images

Elasticfox:

Click "Images"

5. Launch an instance

CLI:

$ euca-run-instances –k unique-name ami-xxxxxxx

where unique-name is the keypair generated above, and

ami-xxxxxxx is the id of the selected image.

This will return an instance id for later use.

Elasticfox:

Right-click on an image, select "Launch instance(s) of this AMI"

specifying desired Instance Type and KeyPair (don't change the

other fields for now).

5a. Provision sshd on a nonstandard port

In some places it's not possible to ssh directly to instances over   
 the standard port 22. If this is the case in your environment,   
 run sshd on port 80 (which is generally allowed through   
 firewalls). This can be done while launching the instance.

To start a new cloud instance with ssh running on port 80 as well   
 as 22, at the bottom of the Elasticfox dialog past the following   
 into the "User Data" field (or stick it in a file and refer to   
 it):

---

#! /bin/bash

echo "Port 80" >> /etc/ssh/sshd\_config

service ssh restart

---

This will get run on first boot.

6. Check status of an instance

CLI:

$ euca-describe-instances <instance-id>

The INSTANCE record contains IP address assigned to your new

instance; it should support ssh login once its status shows

up as “running”.

Elasticfox:

Click "Instances".

7. Assign an externally routable IP address to an instance

CLI:

$ euca-describe-addresses

Shows addresses available for assignment

$ euca-associate-address -i <instance-id> <ip-address>

Elasticfox:

Click "Elastic IPs", right-click an address,

then select "Associate Elastic IP with Instance".

You may need to click the green plus-sign first to allocate an   
 IP from the pool.

8. SSH to an instance

. $ ssh –l root –i <path to unique-name.pk> <instance-ip-address>

9. Release external IP

CLI:

$ euca-release-address <ip-address>

Elasticfox:

Click "Elastic IPs", right-click an assigned address,

then select "Release this address".

10. Save a runnable image

For now, ask an administrator; this will become self-service in

an upcoming Cloud release.

11. Terminate an instance (losing all its data)

CLI:

$ euca-terminate-instances <instance-id>

Elasticfox:

Click "Instances", right-click an instance, select "Terminate".