FuseSource

Fuse ESB Enterprise

Cloud Computing with Fabric

Version 7.1
December 2012

Integration Everywhere

Cloud Computing with Fabric

Version 7.1

Updated: 08 Jan 2014

Trademark Disclaimer

Third Party Acknowledgements

- HAPI-OSGI-Base Module (http://hl7api.sourceforge.net/hapi-osgi-base/) ca.uhn.hapi:hapi-osgi-base:bundle:1.2

  License: Mozilla Public License 1.1 (http://www.mozilla.org/MPL/MPL-1.1.txt)

Table of Contents

1. Amazon EC2 Tutorial 9
Overview of the Tutorial 10
Open an Amazon EC2 Account 12
Set Up an SSH Key Pair 13
Initialize Your Local Container 16
Register the Amazon EC2 Provider 18
Create a Fabric Server 20
Provision Containers 23
Monitor Instances with JMX 26
Terminate Instances 28

Chapter 1. Amazon EC2 Tutorial

This tutorial explains how to get started in the cloud using Amazon's Elastic Compute Cloud (EC2) and FuseSource's Fabric technology. Fuse Fabric makes it possible to provision containers in the cloud rapidly and easily.

Overview of the Tutorial 10
Open an Amazon EC2 Account 12
Set Up an SSH Key Pair 13
Initialize Your Local Container 16
Register the Amazon EC2 Provider 18
Create a Fabric Server 20
Provision Containers 23
Monitor Instances with JMX 26
Terminate Instances 28

Overview of the Tutorial

Basic technologies

This Amazon EC2 tutorial is based on the following technology stack:

- *Amazon Web Services (AWS) EC2*—Amazon's Elastic Compute Cloud provides puts a variety of hardware and operating systems at your disposal, including Red Hat, Ubuntu, SUSE, and Windows. Operating systems are made available as Amazon Machine Images (AMIs).
- *Red Hat Enterprise Linux O/S*—we use a RHEL 6.0 AMI for this tutorial, which is available in Amazon's free usage tier.
- *JDK 1.6*—the Java software is automatically installed by Fuse Fabric, after a compute instance is created.
- *Red Hat JBoss Fuse OSGi container*—the container runtime is automatically installed by Fuse Fabric.
- *Fuse Fabric*—provides the infrastructure for creating, configuring, and administering a collection of containers in the cloud.
- *Profiles*—a Fuse profile is the natural way to package your applications in the context of Fuse Fabric. A profile consists of a collection of OSGi bundles and Karaf features. When a profile is deployed, the specified components are downloaded from Maven repositories and installed into the container.

JClouds library

JClouds[1] is an open-source library that enables you to administer cloud providers remotely. It provides modules for communicating with a great variety of cloud providers[2]. FuseSource supports only the Amazon and Rackspace providers, however, which can be accessed by installing the following JClouds features in your container:

```
jclouds-aws-ec2
jclouds-cloudservers-us
jclouds-cloudservers-uk
```

Fabric commands

The Fabric commands provide another layer on top of the JClouds library, providing commands that integrate container administration and cloud

[1] http://www.jclouds.org/
[2] http://www.jclouds.org/documentation/reference/supported-providers/

administration. You can get access to the Fabric commands by installing the following Karaf features into your container:

```
fabric-jclouds
fabric-commands
```

Tutorial steps

To get started with provisioning containers in Amazon EC2, perform the following steps:

1. "Open an Amazon EC2 Account" on page 12
2. "Set Up an SSH Key Pair" on page 13
3. "Initialize Your Local Container" on page 16
4. "Register the Amazon EC2 Provider" on page 18
5. "Create a Fabric Server" on page 20
6. "Provision Containers" on page 23
7. "Monitor Instances with JMX" on page 26
8. "Terminate Instances" on page 28

Open an Amazon EC2 Account

Signing up

You can sign up to Amazon Web Services (AWS) by navigating to the following page, http://aws.amazon.com/, and following the sign-up instructions.

Free usage tier

For the first year of membership, Amazon offers a free usage tier. Free usage is subject to a number of restrictions—in particular, you can only use the smallest hardware configuration, `t1.micro` on Linux. For full details of what is allowed in the free usage tier, see: http://aws.amazon.com/free.

Important

You will be charged for any services you use that do not lie within the scope of the free usage tier. Always terminate compute instances to minimize charges.

Set Up an SSH Key Pair

Why do you need an SSH key pair?

After creating a compute instance on the cloud, logging in through the SSH protocol is the primary way of accessing the instance. The SSH protocol uses a key pair for authentication, where the public key is installed in the remote instance and the private key is installed on your local machine.

Creating key pairs in AWS

AWS supports the following approaches to creating SSH key pairs:

- *Download a private key*—in this case, you use the AWS Management Console to generate a key pair and you download the private key to your local machine. AWS keeps a copy of the public key, which is then automatically installed into new compute instances.
- *Upload a public key*—in this case, you generate the SSH key pair yourself and upload the public key to AWS (actually, the Fabric console commands will upload the public key for you).

Advantages of uploading a public key

Uploading a public key has several advantages over downloading a private key:

- It is more secure, because the private key *never* gets sent across the Internet.
- You can easily upload the same public key to instances in different provider regions and even use the same public key with different providers.
- You have more control over the parameters of the key pair, because you generate it yourself.

Default key location

The `ssh` command and the Fabric console commands automatically look for the private key in the default key location. It is recommended that you install your key in the default location, because it saves you the trouble of specifying the location explicitly.

On a *NIX operating system, the default locations for an RSA key pair are:

```
~/.ssh/id_rsa
~/.ssh/id_rsa.pub
```

On a Windows operating system, the default locations for an RSA key pair are:

```
C:\Documents and Settings\Username\.ssh\id_rsa
C:\Documents and Settings\Username\.ssh\id_rsa.pub
```

Note

AWS only supports RSA keys. DSA keys do *not* work.

Creating a new SSH key pair

Generate an RSA key pair using the `ssh-keygen` utility. Open a new command prompt and enter the following command:

```
ssh-keygen -t rsa -b 2048
```

The preceding command generates an RSA key with a key length of 2048 bits. You will then be prompted to specify the file name for the key pair:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/User
name/.ssh/id_rsa):
```

Type return to save the key pair in the default location. You will then be prompted for a pass phrase:

```
Enter passphrase (empty for no passphrase):
```

Type return twice to select no pass phrase. This means that the private key is not encrypted.

Important

The Fabric console commands are not compatible with an encrypted private key. Hence, you must ensure that the private key is not encrypted (to protect the private key file, you should change its permissions to ensure that it can be read only by you).

Troubleshooting

Setting up an SSH key pair can sometimes be tricky. Here are a couple of hints to help with troubleshooting:

- If you get the following error when you try to invoke a JClouds or Fabric console command:

```
IOException: Invalid DER: length field too big (186)
```

It probably means you are using an encrypted private key. The Fabric console commands support only *unencrypted* private keys, however.

- If you are having trouble with a particular SSH key pair and you want to try a different one, you must first unregister the cloud provider using the `fabric:cloud-provider-remove` console command, and then re-register the cloud provider using the `fabric:cloud-provider-add` console command (this forces Fabric to reread the SSH private key).
- Make sure you are using an RSA key. DSA format is not supported.

Initialize Your Local Container

Prerequisites

This tutorial assumes you have a fresh installation of Fuse ESB Enterprise on your local machine (see Installation Guide). If you have previously experimented with the local container instance, it might be a good idea to perform a cold start before you proceed.

Forcing a cold start

(Optional) If you are not sure what state your local container is in or if you have previously used your local container in the context of a fabric, it is a good idea to force a *cold start* before you continue. This ensures that your container is in a known state (a plain, unmanaged container) and makes it more likely that you will be able to follow the tutorial steps without any problems.

To force a cold start, perform the following steps:

1. If the container is currently running, shut it down by entering the following console command:

   ```
   karaf@root> shutdown -f
   ```

   Note

   If your container has any child instances running, you must also shut the child instances down. On *NIX systems, you could use the command `ps -ef | grep karaf` to discover any child instances.

2. Delete the *`ESBInstallDir`*`/data/` directory.

   Warning

   This will completely wipe the state of the container (apart from the configuration settings stored under *`ESBInstallDir`*`/etc/`).

   Note

   If your container has any child instances, you must also delete the *`ESBInstallDir`*`/instances/` directory, which contains the data for the child instances.

3. Start the container by entering the following commands:

```
cd ESBInstallDir/bin
./fuseesb
```

Initializing the local container

After a cold start, the container does not have the required cloud commands installed by default. To enable support for accessing and administering the AWS EC2 provider, install the requisite Karaf features, as follows:

```
karaf@root> features:install jclouds-aws-ec2 fabric-jclouds
fabric-commands
```

Register the Amazon EC2 Provider

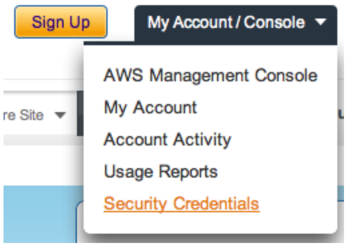
What is the purpose of registering the provider?

Instead of administering your compute instances directly through the AWS Console, Amazon provides a Web service that enables you to administer your compute instances remotely (for example, using the JClouds utilities). Registering the Amazon EC2 provider consists essentially of caching your AWS login credentials in the container, so that the JClouds library and the Fabric console commands can administer the provider.

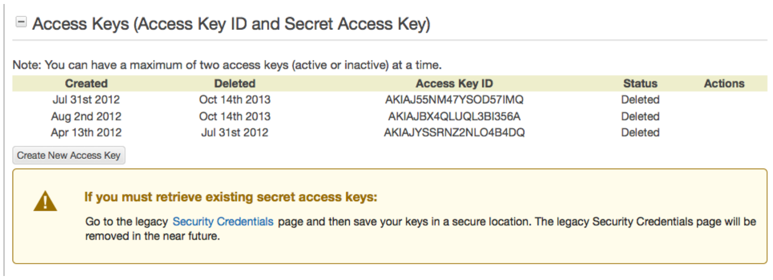
Create a new Access Key

If you do not already have an AWS Access Key (which consists of an *Access Key ID* and a *Secret Access Key*), create a new Access Key as follows:

1. Use your Web browser to navigate to http://aws.amazon.com/
2. Move your mouse to the **My Account/Console** drop-down menu and select the **Security Credentials** menu item.



3. If you are not already logged on to AWS, you are now prompted to log on using the account details you specified in "Open an Amazon EC2 Account" on page 12.
4. Expand the **Access Keys** section and click **Create New Access Key**.


Access Keys (Access Key ID and Secret Access Key)

Note: You can have a maximum of two access keys (active or inactive) at a time.

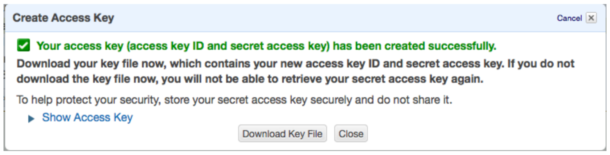
Created	Deleted	Access Key ID	Status	Actions
Jul 31st 2012	Oct 14th 2013	AKIAJ55NM47YSOD57IMQ	Deleted	
Aug 2nd 2012	Oct 14th 2013	AKIAJBX4QLUQL3BI356A	Deleted	
Apr 13th 2012	Jul 31st 2012	AKIAJYSSRNZ2NLO4B4DQ	Deleted	

Create New Access Key

If you must retrieve existing secret access keys:

Go to the legacy Security Credentials page and then save your keys in a secure location. The legacy Security Credentials page will be removed in the near future.

5. The **Create Access Key Dialog** appears. Click **Download Key File** and save the downloaded file, `rootkey.csv`, to a secure location on your filesystem.


Create Access Key

Your access key (access key ID and secret access key) has been created successfully.

Download your key file now, which contains your new access key ID and secret access key. If you do not download the key file now, you will not be able to retrieve your secret access key again.

To help protect your security, store your secret access key securely and do not share it.

Note

This is the only chance you get to download the Secret Access Key from AWS. If you lose the key at this point, you cannot retrieve it later (you would then need to create a new Access Key).

Registering the Amazon EC2 provider

To register the Amazon EC2 provider, perform the following steps:

1. Open the file containing your Access Key (usually named `rootkey.csv`), so that you can read the Access Key ID an the Secret Access Key for your Amazon account.

2. In your local container, register the Amazon EC2 provider by entering the following console command:

```
karaf@root> fabric:cloud-service-add --name aws-ec2 --pro
vider aws-ec2
--identity AccessKeyID --credential SecretAccessKey
```

   This command can take a minute or two to complete. If it is successful, you should see some output like the following:

```
Waiting for aws-ec2 service to initialize.
Fabric has not been initialized. Provider registration will
 be local to the current container.
```

Create a Fabric Server

What is a Fabric Server?

The first step in creating a new fabric in the cloud is to create a Fabric Server instance. A Fabric Server is a container with a Fabric Registry Agent deployed inside it (there must be at least one Fabric Server in a fabric). You can think of it as a seed for the rest of the fabric.

What happens when you create a Fabric Server?

The `fabric:container-create-cloud` console command, which creates the Fabric Server, is a powerful command that does a lot of things:

1. Using the JClouds library, it connects to the AWS and creates a new compute instance in the cloud.
2. It uploads your SSH public key to the compute instance.
3. It installs all of the basic prerequisites for provisioning and running an Fuse ESB Enterprise container in the compute instance (for example, by installing JDK 1.6).
4. It installs Fuse ESB Enterprise in the compute instance.
5. It starts up the Fuse ESB Enterprise container and deploys a Fabric Registry Agent into the container, so that it functions as a Fabric Server.
6. It modifies the EC2 firewall settings for the compute instance (if necessary, creating a new EC2 security group).

Creating a Fabric Server on a micro instance

To create a Fabric Server on a micro instance, `t1.micro` (in Amazon's free usage tier) using a Red Hat Enterprise Linux (RHEL) 6.0 image, enter the following console command:

```
JBossFuse:karaf@root> fabric:container-create-cloud --name
aws-ec2 --ensemble-server
--new-user admin --new-user-password admin --zookeeper-password
 admin
--hardwareId t1.micro --os-family rhel --os-version 6.0 re
gistry
```

This command creates a container called `registry` and the `--ensemble-server` option makes the new container a Fabric Server. This command can take a minute or two to complete. If it is successfully, you should see output like the following:

```
Looking up for compute service.
Creating 1 nodes in the cloud. Using operating system: rhel.
 It may take a while ...
Node fabric-24be8a5c has been created.
Configuring firewall.
Installing fabric agent on container registry. It may take a
 while...
Overriding resolver to publichostname.
                  [id] [container]                    [public
 addresses]             [status]
  us-east-1/i-24be8a5c registry
[50.19.18.91]                    success
```

Creating a Fabric Server on a small instance

The amount of memory available in a micro instance is a bit limited for a Fuse ESB Enterprise container instance. If you want to create a fabric for a production deployment, it is typically better to use at least a small instance, `m1.small`. You can create a small instance, as follows:

```
karaf@root> fabric:container-create-cloud --name aws-ec2 --
ensemble-server
--new-user admin --new-user-password admin --zookeeper-password
 admin
--hardwareId m1.small --os-family rhel --os-version 6.0 re
gistry
```

Important

An `m1.small` instance is not covered by the Amazon free usage tier and incurs an hourly usage charge (until the instance is terminated).

SSH login to the new instance

It should now be possible to log on to the new compute instance using SSH. Open a new command prompt and enter the following command:

```
ssh 50.19.18.91
```

Where you provide the compute instance's public IP address as the argument to the `ssh` command (you could also use the compute instance's public hostname). The ssh command uses the default RSA private key as credentials.

If you did not put your SSH key pair in the default location, you can specify the private key location explicitly, as follows:

```
ssh -i PrivateKeyFile 50.19.18.91
```

Hardware list

If you want to see all of the hardware options available, use the `jclouds:hardware-list` console command:

```
karaf@root> jclouds:hardware-list
[id]                     [cpu]  [cores]   [ram]
cc1.4xlarge               32.0      8.0  23552.0
cg1.4xlarge               32.0      8.0  22528.0
cc2.8xlarge               88.0     16.0  61952.0
t1.micro                   1.0      1.0   630.0
c1.medium                  5.0      2.0  1740.0
c1.xlarge                 20.0      8.0  7168.0
m1.large                   4.0      2.0  7680.0
m1.small                   1.0      1.0  1740.0
m1.medium                  2.0      1.0  3750.0
m1.xlarge                  8.0      4.0  15360.0
m2.xlarge                  6.5      2.0  17510.0
m2.2xlarge                13.0      4.0  35020.0
m2.4xlarge                26.0      8.0  70041.0
```

Image list

If you want to find out what machine images are available on Amazon EC2, enter the following console command:

```
karaf@root> jclouds:image-list
```

Groups and security

Fabric automatically configures an Amazon EC2 security group for the new containers. All of the containers in a fabric should belong to the same security group. By default, Fabric adds all of the cloud containers to the `fabric` group. If necessary, you can specify the group explicitly by specifying the `--group` option on the `fabric:container-create-cloud` command.

Provision Containers

Provisioning containers

All of the prerequisites are now in place for provisioning Fuse ESB Enterprise containers in the cloud. After joining your local container to the fabric (which enables you to administer the fabric remotely), you can provision a new container in the cloud by entering a single console command.

Join the local container to the fabric

To enable administration of the new fabric, first join your local container to the fabric. This makes it possible to administer the fabric by entering console commands in your local container. Invoke the `join` command as follows:

```
karaf@root> fabric:join -n --zookeeper-password admin
50.19.18.91
```

Where `50.19.18.91` is the public IP address of the registry container in the cloud (alternatively, you could use the public hostname here). The remote Fabric Server's Zookeeper password must be provided, in order to join the current container to the fabric.

Important

Don't forget to include the `-n` option, which ensures that the local container remains a non-managed container (if you forget the `-n` option, the local container will be re-provisioned automatically and you will lose access to the JClouds and Fabric console commands).

To check that the join has been successful, try listing the containers in the fabric, as follows:

```
karaf@root> fabric:container-list
[id]                          [version] [alive] [profiles]
                   [provision status]
registry                      1.0       true    fabric, fab
ric-ensemble-0000-1 success
root*                         1.0       true    fabric
```

Create container and deploy profile in one step

If you already know what profiles you want to deploy in the new containers, the most efficient approach is to create the compute instances and specify the profiles all in the same command.

For example, to create two new compute instances as part of the current fabric and to deploy the profiles `mq` and `fabric` into each new container, enter the following console command:

```
JBossFuse:karaf@root> fabric:container-create-cloud --name
aws-ec2
--hardwareId t1.micro --os-family rhel --os-version 6.0
--profile mq --profile fabric mqserver 2
```

Note how you can deploy multiple profiles, by specifying the `--provider` option multiple times. This command produces output like the following:

```
Looking up for compute service.
Creating 2 nodes in the cloud. Using operating system: rhel. It may take a while ...
Node fabric-3ea09446 has been created.
Node fabric-3ca09444 has been created.
Configuring firewall.
Configuring firewall.
Installing fabric agent on container mqserver1. It may take a while...
Installing fabric agent on container mqserver2. It may take a while...
Overriding resolver to publichostname.
Overriding resolver to publichostname.
                 [id] [container]                   [public addresses]            [status]

  us-east-1/i-3ca09444 mqserver2                     [50.19.15.210]                 success

  us-east-1/i-3ea09446 mqserver1                     [184.72.214.117]               success
```

Create container and deploy profile in two steps

You can create containers and deploy profiles in separate steps, as follows:

1. Create two new Fabric Containers, `mqserver1` and `mqserver2`, as follows:

```
JBossFuse:karaf@root> fabric:container-create-cloud --name
 aws-ec2
--hardwareId t1.micro --os-family rhel --os-version 6.0
mqserver 2
```

2. Deploy the profiles `mq` and `fabric` into each of the new Fabric Containers, as follows:

```
karaf@root> fabric:container-change-profile mqserver1 mq
fabric
```

```
karaf@root> fabric:container-change-profile mqserver2 mq
fabric
```

Check the provision status

After creating and deploying, you can test the provision status of the new containers using the `fabric:container-list` command, as follows:

```
karaf@root> fabric:container-list
[id]                          [version] [alive] [profiles]                    [provision
 status]
mqserver1                     1.0       true    fabric, mq                    success
mqserver2                     1.0       true    fabric, mq                    success
registry                      1.0       true    fabric, fabric-ensemble-0000-1 success
root*                         1.0       true    fabric
```

Monitor Instances with JMX

Monitoring with JConsole

All of the containers in the fabric have JMX enabled by default. To monitor a Fabric Container with JConsole, proceed as follows:

1. In the local container, use the `fabric:container-list` console command to obtain the JMX connection URLs for all of the containers, as follows:

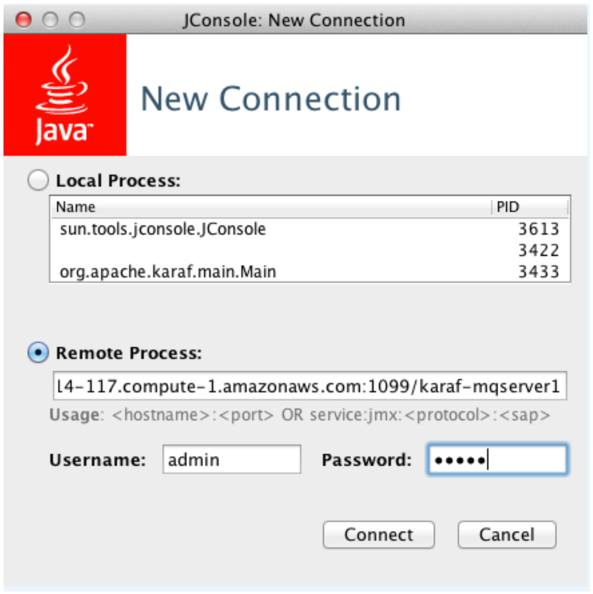
```
karaf@root> fabric:container-list -v
[id]                 [version] [alive] [profiles]                       [ssh url]

 [jmx url]
  [provision status]
mqserver1            1.0       true    mq                               ec2-184-72-214-
117.compute-1.amazonaws.com:8101
 service:jmx:rmi://ec2-184-72-214-117.compute-1.amazonaws.com:44444/jndi/rmi://ec2-184-72-
214-117.compute-1.amazonaws.com:1099/karaf-mqserver1 success
mqserver2            1.0       true    mq                               ec2-50-19-15-
210.compute-1.amazonaws.com:8101
 service:jmx:rmi://ec2-50-19-15-210.compute-1.amazonaws.com:44444/jndi/rmi://ec2-50-19-15-
210.compute-1.amazonaws.com:1099/karaf-mqserver2 success
registry             1.0       true    fabric, fabric-ensemble-0000-1  ec2-50-19-18-
91.compute-1.amazonaws.com:8101
 service:jmx:rmi://ec2-50-19-18-91.compute-1.amazonaws.com:44444/jndi/rmi://ec2-50-19-18-
91.compute-1.amazonaws.com:1099/karaf-registry success
Error executing command: java.lang.RuntimeException: org.apache.zookeeper.KeeperExcep
tion$NoNodeException: KeeperErrorCode = NoNode for /fabric/registry/containers/config/root/pub
lichostname
```

2. In an O/S command prompt, enter the following command to start up JConsole:

   ```
   > jconsole
   ```

3. Copy one of the (rather long) JMX connection URLs from step 1 and paste it into the **Remote Console** text field of the **JConsole: New Connection** dialog. In the **Username** field, type `admin`; and in the **Password** field, type `admin`. Click **Connect**.



Local Process:

Name	PID
sun.tools.jconsole.JConsole	3613
	3422
org.apache.karaf.main.Main	3433

Remote Process: [✓]

l4-117.compute-1.amazonaws.com:1099/karaf-mqserver1

Usage: <hostname>:<port> OR service:jmx:<protocol>:<sap>

Username: admin Password: •••••

4. JConsole now establishes a JMX connection to the cloud container. Click on the **MBeans** tab to explore the status of applications running in the container.

EC2 firewall configuration

Fabric automatically configures the EC2 firewall to let JMX connections through. It should *not* be necessary to make any modifications to the firewall configuration.

Terminate Instances

Terminating EC2 instances

Terminating an EC2 compute instance means that the instance is completely destroyed and the hardware resources are returned to the pool. In contrast, stopping an instance (which is possible for some machine images) merely shuts down the processes running on the instance, preserving any data and software on the instance.

You can use either of the following approaches to terminate instances on Amazon EC2:

- "Terminate individual instances" on page 28
- "Terminate all containers" on page 28

Important

You should always terminate instances when you are finished with them, in order to minimize usage charges. Stopping an instance is *not* the same as terminating an instance

Terminate individual instances

The cleanest way to terminate an individual compute instance is to use the `fabric:container-delete` console command—for example:

```
karaf@root> fabric:container-delete mqserver2
```

The advantage of this command is that all of the corresponding entries in the Fabric Registry are cleaned up, before the container's compute instance is terminated.

Terminate all containers

You can quickly terminate *all* of your compute instances on the EC2 cloud using the `jclouds:node-destroy-all` command, as follows:

```
karaf@root> jclouds:node-destroy-all
```

Note

This example assumes that your entire fabric is hosted in the cloud, so that you do not need to worry about the consistency of registry entries when you delete it. This approach would *not* be appropriate, however, if you have a fabric that overlaps with host machines outside the cloud.