Amazon Elastic Compute Cloud

User Guide

API Version 2011-12-01

Amazon Elastic Compute Cloud: User Guide

Table of Contents

Welcome

This is the User Guide for Amazon Elastic Compute Cloud (Amazon EC2). This guide picks up where the Amazon Elastic Compute Cloud Getting Started Guide leaves off.

Note

The content from the *Amazon Elastic Compute Cloud Developer Guide* is now included here in the user guide. This user guide helps you understand the infrastructure components that Amazon EC2 provides and how to use them. The guide shows you how to access Amazon EC2 with a web-based GUI, with command line tools, and programmatically through the Amazon EC2 API.

Amazon EC2 is a web service that provides resizeable computing capacity—literally server instances in Amazon's data centers—that you use to build and host your software systems. With Amazon EC2, you can get access to infrastructure resources using APIs or web tools and utilities.

How Do I...?

How Do I?	Relevant Topics
Get a general product overview and information about pricing	Amazon EC2 product page
Get a quick hands-on introduction to Amazon EC2	Amazon Elastic Compute Cloud Getting Started Guide
Get a quick summary of the basic infrastructure components that Amazon EC2 provides	Introduction to Amazon EC2 (p. 3)
Get detailed information about how to use the Amazon EC2 components and features, with instructions for several different interfaces	Using Amazon EC2 (p. 14)
Find available libraries for programmatically accessing Amazon EC2	Available Libraries (p. 403)
Get started using the command line tools (i.e., the Amazon EC2 API tools)	Setting Up the Tools (p. 386)

How Do I?	Relevant Topics
Get started using the Query or SOAP API for Amazon EC2	Making API Requests (p. 396)

Introduction to Amazon EC2

Topics

What Is EC2?

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizeable computing capacity—literally, server instances in Amazon's data centers—that you use to build and host your software systems. You can get access to the infrastructure resources that EC2 provides by using APIs, or web tools and utilities.

With EC2, you use and pay for only the capacity that you need. This eliminates the need to make large and expensive hardware purchases, reduces the need to forecast traffic, and enables you to automatically scale your IT resources to deal with changes in requirements or spikes in popularity related to your application or service.

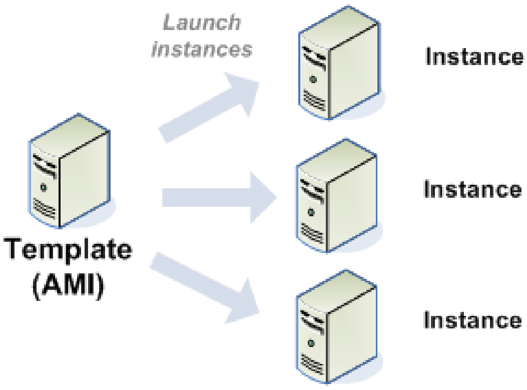
Basic Infrastructure Components

Topics

You might be considering creating a new application to run in the cloud, or moving an existing application from your own servers into the cloud. To do either, you should understand the infrastructure available in the cloud and how it's similar or different from your own data centers. This section gives a brief description of the main components that EC2 provides.

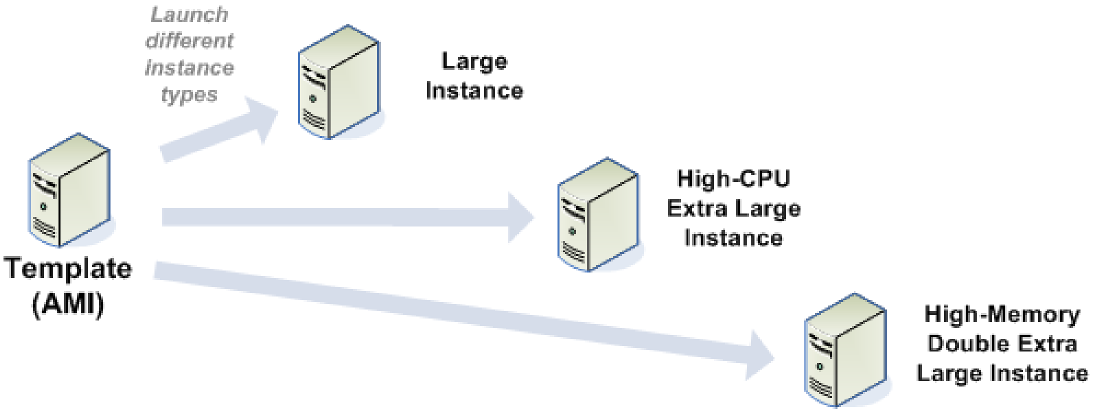
Amazon Machine Images and Instances

An *Amazon Machine Image (AMI)* is a template that contains a software configuration (e.g., operating system, application server, and applications). From an AMI, you launch *instances*, which are running copies of the AMI. You can launch multiple instances of an AMI, as shown in the following figure.



Your instances keep running until you stop or terminate them, or until they fail. If an instance fails, you can launch a new one from the AMI.

You can use a single AMI or multiple AMIs depending on your needs. From a single AMI, you can launch different *types of instances*. An instance type is essentially a hardware archetype. As illustrated in the following figure, you select a particular instance type based on the amount of memory and computing power you need for the application or software that you plan to run on the instance. For more information about the available instance types, see Instance Families and Types (p. 69).


Launch different instance types
Large Instance
High-CPU Extra Large Instance
Template (AMI)
High-Memory Double Extra Large Instance

Amazon publishes many AMIs that contain common software configurations for public use. In addition, members of the AWS developer community have published their own custom AMIs. For more information, go to Amazon Machine Images (AMIs).

You might only need to use AMIs that Amazon or other reputable sources provide, and you can simply customize the resulting instances (e.g., run a script) to provide the data or software you need each time you launch an instance. You can also create your own custom AMI or AMIs; then you can run your application by launching one of your custom AMIs.

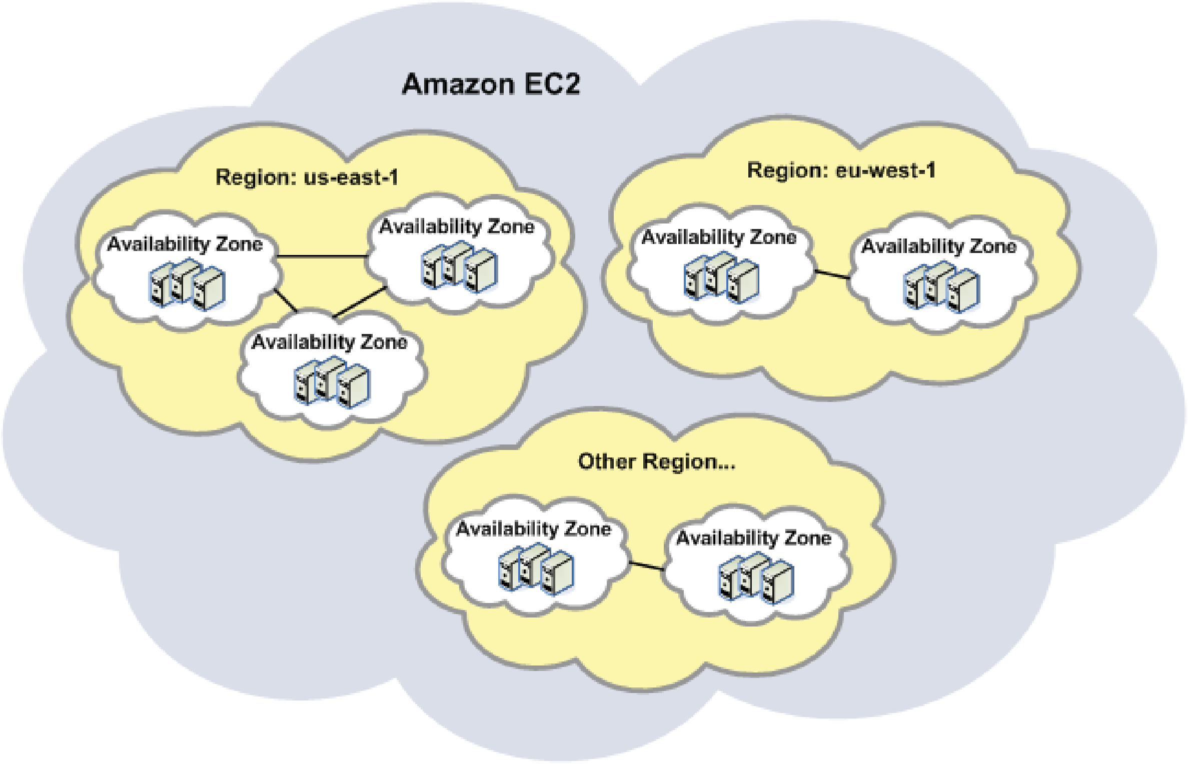
For example, if your application is a web site or web service, your AMI could be preconfigured with a web server, the associated static content, and the code for all dynamic pages. Alternatively, you could configure your AMI to install all required software components and content itself by running a bootstrap script as soon as the instance starts. As a result, after launching the AMI, your web server will start and your application can begin accepting requests.

For information about AMIs and instances, see Using AMIs (p. 15) and Using Instances (p. 69).

Regions and Availability Zones

Amazon has data centers in different areas of the world (e.g., North America, Europe, Asia, etc.). Correspondingly, EC2 is available to use in different *Regions*. By launching instances in separate Regions, you can design your application to be closer to specific customers or to meet legal or other requirements. Prices for Amazon EC2 usage vary by Region (for more information about pricing by Region, go to the Amazon EC2 Pricing page).

Each Region contains multiple distinct locations called *Availability Zones* (illustrated in the following diagram). Each Availability Zone is engineered to be isolated from failures in other Availability zones and to provide inexpensive, low-latency network connectivity to other zones in the same Region. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.



For more information about the available Regions and Availability Zones, see Using Regions and Availability Zones (p. 321).

Storage

When using EC2, you might have data that you need to store. The two most commonly used storage types are:

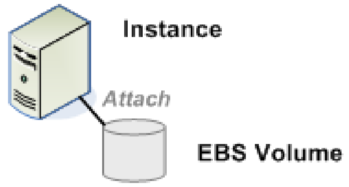
- Amazon Simple Storage Service (Amazon S3)
- Amazon Elastic Block Store (Amazon EBS) volumes

Amazon S3

Amazon S3 is storage for the Internet. It provides a simple web service interface that enables you to store and retrieve any amount of data from anywhere on the web. For more information about Amazon S3, go to the Amazon S3 product page.

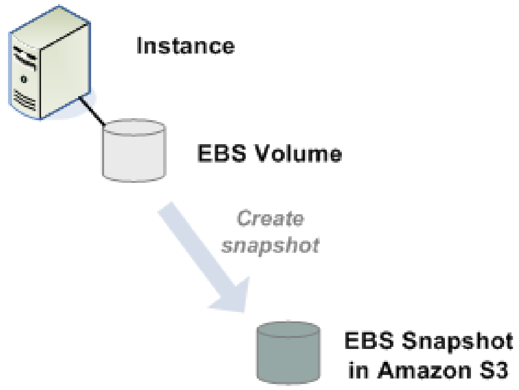
Amazon EBS Volumes

Amazon EBS provides your instances with persistent, block-level storage. Amazon EBS volumes are essentially hard disks that you can attach to a running instance.

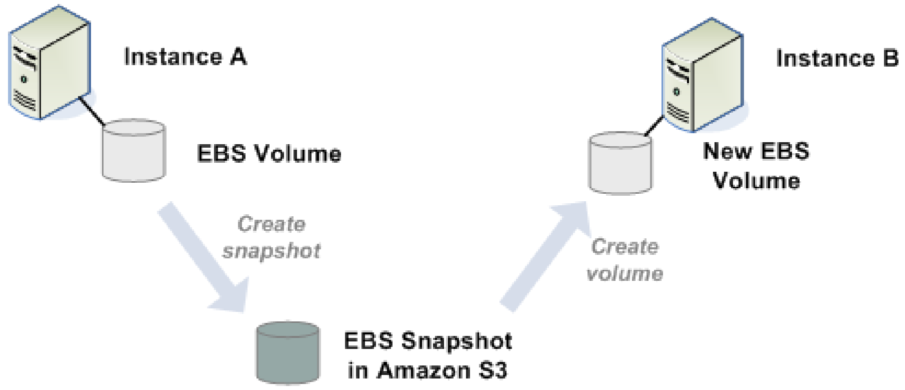


Volumes are especially suited for applications that require a database, a file system, or access to raw block-level storage.

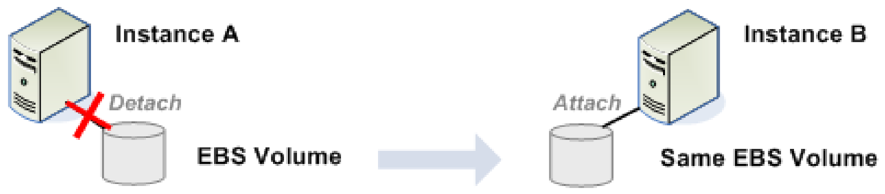
You can attach multiple volumes to an instance. To keep a back-up copy, you can create a snapshot of the volume. As illustrated in the following figure, snapshots are stored in Amazon S3.



You can create a new Amazon EBS volume from a snapshot, and attach it to another instance, as illustrated in the following figure.



You can also detach a volume from an instance and attach it to a different one, as illustrated in the following figure.



For more information about Amazon EBS volumes, see Amazon Elastic Block Store (p. 126).

Ephemeral Storage

Some instance types offer *ephemeral storage*, also referred to as instance store. This is storage that does not persist if the instance is stopped or terminated. For more information, see Amazon EC2 Instance Storage (p. 151).

Root Device Storage

When EC2 was first introduced, all AMIs were *backed by Amazon EC2 instance store*, which means the root device for an instance launched from the AMI is stored in instance store. After we introduced Amazon EBS, we also introduced AMIs that are *backed by Amazon EBS*, which means the root device for an instance launched from the AMI is an Amazon EBS volume. The description of an AMI includes which type it is (you'll see the root device referred to in some places as either *ebs* (for Amazon EBS-backed) or *instance store* (for Amazon instance store-backed). This is important because there are significant differences in what you can do with each type of AMI. For a discussion of these differences, see Using Amazon EBS-Backed AMIs and Instances (p. 177).

Databases

The application you're running on EC2 might need a database. Following are two common ways to implement a database for your application:

- Use Amazon Relational Database Service (Amazon RDS), which enables you to easily get a managed relational database in the cloud
- Launch an instance of a database AMI, and use that EC2 instance as the database

Amazon RDS offers the advantage of handling your database management tasks, such as patching the software, backing up and storing the backups, etc. For more information about Amazon RDS, go to the Amazon RDS product page.

Networking and Security

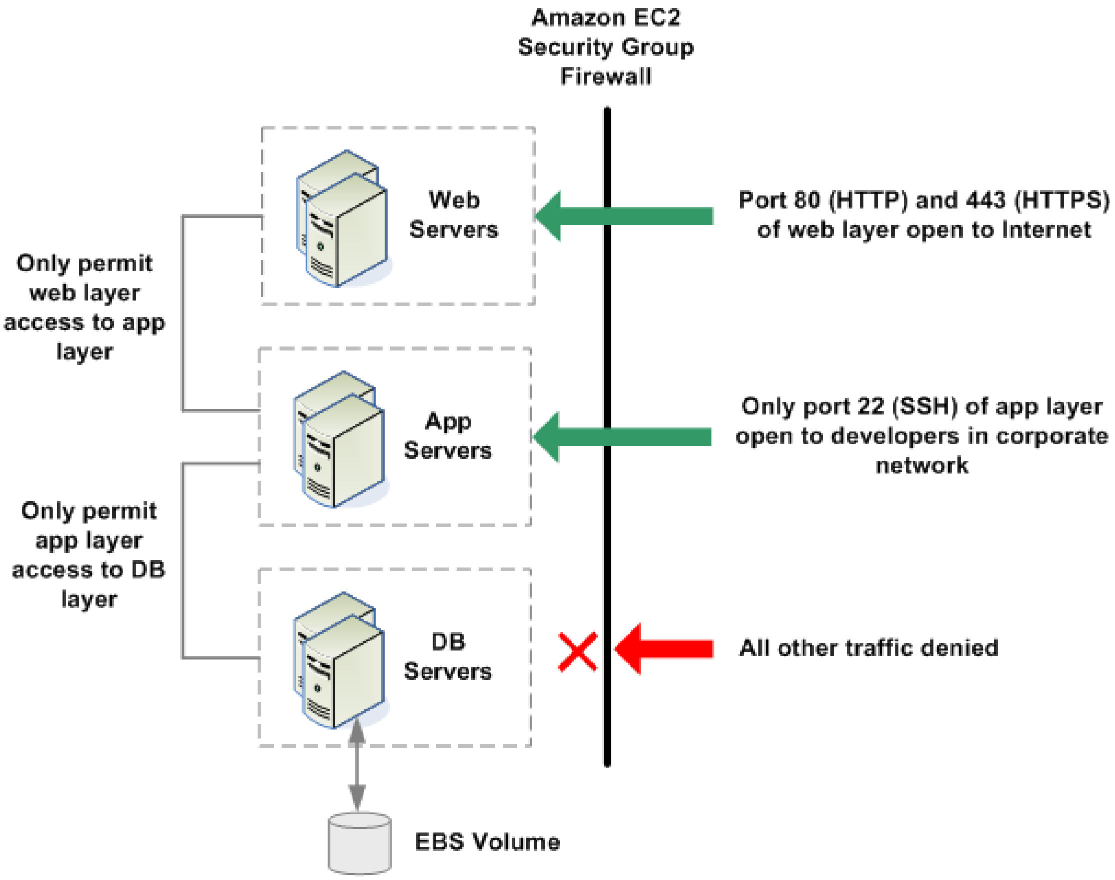
Each instance is launched into the Amazon EC2 network space and assigned a public IP address. Instances can fail or terminate for reasons outside of your control. If one fails and you launch a replacement instance, the replacement will have a different public IP address than the original. However, your application might need a static IP address. Amazon EC2 offers *elastic IP addresses* for those situations. For more information, see Using Instance IP Addresses (p. 289).

You use *security groups* to control who can access your instances. These are analogous to an inbound network firewall that allows you to specify the protocols, ports, and source IP ranges that are allowed to reach your instances. You can create multiple security groups and assign different rules to each group. You can then assign each instance to one or more security groups, and we use the rules to determine which traffic is allowed in to the instance. You can configure a security group so that only specific IP addresses or specific security groups have access to the instance.

The following figure shows a basic three-tier web-hosting architecture running on Amazon EC2 instances. Each layer has a different security group (indicated by the dotted line around each set of instances). The security group for the web servers only allows access from hosts over TCP on ports 80 and 443 (HTTP and HTTPS) and from instances in the *App Servers* security group on port 22 (SSH) for direct host management.

The security group for the app servers allows access from the *Web Servers* security group for web requests, and from your corporate subnet over TCP on port 22 (SSH) for direct host management. Your support engineers could log directly into the application servers from the corporate network, and then access the other instances from the application server boxes.

The *DB Servers* security group permits only the App Servers security group to access the database servers.



For more information about security groups, see Using Security Groups (p. 304).

Monitoring, Auto Scaling, and Load Balancing

AWS provides several features that enable you to do the following:

- Monitor basic statistics for your instances and Amazon EBS volumes
  For more information, see Monitoring Your Instances and Volumes with CloudWatch (p. 348).
- Automatically scale your EC2 capacity up or down according to conditions you define
  For more information, go to the Auto Scaling Developer Guide.

- Automatically distribute incoming application traffic across multiple EC2 instances
  For more information, go to the Elastic Load Balancing Developer Guide.

AWS Identity and Access Management

Amazon EC2 integrates with AWS Identity and Access Management (IAM), a service that lets your organization do the following:

- Create users and groups under your organization's AWS account
- Easily share your AWS account resources between the users in the account
- Assign unique security credentials to each user
- Granularly control users access to services and resources
- Get a single AWS bill for all users under the AWS account

For example, you can use IAM with Amazon EC2 to control which users under your AWS account can create AMIs or launch instances.

For general information about IAM, go to:

- Identity and Access Management (IAM)
- AWS Identity and Access Management Getting Started Guide
- Using AWS Identity and Access Management

For specific information about how you can control User access to Amazon EC2, go to Integrating with Other AWS Products in *Using AWS Identity and Access Management*.

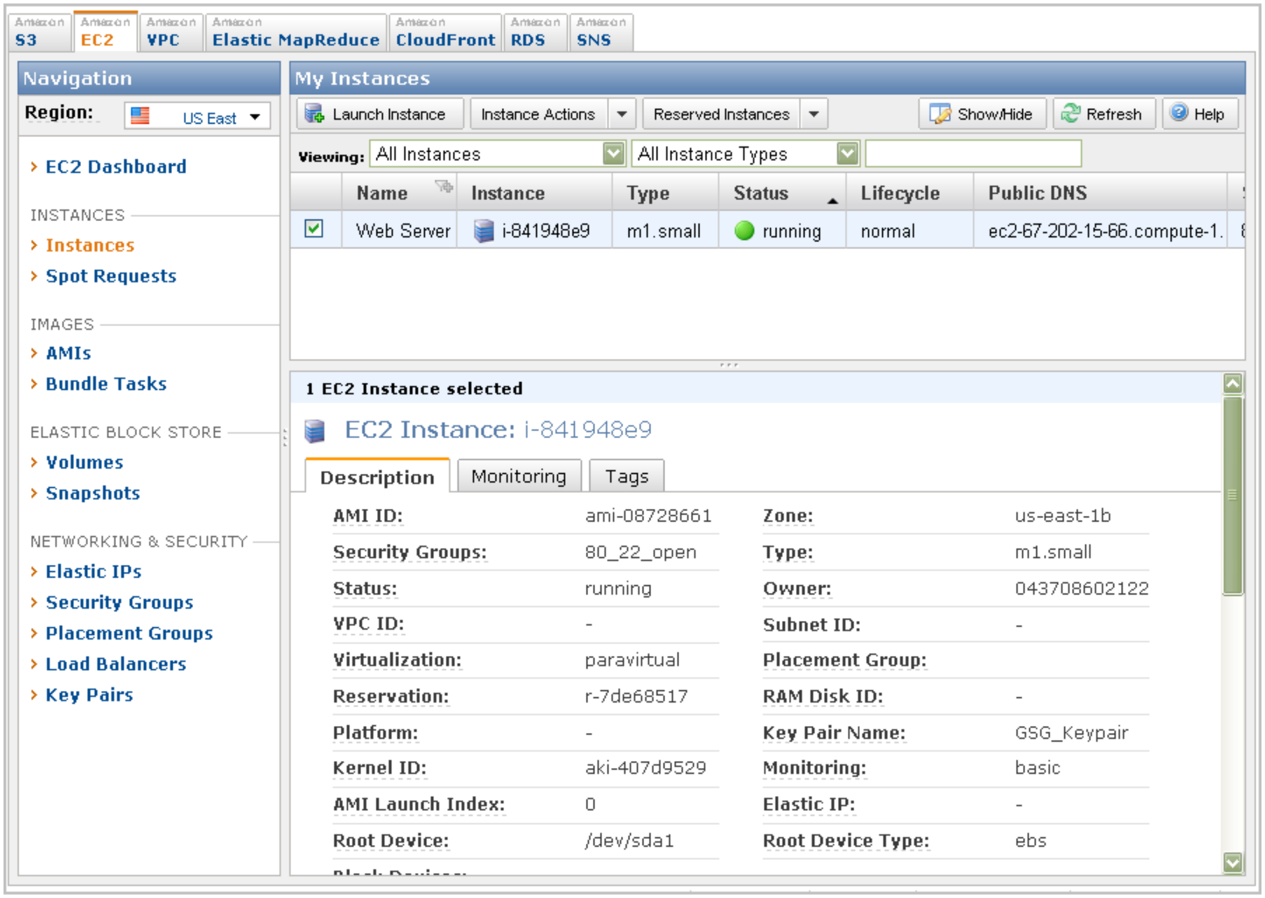
Available Interfaces

Topics

AWS provides different interfaces to access EC2.

AWS Management Console

The AWS Management Console is a simple web-based GUI (see the following screenshot). For more information about using the console, go to the Amazon Elastic Compute Cloud Getting Started Guide.



	Name	Instance	Type	Status	Lifecycle	Public DNS
☑	Web Server	i-841948e9	m1.small	● running	normal	ec2-67-202-15-66.compute-1.

AMI ID:	ami-08728661	Zone:	us-east-1b
Security Groups:	80_22_open	Type:	m1.small
Status:	running	Owner:	043708602122
VPC ID:	-	Subnet ID:	-
Virtualization:	paravirtual	Placement Group:	
Reservation:	r-7de68517	RAM Disk ID:	-
Platform:	-	Key Pair Name:	GSG_Keypair
Kernel ID:	aki-407d9529	Monitoring:	basic
AMI Launch Index:	0	Elastic IP:	-
Root Device:	/dev/sda1	Root Device Type:	ebs

Command Line Tools (API Tools)

EC2 provides a Java-based command-line client that wraps the EC2 SOAP API. For more information, see Getting Started with the Command Line Tools (p. 384).

Programmatic Interface

The following table lists how you can access EC2 programmatically.

Type of Access	Description
AWS SDKs	AWS provides the following SDKs: • AWS SDK for Java • AWS SDK for .NET • AWS SDK for PHP

Type of Access	Description
Third-Party Libraries	Developers in the AWS developer community also provide their own libraries, which you can find at the following AWS developer centers: • AWS Java Developer Center • AWS PHP Developer Center • AWS Python Developer Center • AWS Ruby Developer Center • AWS Windows and .NET Developer Center
EC2 API	If you prefer, you can code directly to the EC2 API (Query or SOAP). For more information, see Making API Requests (p. 396), and go to Amazon Elastic Compute Cloud API Reference.

How You're Charged for EC2

With EC2, you pay for only what you use, and there's no minimum charge. Your charges are broken down into these general parts:

- Instance usage
- Data transfer
- Storage

For a complete list of charges and specific prices, go to the Amazon EC2 pricing page.

To see your bill, go to your AWS Account Activity page. Your bill contains links to usage reports that provide details about your bill.

What's Next?

This section introduced you to the basic infrastructure components that EC2 offers. What should you do next?

Get a Hands-On Introduction to EC2

If you haven't walked through the Amazon Elastic Compute Cloud Getting Started Guide, we recommend you do that next. In that guide, you'll do a quick hands-on exercise where you launch an instance and connect to it.

Understand Differences Between the Cloud and Your Data Center

You need to understand the key differences between running your application on infrastructure in the cloud versus on servers in your own data center. For more information, go to the technical whitepaper: Architecting for the Cloud: Best Practices.

Start Thinking about Instance Failure and Fault Tolerance

It's important for you to design your application to handle the failure of a cloud infrastructure component. For example, EC2 instances will eventually fail; it's just a matter of when. An instance failure isn't a problem if your application is prepared for it. For more information, go to the technical whitepaper: Building Fault-Tolerant Applications on AWS.

For a complete list of the AWS whitepapers, go to the AWS Cloud Computing Whitepapers page.

Learn More about EC2

The next section in this guide (Using Amazon EC2 (p. 14)) describes in more detail the technical aspects of the infrastructure components that were briefly described in the preceding sections. We recommend you understand how these components work before designing your application or service to run on EC2.

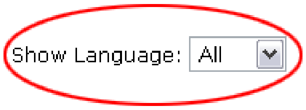
Using Amazon EC2

Topics

This section expands on the basic concepts presented in the preceding section (see Introduction to Amazon EC2 (p. 3)), and includes procedures for using the EC2 components and features. This section also gives important details about some special types of instances (e.g., micro instances, cluster compute instances).

The procedures in this section include instructions for the AWS Management Console, the command line tools (i.e., the API tools), and the EC2 Query API. In the HTML version of this document, you can show instructions for a single interface. There is an interface selection menu in the upper-right corner of pages.

Select your interface of choice to hide all others, or select All to show the instructions in all available interfaces.



Show Language: All

For an introduction to using the AWS Management Console, go to the Amazon Elastic Compute Cloud Getting Started Guide.

For information about setting up the command line tools, see Setting Up the Tools (p. 386). For detailed information about each command, go to the Amazon Elastic Compute Cloud Command Line Reference.

For information about using the EC2 API, see Making API Requests (p. 396). For detailed information about each API action, go to the Amazon Elastic Compute Cloud API Reference.

Using AMIs

Topics

This section gives information about AMIs, the basic building blocks of Amazon EC2. Before accomplishing anything with Amazon EC2, you must understand the concepts in this section.

AMI Basics

An Amazon Machine Image (AMI) contains all information necessary to boot instances of your software. For example, an AMI might contain all the software to act as a web server (e.g., Linux, Apache, and your web site) or it might contain all the software to act as a Hadoop node (e.g., Linux, Hadoop, and a custom application). You launch one or more instances of an AMI. An instance might be one web server within a web server cluster or one Hadoop node.

Available AMIs

A large selection of public AMIs is available from Amazon and the Amazon EC2 community. For more information, go to Amazon Machine Images (AMIs) at the AWS web site.

The AWS Management Console (at http://aws.amazon.com/console) enables you to search for AMIs that meet specific criteria, and then launch instances of those AMIs. For example, you can view the AMIs Amazon has provided, or the AMIs the EC2 community has provided, or AMIs that use certain platforms (e.g., Windows, Red Hat, CentOS, Ubuntu), etc.

You might find that public AMIs suit your needs. However, you might want to customize a public AMI and then save that customized AMI for your own use (i.e., create a new AMI). For more information about creating your own AMIs, see Creating Your Own AMIs (p. 17).

After you create a new AMI, you can keep it private so that only you can use it, or you can share it with only specific AWS accounts that you specify. Another option is to make your customized AMI public so that the EC2 community can use it. Building safe, secure, usable AMIs for public consumption is a fairly straightforward process, if you follow a few simple guidelines. For information on how to use shared AMIs and how to share AMIs, see Using Shared AMIs (p. 102) and Sharing AMIs Safely (p. 47).

Paid AMIs are AMIs that you purchase from developers or AMIs that come with service contracts from organizations such as Red Hat. If you're interested in selling an AMI to other developers, go to http://aws.amazon.com/devpay to learn about Amazon DevPay.

To help categorize and manage your AMIs, you can assign *tags* of your choice to them. For more information, see Using Tags (p. 275).

Storage for the Root Device

All AMIs are categorized as either *backed by Amazon EBS* or *backed by instance store*. The former means that the root device is an Amazon EBS snapshot and appears as an Amazon EBS volume when an instance is launched from the AMI. The latter means that the root device is stored in Amazon S3 and appears as instance store when an instance is launched from the AMI.

There are other important differences between the two types of AMIs. For an in-depth discussion of these differences, see Using Amazon EBS-Backed AMIs and Instances (p. 177). The following table gives a quick summary of the differences.

Characteristic	Amazon EBS-Backed	Amazon instance store-backed
Boot Time	Usually Less than 1 minute	Usually Less than 5 minutes
Size Limit	1 TiB	10 GiB
Root Device Location	Amazon EBS volume	Instance storage
Data Persistence	Data persists on instance failure and can persist on instance termination	Data persists for the life of the instance; nonroot devices can use Amazon EBS
Upgrading	The instance type, kernel, RAM disk, and user data can be changed while the instance is stopped.	Instance attributes are fixed for the life of an instance
Charges	Instance usage, Amazon EBS volume usage, and Amazon EBS snapshot charges for AMI storage	Instance usage and Amazon S3 charges for AMI storage
AMI Creation/Bundling	Uses a single command/call	Requires installation and use of AMI tools
Stopped State	Can be placed in stopped state where instance is not running, but the instance is persisted in Amazon EBS	Cannot be in stopped state; instances are running or not

Related Topics

- Launching and Using Instances (p. 75)
- Connecting to Instances (p. 111)
- Creating Your Own AMIs (p. 17)
- Sharing AMIs Safely (p. 47)
- Using Tags (p. 275)

Creating Your Own AMIs

Topics

When the available public AMIs do not address what you're looking for, you have a number of options for obtaining just the right AMI for your needs. You can create your own AMIs from scratch or from customized instances.

Note

Although some customers and third-party documentation refer to creating AMIs as *bundling*, we use the term in this documentation only to refer to the process of creating Amazon EC2 instance store-backed AMIs.

Creating your own AMIs helps you make the most of Amazon EC2. Your AMIs become the basic unit of deployment; they enable you to rapidly boot new custom instances as you need them. This section gives an overview of your options, maps the process flow, identifies the tools you need, and walks you through the steps.

We assume that you have considered the *public AMIs* and decided that they don't meet your requirements. We also assume that you're familiar with AMI and instance concepts.

For background information, see the following sections:

- Using AMIs (p. 15)
- Using Instances (p. 69)
- Root Device Storage (p. 8)
- Amazon Elastic Block Store (p. 126)

For information about available AMIs , see Amazon Machine Images (AMIs) on the web.

Overview of the AMI Creation Process

The root storage device you select for the AMI determines the process you must follow to create the AMI. It's either an Amazon EBS-backed AMI or an Amazon EC2 instance store-backed AMI. There are significant differences between Amazon EBS-backed and Amazon EC2 instance store-backed AMIs, including AMI size limits and storage and persistence of data. For information on the differences between these choices, see Summary of Differences (p. 177).

First, select the root storage device, then you'll have an idea of the process for creating the AMI:

- **Amazon EBS-Backed AMIs—**General process applies to Windows and Linux/UNIX systems.
  For information, see Creating Amazon EBS-Backed AMIs (p. 18).
- **Amazon EC2 instance store-backed AMIs—**Separate processes apply to Windows and Linux/UNIX systems.
  The process depends on the operating system you want.
  - **Amazon EC2 instance store-backed Windows AMIs—**Can start only with a Windows instance.

For information, see Bundling Amazon EC2 instance store-backed Windows AMIs (p. 21).

- **Amazon EC2 instance store-backed Linux/UNIX AMIs—**Start with an existing AMI or a fresh installation.
  For information about bundling Amazon EC2 instance store-backed Linux/UNIX AMIs, see Bundling Amazon EC2 instance store-backed Linux/UNIX AMIs (p. 25).

  To start with an existing Amazon EC2 instance store-backed AMI, see From an Existing AMI (p. 27).

  To start with a fresh installation, see From a Loopback (p. 31).

Interfaces for AMI Creation and Management

You can use different interfaces to create AMIs depending on the type of AMI you want. If you are creating Amazon EBS-backed AMIs for Linux/UNIX or Windows systems, or if you are bundling Amazon EC2 instance store-backed Windows AMIs, you can use the command line, the API, or the AWS management console. If you are creating Amazon EC2 instance store-backed Linux/UNIX systems, you will need to install AMI tools on the instance you are preparing to bundle. For information about AMI tools, see Tools You Need (p. 26).

Each discussion of a specific AMI creation process identifies the tools that can be used as well as how to use them.

For background information about tools, see the following sections in the Amazon Elastic Compute Cloud (EC2) documentation:

- Available Interfaces (p. 10)
- Getting Started with the Command Line Tools (p. 384)

Creating Amazon EBS-Backed AMIs

You can create an AMI that uses an Amazon EBS volume as its root device with Windows or Linux/UNIX operating systems.

The process is simple. You start with an Amazon EBS-backed AMI (for example, one of the public AMIs that Amazon provides), and modify it to suit your particular needs. You can't create an Amazon EBS-backed AMI starting with an Amazon EC2 instance store-backed instance.

Important

You cannot delete a snapshot of the root device of an EBS volume used by a registered AMI. You must first de-register the AMI before you can delete the snapshot.

If the AMI you start with doesn't already have the storage devices you want attached, you can add them by creating EBS volumes or using block device mapping. To create EBS volumes, use `ec2-create-volume` and `ec2-attach-volume` or `CreateVolume` and `AttachVolume`. You also can call `ec2-run-instances` or `RunInstances` with block device mapping information for the devices you want to add. For more information about block device mapping, see Block Device Mapping (p. 167).

Important

If you customize your instance with ephemeral storage devices or additional EBS volumes besides the root device, the new AMI contains block device mapping information for those storage devices and volumes. When you then launch an instance from your new AMI, the instance automatically launches with the additional devices and volumes.

When you have a running instance in the state you want, use `ec2-create-image` or `CreateImage` and specify the instance ID. Amazon EC2 powers down the instance, takes images of any volumes that were attached, creates and registers the AMI, and then reboots the instance. It takes several minutes for the entire process to complete. If you customized the local instance with storage devices or additional EBS volumes besides the root device, the new AMI contains block device mapping information for those storage devices. When you launch an instance from your new AMI, the instance automatically launches with the additional devices. The instance stores, also called ephemeral stores, are new and don't contain any data from the instance store of the original instance used to create the AMI.

The reason Amazon EC2 powers down the instance before creating the AMI is to ensure that everything on the instance is stopped and in a consistent state during the creation process. If you're confident that your instance is in a consistent state appropriate for AMI creation, you can add the --no-reboot flag to `ec2-create-image` or `CreateImage` that tells Amazon EC2 not to power down and reboot the instance. You can't specify this flag when creating the AMI in the AWS console. With this flag, the instance remains running throughout the AMI creation process. Some file systems, such as xfs, can freeze and unfreeze activity, making it safe to create the image without rebooting the instance.

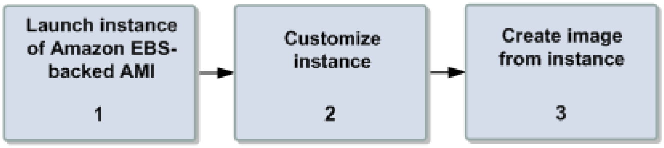
Note

Amazon EBS-backed instances are stored as Amazon EBS data. Standard storage rates apply.

Sysprep does not automatically run for Amazon EBS-backed Windows instances.

Process for Creating Amazon EBS-Backed AMIs

The following diagram and table describe the process for creating an Amazon EBS-backed AMI. The information is applicable to both Linux/UNIX and Windows AMIs.


Launch instance of Amazon EBS-backed AMI
1
Customize instance
2
Create image from instance
3

General Tasks in Creating Amazon EBS-Backed AMIs

1	Start by launching an instance of an Amazon EBS-backed AMI that is similar to the AMI you want to create. For example, you might take a public AMI that uses the operating system you want to use for your AMI. The instance must be from an Amazon EBS-backed AMI; you can't start with an instance of an Amazon EC2 instance store-backed AMI.
2	When the instance is running, customize it as you want. For example, you could attach additional Amazon EBS volumes, or load applications or data on to the instance. **Important** If you customize your instance with ephemeral storage devices or additional EBS volumes besides the root device, the new AMI contains block device mapping information for those storage devices and volumes. When you then launch an instance from your new AMI, the instance automatically launches with the additional devices and volumes.
3	When the instance is set up the way you want it, create an image from that instance.

Special Cases

In some cases, the general tasks in creating Amazon EBS-backed AMIs don't apply:

- You don't have the original AMI to launch instances from.
  In this case, you can create an Amazon EBS-backed AMI by registering a snapshot of a root device. You must own the snapshot and it must be a Linux/UNIX system (this process is not available for Windows instances). For more information about creating an AMI this way, see Launching an Instance from a Snapshot (p. 189).
- You have an Amazon EC2 instance store-backed Linux/UNIX AMI.
  In this case, you can convert the AMI to be backed by Amazon EBS. You cannot convert a Windows AMI backed by instance store. For more information about converting a Linux/UNIX AMI, see Converting Amazon EC2 instance store-backed AMIs to EBS-Backed AMIs (p. 21).

How to Create Amazon EBS-Backed AMIs

You perform the tasks of creating an Amazon EBS-backed AMI by using the AWS Management Console, the command line tools, or the API. The following section describes the steps using these tools and interface.

AWS Management Console

For instructions that use the AWS Management Console, see Creating an Image from a Running Instance (p. 185).

Command Line Tools

To create an Amazon EBS-backed AMI

1. Enter the following command to create an image:

   ```
   PROMPT>  ec2-create-image -n your_image_name  instance_id
   ```

   For example:

   ```
   PROMPT>  ec2-create-image -n "My AMI" i-eb977f82
   ```

   Amazon EC2 creates an image and returns an AMI ID.

   ```
   IMAGE ami-8675309
   ```

2. If you want to check whether the AMI is ready, enter the following command:

   ```
   $  ec2-describe-images -o self
   ```

   Amazon EC2 returns information about the AMI.

API

To create an Amazon EBS-backed AMI

- Construct the following Query request to create an image:

```
https://ec2.amazonaws.com/
?Action=CreateImage
&InstanceId=instance_id
&Name=My_Ami
&...auth parameters...
```

In the following example response, Amazon EC2 creates the image and returns its AMI ID.

```
<CreateImageResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <imageId>ami-8675309</imageId>
</CreateImageResponse>
```

AMI creation can take time. You can check whether the AMI is ready by using .

Converting Amazon EC2 instance store-backed AMIs to EBS-Backed AMIs

There's no simple API or button in the AWS Management Console that converts an existing Amazon EC2 instance store-backed AMI to an Amazon EBS-backed AMI.

Important

It isn't possible to convert an Amazon EC2 instance store-backed Windows AMI. You need to take a public Amazon EBS-backed Windows AMI, modify it to meet your specifications, and then create an image from it. For information, see Creating an Image from a Running Instance (p. 185).

You can convert Amazon EC2 instance store-backed Linux/UNIX AMIs to EBS-backed systems. The following table describes the process.

Process for Converting a Linux/UNIX Amazon EC2 instance store-backed AMI to an EBS-Backed AMI

1	Copy the AMI's root device information to an Amazon EBS volume. For more information, see the related task list in Amazon EC2 Root Device Storage Usage Scenarios (p. 163)
2	Create a snapshot of that volume. For more information, see Creating an Amazon EBS Snapshot (p. 140).
3	Register the image with a block device mapping that maps the root device name of your choice to the snapshot you just created. For an example of how to register an image, see Automatically Attaching Volumes (p. 187).

You might find it useful to refer to available blog posts that discuss conversion. The following are two example blogs; AWS, however, takes no responsibility for the completeness or accuracy of the content:

- http://www.elastician.com/2009/12/creating-ebs-backed-ami-from-s3-backed.html
- http://coderslike.us/2009/12/07/amazon-ec2-boot-from-ebs-and-ami-conversion/

Bundling Amazon EC2 instance store-backed Windows AMIs

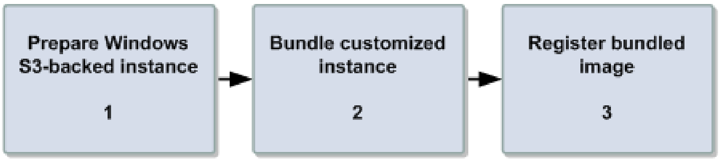
You can create Amazon EC2 instance store-backed AMIs with Windows systems. Because you can't create Amazon EC2 instance store-backed Windows AMIs from scratch, you must start with a Windows instance, customize it to meet your requirements, and then use the command line tools, the API, or console to bundle the instance and register the image.

Note

Before bundling an instance, you can configure the instance using the EC2Config service. For more information, see Appendix C: Windows Configuration Service (p. 424).

For information about Windows instance types, see Windows Instance Types (p. 72). For information about connecting to Windows instances, see Connecting to Windows Instances (p. 122)

The following diagram shows the general tasks in bundling Amazon EC2 instance store-backed Windows AMIs.


Prepare Windows S3-backed instance
1
Bundle customized instance
2
Register bundled image
3

After you've prepared your instance, the bundling process performs the following tasks, which are listed in the order that they usually take place:

- Excludes any ephemeral drives (i.e., the D: drive on your instance is not included in the bundled AMI)
- Compresses the image to minimize bandwidth usage and storage requirements
- Encrypts and signs the compressed image to ensure confidentiality and authenticates the image against its creator
- Splits the encrypted image into manageable parts for upload
- Runs `sysprep` to strip out computer-specific information (e.g., the MAC address and computer name) to prepare the Windows image for virtualization
- Creates a manifest file that contains a list of the image parts with their checksums
- Puts all the parts of the AMI into an Amazon S3 bucket you specify

Note

All Amazon EC2 instance store-backed AMIs are loaded from Amazon S3 storage. When you create the AMI, you must upload it to an existing account on Amazon S3. Amazon S3 stores data objects in buckets, which are similar in concept to directories. Buckets have globally unique names and are owned by unique AWS accounts.

Caution

Instance store drives (e.g., the D: drive) are not included in the bundled AMI. Instance store drives and their data are deleted when the instance is terminated. You must store any data that you want to use with the new AMI on the root drive or an Amazon EBS volume. For more information about Amazon EBS volumes, see Amazon Elastic Block Store (p. 126).

You cannot launch the new AMI until the bundling is complete and you have registered it. The bundling process can take time and you can monitor the task by using the `ec2-describe-bundle-tasks` command. While bundling is in progress, the task moves through a succession of states, including "waiting-for-shutdown," "storing," and "complete" states. The output during the process looks like this:

```
BUNDLE bun-1509ed7c i-cb2a81a0 mybucket myimage 2010-03-19T08:22:48+0000 2010-
03-19T08:23:50+0000 bundling 12%
```

When bundling is complete, the status changes to "complete."

You must register your bundled image with Amazon EC2, so Amazon EC2 can locate it and run instances based on it. You don't have to register the bundled image immediately after the bundle task completes. You can still register the bundled image even if the bundle task no longer appears in your list of completed bundle tasks (each task remains on the list for only a limited time).

Note

If you make any changes to the source image stored in Amazon S3, you must reregister the image.

How to Bundle Amazon EC2 instance store-backed Windows AMIs

You can use the AWS Management Console, command line, or API to bundle Amazon EC2 instance store-backed Windows AMIs.

AWS Management Console

Bundling and registering Amazon EC2 instance store-backed Windows images using the console is easy. You can click buttons for each task.

To bundle Amazon EC2 instance store-backed AMIs

1. Log in to the AWS Management Console. Decide on the Windows instance you want to use, and prepare the instance to meet your specifications.

   For information about running an instance, see Running an Instance (p. 85).
2. Right-click the instance you customized and select **Bundle Instance (S3 AMI)**.

   The **Bundle Instance** dialog box opens. It shows the ID of the instance you want to bundle.
3. Provide your **Amazon S3 Key Name** and the **Amazon S3 Bucket Name** where you want the new AMI to be stored, and then click **Bundle**.

   You should see the **Bundle Instance** message box informing you that you successfully made the bundling request. The message box also provides the **Bundle Task ID**.

   Click **View Bundling Tasks** to see the status of the task. Click **Close** to close the message box.

   Note

   The **Bundle Tasks** status can show **waiting-for-shutdown** when Amazon EC2 is bundling an Amazon EC2 instance store-backed instance. Amazon EC2 shuts down the instance, bundles it, and puts the new bundle into Amazon S3.

4. Navigate to the list of your AMIs when the bundling task is complete, right-click the newly-bundled AMI, and then select **Register New AMI**.

   The **Register Image** dialog box opens. Provide the **AMI Manifest Path** and click **Register**.

Command Line Tools

To bundle Amazon EC2 instance store-backed AMIs

1. Log in to the Windows instance and modify it to meet your requirements.

   Note

   We highly recommend that you change the password of the AMI. If you use the Amazon EC2-provided password, write it down so you can access instances launched from this AMI. You cannot get the password of new instances using the `ec2-get-password` command.

2. If you want to reduce your startup time, delete any temporary files on your instance using the Disk Cleanup tool, defragment your system using Disk Defragmenter, and zero out free space using `sdelete -c C:\`.
   You can download the `sdelete` utility from the *sdelete Download Page* or the *Microsoft Web Site*
   .

3. Enter the following command to bundle the instance into Amazon S3 on your local system where you have installed the API tools (not on the instance you are bundling):

```
PROMPT> ec2-bundle-instance <instance_id> -b <bucket_name> -p <bundle_name>
 -o <access_key_id> -w <secret_access_key>
```

   The <*instance_id*> is the name of the instance, <*bucket_name*> is the name of the bucket in which to store the AMI, and <*bundle_name*> is the common name for the files to store in Amazon S3.

   Note

   To perform this task, you need your AWS Access Key ID (<aws-access-key-id>) and AWS Secret Access Key (<aws-secret-access-key>). For more information, see How to Get Your Access Key ID and Secret Access Key (p. 267).

   The **ec2-bundle-instance** utility uploads the bundled AMI to a specified bucket. If you have used Amazon S3 before, you can use any of your existing buckets or just give **ec2-bundle-instance** any name that makes sense to you. If the specified bucket does not exist, the command creates it. If the specified bucket belongs to another AWS account, **ec2-bundle-instance** fails, and you have to specify a different name.

   The following is an example of a fully specified `ec2-bundle-instance` command.

```
PROMPT> ec2-bundle-instance -31c2425a -b mybucket -p myimage -o AKI
ADQKE4SARGYLE -w eW91dHViZS5jb20vd2F0Y2g/dj1SU3NKMTlzeTNKSQ==
BUNDLE  bun-e3a4418a   i-31c2425a      mybucket myimage 2010-03-
19T08:22:48+0000 2010-03-19T08:22:48+0000        pending
```

   Amazon EC2 shuts down the instance, saves it as an AMI, and restarts it.

4. Enter the following command to register the image:

```
PROMPT> ec2-register <your-s3-bucket>/image.manifest.xml -n image_name
IMAGE ami-2bb65342
```

Amazon EC2 returns an AMI identifier, the value next to the `IMAGE` tag (`ami-2bb65342` in the example) that you can use to run instances.

API

The following procedure mirrors the steps used with the command line tools.

To bundle Amazon EC2 instance store-backed AMIs

1. Log in to the Windows instance and modify it to meet your requirements.
2. Construct the following request to bundle the instance into Amazon S3 on your local system where you have installed the API tools (not on the instance you are bundling):

   ```
   https://ec2.amazonaws.com/
   ?Action=BundleInstance
   &InstanceId=-i-e468cd8d
   &Storage.S3.AWSAccessKeyId=10QMXFEV71ZS32XQFTR2
   &Storage.S3.Bucket=my-bucket
   &Storage.S3.Prefix=winami
   &Storage.S3.UploadPolicy=eyJleHBpcmF0aW9uIjogIjIwMDgtMDgtMzBUMDg6NDk6MDlaIi
   wiY29uZGl0aW9ucyI6IFt7ImJ1Y2tldCI6ICJteS1idWNrZXQifSxbInN0YXJ0cy13aXRoIiw
   gIiRrZXkiLCAibXktbmV3LWltYWdlIl0seyJhY2wiOiAiZWMyLWJ1bmRsZS1yZWFkIn1dfQ%3D%3D
   &Storage.S3.UploadPolicySignature=fh5tyyyQD8W4COEthj3nlGNtJMU%3D
   &AuthParams
   ```

   Note

   To perform this task, you need your AWS Access Key ID (<aws-access-key-id>) and AWS Secret Access Key (<aws-secret-access-key>). For more information, see How to Get Your Access Key ID and Secret Access Key (p. 267).

   For information about the `BundleInstance` command, see BundleInstance in the *Amazon Elastic Compute Cloud API Reference*.
3. Construct the following command to register the image:

   ```
   https://ec2.amazonaws.com/
   ?Action=RegisterImage
   &ImageLocation=full-path-to-amazon-manifest
   &AuthParams
   ```

   Amazon EC2 returns an AMI identifier that you can use to run instances.

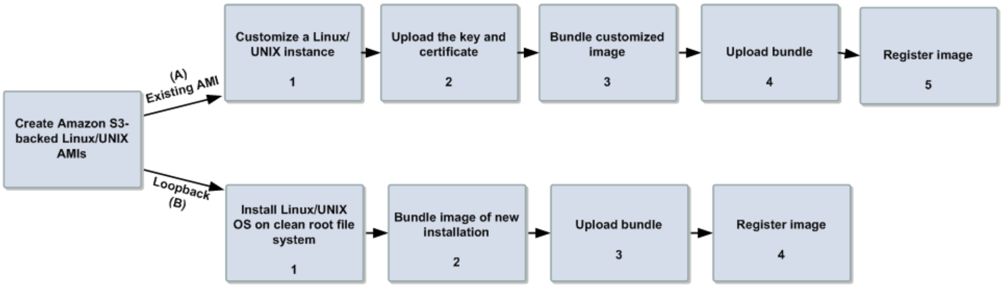
Bundling Amazon EC2 instance store-backed Linux/UNIX AMIs

Topics

For Linux/UNIX systems, you have two common ways to prepare Amazon EC2 instance store-backed AMIs. The easiest method (A) involves launching an existing public AMI and modifying it according to your requirements. For more information, see From an Existing AMI (p. 27).

Another approach (B) is to build a fresh installation either on a stand-alone machine or on an empty file system mounted by loopback. The process entails building an operating system installation from scratch. After you've built the installation package to your satisfaction, you must bundle it using the AMI tool for bundling volumes and register it using the command line tool for registering images. For information, see From a Loopback (p. 31).

The following diagram shows the general tasks in creating Amazon EC2 instance store-backed Linux/UNIX AMIs.


Create Amazon S3-backed Linux/UNIX AMIs
(A) Existing AMI
Customize a Linux/ UNIX instance 1
Upload the key and certificate 2
Bundle customized image 3
Upload bundle 4
Register image 5
Loopback (B)
Install Linux/UNIX OS on clean root file system 1
Bundle image of new installation 2
Upload bundle 3
Register image 4

This section discusses the steps for creating AMIs from an existing file and from a loopback, and some basics about the AMI tools.

Tools You Need

Amazon created the Amazon EC2 AMI tools to help you perform specific tasks for Amazon EC2 instance store-backed Linux/UNIX AMIs. You use these AMI tools, which are a set of command line utilities, for bundling and uploading Amazon EC2 instance store-backed Linux/UNIX AMIs. You also use these AMI tools for managing these bundled images. For information about the specific AMI tools, see AMI Tools Reference in the *Amazon Elastic Compute Cloud Command Line Reference*.

When you bundle an Amazon EC2 instance store-backed Linux/UNIX AMI and you start with an instance, you use the AMI tools for bundling and uploading the bundle, and then you use the API tools to register the image. If you are creating an Amazon EC2 instance store-backed AMI via loopback, you first prepare the instance, then use the AMI tools to bundle before you use the API tools to register the image you created.

If you are starting with an instance of an Amazon public AMI, it might already have the AMI tools installed. Try running the command `ec2-bundle-vol` to check if the instance already has the AMI tools.

If the tools are already installed, you can jump to the section that discusses the bundling process you want to complete:

- From an Existing AMI (p. 27)
- From a Loopback (p. 31)

If the tools aren't installed, read on. This section describes installation and usage information when using AMI tools.

Install the AMI Tools

The AMI tools are available in both a Zip file and as an RPM suitable for running on Fedora Core with Ruby 1.8.2 (or greater) installed. You need root privileges to install the software.

For information about installing the AMI tools, go to Amazon EC2 AMI Tools.

To install the AMI tools

1. Install Ruby using the YUM package manager.

```
# yum install ruby
```

2. Install the AMI tools RPM.

```
# rpm -i ec2-ami-tools-x.x-xxxx.i386.rpm
```

View the AMI Tools Documentation

This section describes how to view Linux/UNIX documentation.

To view the manual for each tool

- Append `--manual` to the command that invokes the tool.

```
# ec2-bundle-image --manual
```

To view help for each tool

- Append `--help` to the command that invokes the tool.

```
# ec2-bundle-image --help
```

From an Existing AMI

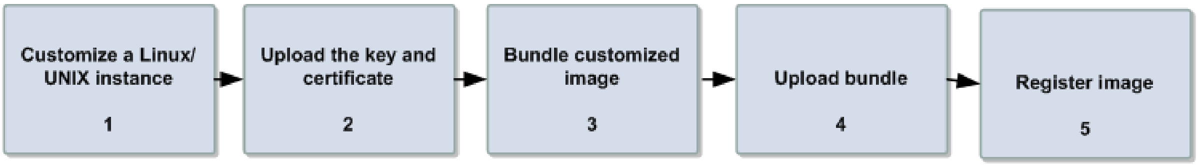
To quickly and easily get a new working AMI, start with an existing public AMI or one of your own. You can then modify it and create a new AMI.

Note

Before you select an AMI, determine whether the instance types you plan to launch are 32-bit or 64-bit. For more information, see Instance Families and Types (p. 69).

Make sure you are using GNU Tar 1.15 or later.

The following diagram shows the general tasks in creating Amazon EC2 instance store-backed Linux/UNIX AMIs from an existing AMI.


Customize a Linux/ UNIX instance
1
Upload the key and certificate
2
Bundle customized image
3
Upload bundle
4
Register image
5

Tasks to Use an Existing AMI to Create a New AMI

1	Customize an Instance (p. 28)
2	Upload the Key and Certificate (p. 28)
3	Bundle a Customized Image (Requires Root Privileges) (p. 29)
4	Upload a Bundled AMI (p. 30)
5	Register the AMI (p. 31)

Customize an Instance

Customizing an instance involves the following series of steps:

1. Selecting an AMI from available AMIs.
2. Launching an instance from the AMI you selected.
3. Making changes to (thus, *customizing*) the instance, such as altering the Linux configuration, adding software, and configuring web applications.

For more information, see Launching and Using Instances (p. 75).

After you've launched an instance according to your specifications, proceed to the next steps to create a new AMI using the customized instance.

Upload the Key and Certificate

Your new AMI should be encrypted and signed to ensure that only you and Amazon EC2 can access it. To make this happen, you must upload your Amazon EC2 private key and X.509 certificate to an instance store directory on your running instance. The private key and the certificate will be used in the AMI bundling process. For information on obtaining your Amazon EC2 private key and X.509 certificate, see How to Create an X.509 Certificate and Private Key (p. 267).

An Amazon Linux AMI mounts the instance store on */media/ephemeral0*. If you are using an Amazon Linux AMI, you must first login to your running instance as *ec2-user* and use the following code to grant write permissions to the instance store before you can upload the private key and the certificate.

```
$  sudo chmod 777/media/ephemeral0
```

Note

Non Amazon Linux AMIs use different login names (for example, root, ubuntu, etc.,) and a different location to mount the instance store (for example, */mnt*).

To upload your Amazon EC2 private key and X.509 certificate

- Copy your Amazon EC2 private key and X.509 certificate to the directory where an instance store is mounted using a secure copy function such as SCP.

  The following code shows the syntax to use with the `scp` command.

```
$ scp -i <keypair_name> <private_keyfile> <certificate_file> <user
name>@<dns_location>
       <instance store directory>
```

Where,

Parameter	Description
keypair_name	Name of your key pair.
private_keyfile	File that contains the private key.
certificate_file	File that contains the certificate.
username	Login name you use to log in to your instance.
dns_location	DNS location of the instance within Amazon EC2.
instance store directory	Directory where your instance store is mounted.

Note

It is important to upload the key and certificate files into the instance store directory of your instance to prevent them from being bundled with the new AMI.

Amazon EC2 returns the name of the files and some performance statistics.

Example

The following is an example of a fully specified `scp` command using the Amazon Linux AMI.

```
$ scp  -i id_rsa-gsg-keypair pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
ec2-user@ec2-67-202-51-223.compute-1.amazonaws.com:/media/ephemeral0
pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem  100%  717     0.7KB/s   00:00
cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
100%  685     0.7KB/s   00:00
```

Bundle a Customized Image (Requires Root Privileges)

When you have the image that meets your specifications, you need to bundle it.

To bundle a customized image

- Enter the following command:

```
# ec2-bundle-vol -k <private_keyfile> -c <certificate_file> -u <user_id>
```

The <*private_keyfile*> is the file that contains the private key, <*certificate_file*> is the file that contains the certificate, and <*user_id*> is the ID associated with your AWS account.

Note

Make sure to disable SELinux when running `ec2-bundle-vol`.

Note

The user ID is your AWS account ID without dashes. It consists of 12 to 15 characters, and it's *not* the same as your Access Key ID. For information about viewing your account ID, see Viewing Your Account ID (p. 268).

Example

This command bundles the local machine root file system.

```
# ec2-bundle-vol -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem -c cert-HKZYK
TAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem -u 999988887777

Please specify a value for arch [i386]:
Copying / into the image file /tmp/image...
Excluding:
/sys
...
...
1+0 records in
1+0 records out
1048576 bytes (1.0 MB) copied, 0.00172 s, 610 MB/s
mke2fs 1.40.4 (31-Dec-2007)
...
Bundling image file...
Splitting /tmp/image.tar.gz.enc...
Created image.part.00
Created image.part.01
...
Created image.part.NN
Generating digests for each part...
Digests generated.
Creating bundle manifest...
ec2-bundle-vol complete.
```

Upload a Bundled AMI

You must upload the bundled AMI to Amazon S3 before Amazon EC2 can access it. This task is necessary when you create Amazon EC2 instance store-backed AMIs from an existing file or from a loopback. Use `ec2-upload-bundle` to upload the bundled AMI that you created earlier. Amazon S3 stores data objects in buckets, which are similar to directories.

Buckets must have globally unique names. The `ec2-upload-bundle` utility uploads the bundled AMI to a specified bucket. If the specified bucket does not exist, it will be created. If the specified bucket exists and belongs to another AWS account, the `ec2-upload-bundle` command will fail.

To upload the bundled AMI

- Enter the following command:

```
# ec2-upload-bundle -b <bucket> -m <manifest_path> -a <access_key> -s
<secret_key>
```

The <*bucket*> is the target bucket (you can bundle to a "subfolder"; e.g., my-bucket/ami-image-folder), <*access_key*> is your AWS Access Key, and <*secret_key*> is your AWS Secret Key. The *-m* <*manifest_path*> is the full path to the manifest file (e.g., /tmp/image.manifest.xml).

The AMI manifest file and all image parts are uploaded to Amazon S3. The manifest file is encrypted with the Amazon EC2 public key before being uploaded.

Register the AMI

Your must register your image with Amazon EC2, so Amazon EC2 can locate it and run instances based on it. This task is necessary when you create Amazon EC2 instance store-backed AMIs from an existing file or from a loopback.

Note

If you make any changes to the source image stored in Amazon S3, you must reregister the image.

To register the AMI that you created and uploaded to Amazon S3

- Enter the following command (note that it is part of the Amazon EC2 API tools, and not the AMI tools):

```
PROMPT> ec2-register <your-s3-bucket>/image.manifest.xml -n image_name
IMAGE ami-2bb65342
```

Amazon EC2 returns an AMI identifier, the value next to the `IMAGE` tag (`ami-2bb65342` in the example), that you can use to run instances.

From a Loopback

Creating AMIs through a loopback involves doing a full operating system installation on a clean root file system, but avoids having to create a new root disk partition and file system on a physical disk. After you have installed your operating system, you can bundle the resulting image as an AMI with the `ec2-bundle-image` command, which is part of the AMI tools (and not an API action). For more information about the `ec2-bundle-image` command and the AMI tools, go to the Amazon Elastic Compute Cloud Command Line Reference.

Note

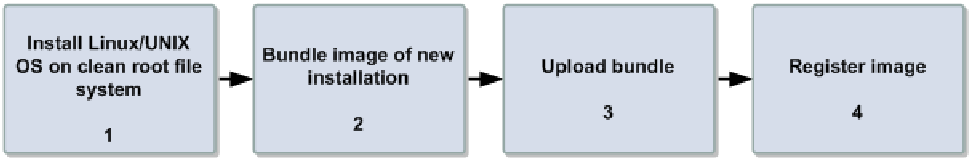
This method works only with AMIs that use instance stores for their root devices. This method is not applicable for AMIs backed by Amazon EBS.

Before you select an AMI, determine whether the instance types you plan to launch are 32-bit or 64-bit. For more information, see Instance Families and Types (p. 69).

Make sure you are using GNU Tar 1.15 or later.

These examples use Fedora Core 4. Please make any adjustments for your distribution.

The following diagram shows the general tasks in creating Amazon EC2 instance store-backed Linux/UNIX AMIs from a loopback.


Install Linux/UNIX OS on clean root file system
1
Bundle image of new installation
2
Upload bundle
3
Register image
4

Tasks to Create a New AMI Through a Loopback

1	Install Linux/UNIX and Prepare the System a. Create a File to Host the AMI (p. 32) b. Create a Root File System Inside the File (p. 32) c. Mount the File through Loopback (p. 33) d. Prepare for the Installation (p. 34) e. Install the Operating System (p. 35) f. Configure the Operating System (p. 36)
2	Bundle the Loopback File Image (p. 37)
3	Upload a Bundled AMI (p. 30)
4	Register the AMI (p. 31)

Create a File to Host the AMI

The dd utility can create files of arbitrary sizes. Make sure to create a file large enough to host the operating system, tools, and applications that you will install. For example, a baseline Linux/UNIX installation requires about 700 MB, so your file should be at least 1 GB.

To create a file to host the AMI

- Enter the following command:

```
# dd if=/dev/zero of=image_name bs=1M count=size
```

The <*image_name*> is the name of the image file you are creating and <*size*> is the size of the file in megabytes.

Example

The following example creates a 1 GB file (1024*1 MB).

```
# dd if=/dev/zero of=my-image.fs bs=1M count=1024
1024+0 records in
1024+0 records out
```

Create a Root File System Inside the File

The `mkfs` utility has several variations that can create a file system inside the image file you are creating. Typical Linux/UNIX installations default to `ext2` or `ext3` file systems.

To create an `ext3` file system

- Enter the following command:

```
# mke2fs -F -j <image_name>
```

  The <*image_name*> is the name of the image file.

Example

The following example creates an `ext3` file system.

```
# mke2fs -F -j my-image.fs
mke2fs 1.38 (30-Jun-2005)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
131072 inodes, 262144 blocks
13107 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=268435456
8 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 24 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
```

Mount the File through Loopback

The loopback module enables you to use a normal file as if it were a raw device, which gives you a file system within a file. Mounting a file system image file through loopback presents it as part of the normal file system. You can then modify it using your favorite file management tools and utilities.

To mount the file through loopback

1. Enter the following command to create a mount point in the file system where the image will be attached:

   ```
   # mkdir <image_mountpoint>
   ```

   The <*image_mountpoint*> is the location where the image will be mounted.
2. Mount the file system image:

   ```
   # mount -o loop  <image_name> <image_mountpoint>
   ```

   The <*image_name*> is the name of the image file and <*image_mountpoint*> is the mount location.

Example

The following commands create and mount the my-image.fs image file.

```
# mkdir /mnt/ec2-fs
# mount -o loop my-image.fs /mnt/ec2-fs
```

Prepare for the Installation

Before the operating system installation can proceed, you must create and prepare the newly created root file system.

To prepare for the installation

1. Create a /dev directory and populate it with a minimal set of devices. You can ignore the errors in the output.

   ```
   # mkdir /mnt/ec2-fs/dev
   # /sbin/MAKEDEV -d <image_mountpoint>/dev -x console
   # /sbin/MAKEDEV -d <image_mountpoint>/dev -x null
   # /sbin/MAKEDEV -d <image_mountpoint>/dev -x zero
   ```

   The <*image_mountpoint*> is the mount location.
2. Create the fstab file within the /etc directory and add the following:

   ```
   /dev/sda1  /         ext3    defaults        1 1
   none       /dev/pts  devpts  gid=5,mode=620  0 0
   none       /dev/shm  tmpfs   defaults        0 0
   none       /proc     proc    defaults        0 0
   none       /sys      sysfs   defaults        0 0
   ```

3. Create a temporary YUM configuration file (e.g., yum-xen.conf) and add the following content.

   ```
   [main]
   cachedir=/var/cache/yum
   debuglevel=2
   logfile=/var/log/yum.log
   exclude=*-debuginfo
   gpgcheck=0
   obsoletes=1
   reposdir=/dev/null

   [base]
   name=Fedora Core 4 - $basearch - Base
   mirrorlist=http://fedora.redhat.com/download/mirrors/fedora-core-$releasever
   enabled=1

   [updates-released]
   name=Fedora Core 4 - $basearch - Released Updates
   mirrorlist=http://fedora.redhat.com/download/mirrors/updates-released-
   fc$releasever
   enabled=1
   ```

This step ensures that all the required basic packages and utilities are installed. You can locate this file anywhere on your main file system (not on your loopback file system) and is used only during installation.

4. Enter the following:

```
# mkdir <image_mountpoint>/proc
# mount -t proc none <image_mountpoint>/proc
```

The <*image_mountpoint*> is the mount location. A groupadd utility bug in the shadow-utils package (versions prior to 4.0.7-7) requires you to mount the new proc file system manually with the preceding command.

Example

These commands create the /dev directory and populate it with a minimal set of devices:

```
# mkdir /mnt/ec2-fs/dev
# /sbin/MAKEDEV -d /mnt/ec2-fs/dev -x console
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
# /sbin/MAKEDEV -d /mnt/ec2-fs/dev -x null
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
# /sbin/MAKEDEV -d /mnt/ec2-fs/dev -x zero
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
MAKEDEV: mkdir: File exists
```

This example creates and mounts the /mnt/ec2-fs/proc directory.

```
# mkdir /mnt/ec2-fs/proc
# mount -t proc none /mnt/ec2-fs/proc
```

Install the Operating System

At this stage, the basic directories and files are created and you are ready to install the operating system. Depending on the speed of the host and network link to the repository, this process might take a while.

To install the operating system

- Enter the following command:

```
# yum -c <yum_configuration_file> --installroot=<image_mountpoint> -y groupin
stall Base
```

The <*yum_configuration_file*> is the name of the YUM configuration file and <*image_mountpoint*> is the mount location.

You now have a base installation, which you can configure for operation inside Amazon EC2 and customize for your use.

Example

This example installs the operating system at the /mnt/ec2-fs mount point using the yum-xen.conf YUM configuration file.

```
# yum -c yum-xen.conf --installroot=/mnt/ec2-fs -y groupinstall Base
Setting up Group Process
Setting up repositories
base                      100% |=========================| 1.1 kB    00:00
updates-released          100% |=========================| 1.1 kB    00:00
comps.xml                 100% |=========================| 693 kB    00:00
comps.xml                 100% |=========================| 693 kB    00:00
Setting up repositories
Reading repository metadata in from local files
primary.xml.gz            100% |=========================| 824 kB    00:00
base      : ################################################## 2772/2772
Added 2772 new packages, deleted 0 old in 15.32 seconds
primary.xml.gz            100% |=========================| 824 kB    00:00
updates-re: ################################################## 2772/2772
Added 2772 new packages, deleted 0 old in 10.74 seconds
...
Complete!
```

Configure the Operating System

After successfully installing the base operating system, you must configure your networking and hard drives to work in the Amazon EC2 environment.

To configure the operating system

1. Edit (or create) /mnt/ec2-fs/etc/sysconfig/network-scripts/ifcfg-eth0 and make sure it contains at least the following information:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
TYPE=Ethernet
USERCTL=yes
PEERDNS=yes
IPV6INIT=no
```

Note

The Amazon EC2 DHCP server ignores hostname requests. If you set DHCP_HOSTNAME, the local hostname will be set on the instance but not externally. Additionally, the local hostname will be the same for all instances of the AMI, which might be confusing.

2. Verify that the following line appears in the /mnt/ec2-fs/etc/sysconfig/network file so that networking starts:

```
NETWORKING=yes
```

3. Add the following lines to /mnt/ec2-fs/etc/fstab so that local disk storage on /dev/sda2 and swap space on /dev/sda3 are mounted at system startup:

```
/dev/sda2  /mnt      ext3    defaults        0 0
/dev/sda3  swap      swap    defaults        0 0
```

Note

The /dev/sda2 and /dev/sda3 storage locations only apply to small instances. For more information on instance storage, see Amazon EC2 Instance Storage (p. 151).

4. Allocate appropriate system run levels so that all your required services start at system startup. For example, to enable the service my-service on multiuser and networked run levels, enter the following commands:

   ```
   # chroot /mnt/ec2-fs /bin/sh
   # chkconfig --level 345 my-service on
   # exit
   ```

   Your new installation is successfully installed and configured to operate in the Amazon EC2 environment.
5. Enter the following commands to umount the image:

   ```
   # umount <image_mountpoint>/proc
   # umount -d <image_mountpoint>
   ```

   The <*image_mountpoint*> is the mount location.

Example

The following example unmounts the installation from the /mnt/ec2-fs mount point.

```
# umount /mnt/ec2-fs/proc
# umount -d /mnt/ec2-fs
```

Bundle the Loopback File Image

To bundle the loopback file image

- Enter the following command:

  ```
  # ec2-bundle-image -i <image_name>.img -k <private_keyfile> -c <certific
  ate_file> -u <user_id>
  ```

  The <*image_name*> is the name of the image file, <*private_keyfile*> is the file that contains the private key, <*certificate_file*> is the file that contains the certificate, and <*user_id*> is the ID associated with your AWS account.

  Note

  The user ID is your AWS account ID without dashes. It consists of 12 to 15 characters, and it's *not* the same as your Access Key ID. For information about viewing your account ID, see Viewing Your Account ID (p. 268).

Example

This command bundles an image created in a loopback file.

```
# ec2-bundle-image -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem -c cert-HKZYK
TAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem -u 999988887777 -i image.img -d bundled/ -p
fred -r x86_64
 Please specify a value for arch [i386]:
 Bundling image file...
 Splitting bundled/fred.gz.crypt...
 Created fred.part.00
 Created fred.part.01
 Created fred.part.02
 Created fred.part.03
 Created fred.part.04
 Created fred.part.05
 Created fred.part.06
 Created fred.part.07
 Created fred.part.08
 Created fred.part.09
 Created fred.part.10
 Created fred.part.11
 Created fred.part.12
 Created fred.part.13
 Created fred.part.14
 Generating digests for each part...
 Digests generated.
 Creating bundle manifest...
 ec2-bundle-image complete.
```

Creating Paid AMIs

Topics

This section gives an introduction to Amazon DevPay and paid AMIs, which are AMIs you sell to other Amazon EC2 users.

Important

Amazon DevPay does not support Amazon EBS-backed AMIs. All paid AMIs must be backed by Amazon instance store.

Amazon DevPay and Paid AMIs

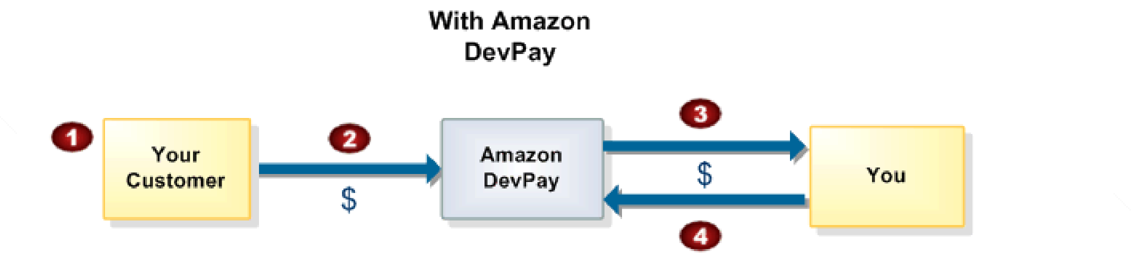
A paid AMI is an AMI that you sell to other Amazon EC2 users. They pay you according to the price you set. To be able to create a paid AMI, you use Amazon DevPay. What is Amazon DevPay?

Important

Amazon DevPay does not support Amazon EBS-backed AMIs. All paid AMIs must be backed by Amazon instance store.

Amazon DevPay is a billing and account management service that enables you to get paid for an AMI you create and that other Amazon EC2 users use. Amazon DevPay creates and manages the order pipeline and billing system for you. Your customers sign up for your AMI, and Amazon DevPay automatically meters their usage of Amazon EC2, bills them based on the pricing you set, and collects their payments. DevPay offers the following:

- You can charge customers for your Amazon EC2 instance store-backed AMI; the charges can include recurring charges based on the customer's usage Amazon EC2, a fixed one-time charge, and a recurring monthly charge.
- Your customers can easily sign up and pay for your Amazon EC2 instance store-backed AMI with their trusted Amazon.com accounts.
- Your customers are authenticated, thus ensuring they have access only to what they should.
- If your customers don't pay their bills, DevPay turns off their access to your AMI for you.
- Amazon Payments handles payment processing.



Basic DevPay Flow

1	Your customer uses an Amazon.com account to sign up and pay for your Amazon EC2 instance store-backed AMI. The sign-up page indicates that you have teamed up with Amazon Payments to make billing easy and secure.
2	Your customer pays the price you've defined to use your product.
3	DevPay subtracts a fixed transaction fee and pays you the difference.
4	You pay the costs of Amazon EC2 that your Amazon EC2 instance store-backed AMI used, and a percentage-based DevPay fee.

For more information about Amazon DevPay, refer to the *Amazon DevPay Developer Guide*.

Summary of How Paid AMIs Work

With a paid AMI, your customers:

- Must be signed up to use Amazon EC2 themselves
- Buy your paid Amazon EC2 instance store-backed AMI and then launch instances of it
- Always use *their own* AWS credentials when launching instances; you don't launch instances of your paid AMI for them with your credentials
- Pay the price you set for the paid AMI, and not the normal Amazon EC2 rates

Important

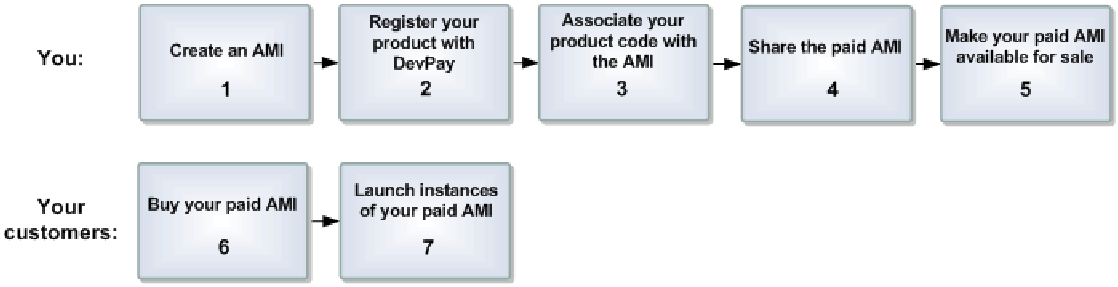
The discounts you get with Amazon EC2 Reserved Instances don't apply to Amazon DevPay products. That is, if you purchase Reserved Instances, you don't get the lower price associated with them when your customers launch your paid or supported AMIs. Also, if your customers purchase Reserved Instances, when they use your paid or supported AMIs, they continue to pay the price you specified for the use of your paid or supported AMIs. For more information about Reserved Instances, see Reserved Instances (p. 224).

You can also use Amazon EC2 and Amazon DevPay together with a *supported AMI*. For more information about supported AMIs, see Supported AMIs (p. 45).

The following figure and table summarize the basic flow for creating and using paid AMIs.

Note

Detailed information about most of the following steps is provided in the *Amazon DevPay Developer Guide*.


You:
Create an AMI
1
Register your product with DevPay
2
Associate your product code with the AMI
3
Share the paid AMI
4
Make your paid AMI available for sale
5
Your customers:
Buy your paid AMI
6
Launch instances of your paid AMI
7

Paid AMI Process

1	You create an Amazon EC2 instance store-backed AMI as described elsewhere in this guide.
2	You register a product with Amazon DevPay. For more information, see Product Registration (p. 42). As part of this process, you provide a product description, product pricing, etc. This registration process creates a product code for the product and a URL where customers can sign up to use the product (called the *purchase URL*).
3	You use an Amazon EC2 command or API call to associate the product code with your Amazon EC2 instance store-backed AMI. For more information, see Associating a Product Code with an AMI (p. 43). This makes the AMI a paid AMI.
4	You use an Amazon EC2 command or API call to share the Amazon EC2 instance store-backed AMI with select customers or the public. For more information, see Sharing Your Paid AMI (p. 44). **Note** Even if you share a paid AMI and it has a product code, no one can use the AMI until they sign up for it (see the following steps).

5	You make your paid AMI available for sale. To do this, you make the aforementioned purchase URL available. You can advertise your paid AMI in the Solutions Catalog on the AWS Developer Connection site and on the Amazon Machine Images (AMIs) page on the AWS Resource Center.
6	Customers use the purchase URL you provide to sign up for and purchase your product. If they're not already signed up for Amazon EC2, they'll be prompted to sign up. They purchase your product with their Amazon.com accounts. They must have the credentials needed to launch Amazon EC2 instances. At this point, they have the AMI ID (from step 5).
7	Customers then launch an Amazon EC2 instance specifying the AMI ID. Because you associated the shared AMI with the product code, the customers are charged at the rate you set. For more information, see Paying for AMIs (p. 104).

Note

You can associate your DevPay product code with more than one Amazon EC2 instance store-backed AMI. However, a single AMI can be associated with only one product code. If you plan to sell multiple AMIs, you could sell them all under a single product code, or different product codes (by registering multiple DevPay products). For information about why you might choose a single product code or multiple product codes, go to If You Have Multiple AMIs to Sell in the *Amazon DevPay Developer Guide*.

Each customer's bill for the AMI is displayed on their Application Billing page, which shows the activity for DevPay products. Also, at any time, you can confirm the customer is still currently subscribed to your product. For more information, refer to the *Amazon DevPay Developer Guide*.

Note

In the preceding process, you associate your product code with your own Amazon EC2 instance store-backed AMI and sell the AMI as a DevPay product. There's another scenario for using DevPay with Amazon EC2 in which you sell software or a service to EC2 users and let them associate your product code with their own Amazon EC2 instance store-backed AMIs. For more information, see Supported AMIs (p. 45).

The Product Code and AMI Rebundling

Associating a product code with an AMI turns it into a paid AMI that EC2 users must sign up for to use. Can you ensure that the product code stays with the AMI if someone rebundles the AMI? The answer varies for Linux/UNIX AMIs and Windows AMIs. These are described in the following sections.

Linux/UNIX AMIs

If you give the customer root access to your paid Linux/UNIX AMI, the customer can rebundle it (for more information, see Creating Your Own AMIs (p. 17)). If your customer uses AWS tools to rebundle the AMI, the rebundled AMI inherits the product code. When launching instances of the rebundled AMI, the customer is still billed for usage based on your price. However, if the customer doesn't use the AWS tools when rebundling, the rebundled AMI won't inherit the product code, and the customer will pay normal Amazon EC2 rates and not your price. Also, a customer with root access could find some other way to remove the product code from the AMI.

When a customer contacts you for support for a paid AMI, you can confirm your product code is associated with the AMI and the customer's instance is currently running the AMI. For more information, go to Confirming an Instance Is Running with a Product Code (p. 44).

If you have software installed on the AMI, the software can retrieve the instance metadata to determine if the product code is associated with the instance. For more information, see Getting the Product Code from Within an Instance (p. 45).

Keep in mind that the preceding methods for confirming the association of the product code with the instance are not foolproof because a customer with root access to the instance could return false information indicating the product code is associated with the instance.

Windows AMIs

When you associate a product code with a Windows AMI, the association is permanent. Therefore, we recommend you keep a separate, base copy of the AMI that has no product code associated with it.

Anyone who purchases a Windows AMI can rebundle it (for more information, see Creating Your Own AMIs (p. 17)). The product code is automatically transferred to the rebundled AMI. When EC2 users launch instances of the rebundled AMI, they pay the rates you set when you registered your DevPay product. In turn, you're charged for the EC2 costs they incur.

Product Registration

You must register a product with Amazon DevPay. The product can cover a single Amazon EC2 instance store-backed AMI that you want to sell or multiple Amazon EC2 instance store-backed AMIs. During registration, you provide product information such as pricing, and you receive information you need to sell your product.

Important

The *Amazon DevPay Developer Guide* covers the procedure for registering your product with Amazon DevPay. Before you register your product, we recommend you read the information in that guide about how to set your AMI's price and how billing for Amazon DevPay products works.

You provide the following information during registration:

- Company name
- Product name
- Product description (as you want your customers to see it)
- Redirect URL (the page you want customers to see after they have purchased the product)
- Any terms and conditions you want displayed (optional)
- Contact e-mail address and telephone number (to be used by AWS and not displayed to customers)
- Contact e-mail or URL (to be displayed to customers)
- The specific Regions, environments, and instance types the product covers
- Pricing for use of the product (you can set different prices based on Region, environment, and instance type)

The information you display at the redirect URL should give information about the AMI.

Registration provides you with the following information:

- Product code
- Product token
- Purchase URL

You need the product code and purchase URL to integrate your product with DevPay as described in Summary of How Paid AMIs Work (p. 39) and Supported AMIs (p. 45). You need the product token if you're going to set up your system to later verify whether a customer is still subscribed to your product. For more information, refer to the *Amazon DevPay Developer Guide*.

Note

AWS must approve your product after you register it. The approval process typically takes one business day.

Associating a Product Code with an AMI

You must be the owner of an Amazon EC2 instance store-backed AMI to associate a product code with it. Each AMI can have only a single product code associated with it, but you can associate a single product code with more than one AMI. You might do this if you have similar versions of an AMI (for example, a 32-bit version and a 64-bit version), you've assigned them all the same price, and you'd like to minimize the number of Amazon DevPay product codes you have (to make your bookkeeping easier).

To associate a product code with an AMI

- Enter the following command:

```
PROMPT>   ec2-modify-image-attribute <ami_id> --product-code <product_code>
```

The <`ami_id`> is the AMI ID and <`product_code`> is the product code.

To verify the product code is associated with the AMI

- Enter the following command:

```
PROMPT>  ec2-describe-image-attribute  <ami_id> --product-code
```

You can't change or remove the `productCodes` attribute after you've set it. If you want to use the same image without the product code or associate a different product code with the image, you must reregister the image to obtain a new AMI ID. You can then use that AMI without a product code or associate the new product code with the AMI ID.

Example

The following example associates the ami-2bb65342 AMI with the 774F4FF8 product code.

```
PROMPT>   ec2-modify-image-attribute ami-2bb65342 --product-code 774F4FF8

productCodes        ami-2bb65342            productCode   774F4FF8
```

This example verifies that the product code is associated with the AMI.

```
PROMPT>  ec2-describe-image-attribute ami-2bb65342 --product-code

productCodes        ami-2bb65342            productCode   774F4FF8
```

Sharing Your Paid AMI

After you associate the product code with the Amazon EC2 instance store-backed AMI, you need to share the AMI with select customers or the public by using the **ec2-modify-image-attribute** command.

To share the AMI

- Enter the following command:

```
PROMPT>   ec2-modify-image-attribute <ami_id> --launch-permission -a all
```

  The *<ami_id>* is the AMI ID.

Even though you've shared the AMI, no one can use it until they sign up for your product by going to the purchase URL. Once customers sign up, any instances of the paid AMI they launch will be billed at the rate you specified during product registration.

Example

The following example shares the ami-2bb65342 AMI with the public.

```
PROMPT>   ec2-modify-image-attribute ami-2bb65342 --launch-permission -a all

launchPermission        ami-2bb65342    ADD     group   all
```

Confirming an Instance Is Running with a Product Code

If you have created a product for others to use with their AMIs (the supported AMI scenario), you might want to confirm that a particular AMI is associated with your product code and a particular instance is currently running that AMI.

Note

You must be the owner of the product code to successfully call **ec2-confirm-product-instance** with that product code.

Because your customers don't own the product code, they should describe their instances to confirm their instances are running with your product code.

To confirm an instance is running an AMI associated with your product code

- Enter the following command:

```
PROMPT>   ec2-confirm-product-instance <product_code> -i <instance>
```

  The <*product_code*> is the product code and <*instance*> is the instance.

If the AMI is associated with the product code, `true` is returned with the AMI owner's account ID. Otherwise, `false` is returned.

Example

The following example confirms whether the i-10a64379 instance is running the 6883959E product code.

```
PROMPT>   ec2-confirm-product-instance 6883959E -i i-10a64379

6883959E i-10a64379 true 495219933132
```

Getting the Product Code from Within an Instance

A running Amazon EC2 instance can determine if has an Amazon DevPay product code. The instance retrieves the product code similarly to how retrieves other metadata. For more information about retrieving metadata, see Using Instance Metadata (p. 96).

To retrieve a product code, query a web server with this REST-like API call:

```
GET http://169.254.169.254/2007-03-01/meta-data/product-codes
```

Amazon EC2 returns a response similar to the following:

```
774F4FF8
```

Supported AMIs

Supported AMIs are different from paid AMIs. With a supported AMI, you charge for software or a service you provide that customers use with their own AMIs.

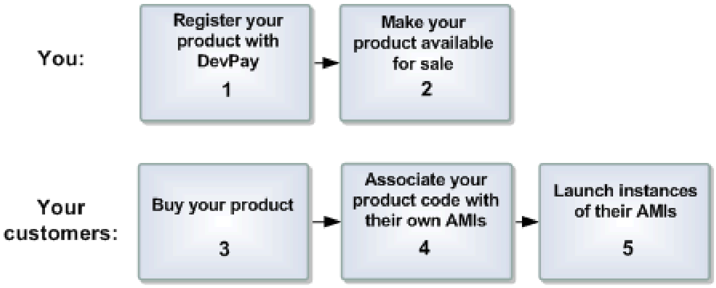
The main difference between a paid AMI and a *supported AMI* is how the AMI is associated with a product code:

- **Paid AMI—**You associate your own product code with your own AMI
- **Supported AMI—**Other EC2 users associate your product code with their own AMIs

Important

If your customers purchase Reserved Instances, they don't get the Reserved Instance price discount with supported AMIs. That is, if they associate your product code with their AMIs, they don't get the lower price associated with their Reserved Instances when they launch those AMIs. They always pay the price that you specified for your DevPay product. For more information, see Reserved Instances (p. 224).

The following figure and table summarizes the flow for creating and using supported AMIs.


You:
Register your product with DevPay
1
Make your product available for sale
2
Your customers:
Buy your product
3
Associate your product code with their own AMIs
4
Launch instances of their AMIs
5

Supported AMI Process

1	You register a product with Amazon DevPay. For more information, see Product Registration (p. 42). As part of this process, you provide a product description, product pricing, etc. This registration process creates a product code for the product and a URL where customers can sign up to use the product (called the *purchase URL*).
2	You make your product available for sale.
3	Customers use the purchase URL to sign up for and purchase your product. If they're not already signed up for Amazon EC2, they'll be prompted to sign up. They purchase your product with their Amazon.com accounts. They must have the credentials needed to launch Amazon EC2 instances. At this point, they have the product code (from step 2).
4	Customers then use an Amazon EC2 command or API call to associate the product code with their Amazon EC2 instance store-backed AMIs. For more information, see Associating a Product Code with an AMI (p. 43).
5	Customers then launch one or more instances of the AMIs. Because the customers associated their AMIs with the product code, they are charged at the rate you set.

Note

Amazon EC2 prevents your customers (but not you as the product code owner) from associating your product code with AMI types the product isn't configured for. For example, if the product is configured only for Linux/UNIX AMIs, your customers can't associate the product code with Windows AMIs. Also, Amazon EC2 prevents your customers from launching specific instance types your product isn't configured for. For more information about product configuration, go to Your Product's Configuration and Price in the *Amazon DevPay Developer Guide*.

Each customer's bill for the AMI is displayed on their Application Billing page, which shows the activity for DevPay products. For more information, refer to the *Amazon DevPay Developer Guide*.

When a customer contacts you for support for an AMI, you can confirm your product code is associated with the AMI and the customer's instance is currently running the AMI. For more information, see Confirming an Instance Is Running with a Product Code (p. 44).

Sharing AMIs Safely

Topics

Shared AMIs are AMIs that developers build and make available for other AWS developers to use. Building safe, secure, usable AMIs for public consumption is a fairly straightforward process if you follow a few simple guidelines.

For additional information about sharing AMIs safely, go to the following articles on the AWS Developer Resources website.

- How To Share and Use Public AMIs in A Secure Manner
- Public AMI Publishing: Hardening and Clean-up Requirements

For information on building shared AMIs, see Protecting a Shared AMI (Linux/UNIX) (p. 47). For information on sharing AMIs, see Sharing AMIs (p. 51).

Protecting a Shared AMI (Linux/UNIX)

Following these guidelines produces a better user experience, makes your users' *instances* less vulnerable to security issues, and helps protect you.

To build a shared AMI, follow these guidelines:

Shared AMI Guidelines

1	Update the AMI Tools at Boot Time (p. 47)
2	Disable Password-Based Logins for Root (p. 48)
3	Install Public Key Credentials (p. 49)
4	Disabling sshd DNS Checks (optional) (p. 50)
5	Identify Yourself (p. 50)
6	Protect Yourself (p. 51)
7	Protect Paid AMIs (p. 51)

Note

These guidelines are written for Fedora distributions, but the principles apply to any AMI. You might need to modify the provided examples for other distributions. For other distributions, review their documentation or search the AWS forums in case someone else has done it already.

Update the AMI Tools at Boot Time

For AMIs backed by instance store, we recommend that your AMIs download and upgrade the Amazon EC2 AMI creation tools during startup. This ensures that new AMIs based on your shared AMIs will have the latest AMI tools.

To update the AMI tools at startup on Fedora

- Add the following to `rc.local`:

```
# Update the Amazon EC2 AMI creation tools
echo " + Updating ec2-ami-tools"
wget http://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.noarch.rpm && \
rpm -Uvh ec2-ami-tools.noarch.rpm && \
echo " + Updated ec2-ami-tools"
```

Use this method to automatically update other software on your image.

Note

When deciding which software to automatically update, consider the amount of WAN traffic that the update will generate (your users will be charged for it) and the risk of the update breaking other software on the AMI.

Note

The preceding procedure applies to Fedora distributions. For other distributions:

- On most Red Hat systems, add these steps to your `/etc/rc.d/rc.local` script.
- On Gentoo systems, add them to `/etc/conf.d/local.local`.
- On Ubuntu systems, add them to `/etc/rc.local`.
- On Debian, you might need to create a start up script in `/etc/init.d` and use `update-rc.d <scriptname> defaults 99` (where `<scriptname>` is the name of the script you created) and add the steps to this script.

Disable Password-Based Logins for Root

Using a fixed root password for a public AMI is a security risk that can quickly become known. Even relying on users to change the password after the first login opens a small window of opportunity for potential abuse.

To solve this problem, disable password-based logins for the root user. Additionally, we recommend you randomize the root password at boot.

To disable password-based logins for root

1. Open the `/etc/ssh/sshd_config` file with a text editor and locate the following line:

```
#PermitRootLogin yes
```

2. Change the line to:

```
PermitRootLogin without-password
```

The location of this configuration file might differ for your distribution, or if you are not running OpenSSH. If this is the case, consult the relevant documentation.

3. To randomize the root password, add the following to your boot process:

```
if [ -f "/root/firstrun" ] ; then
  dd if=/dev/urandom count=50|md5sum|passwd --stdin root
  rm -f /root/firstrun
else
  echo "* Firstrun *" && touch /root/firstrun
```

   Note

   This step assumes that a /root/firstboot file is bundled with the image. If the file was not created, the root password will never be randomized and will be set to the default.

Note

If you are using a distribution other than Fedora, you might need to consult the documentation that accompanied the distribution.

Remove SSH Host Key Pairs

If you plan to share an AMI derived from a public AMI, remove the existing SSH host key pairs located in `/etc/ssh`. This forces SSH to generate new unique SSH key pairs when someone launches an instance using your AMI, improving security and reducing the likelihood of "man-in-the-middle" attacks.

The following list shows the SSH files to remove.

- ssh_host_dsa_key
- ssh_host_dsa_key.pub
- ssh_host_key
- ssh_host_key.pub
- ssh_host_rsa_key
- ssh_host_rsa_key.pub

Important

If you forget to remove the existing SSH host key pairs from your public AMI, our routine auditing process will notify you and all customers running instances of your AMI of the potential security risk. After a short grace period, we will mark the AMI private.

Install Public Key Credentials

After configuring the AMI to prevent logging in using a password, you must make sure users can log in using another mechanism.

Amazon EC2 allows users to specify a public-private key pair name when launching an instance. When a valid key pair name is provided to the `RunInstances` API call (or through the command line API tools), the public key (the portion of the key pair that Amazon EC2 retains on the server after a call to `CreateKeyPair` or `ImportKeyPair`) is made available to the instance through an HTTP query against the instance metadata.

To login through SSH, your AMI must retrieve the key value at boot and append it to `/root/.ssh/authorized_keys` (or the equivalent for any other user account on the AMI). Users will be able to launch instances of your AMI with a key pair and log in without requiring a root password.

```
if [ ! -d /root/.ssh ] ; then
        mkdir -p /root/.ssh
        chmod 700 /root/.ssh
fi
# Fetch public key using HTTP
curl http://169.254.169.254/latest//meta-data/public-keys/0/openssh-key >
/tmp/my-key
if [ $? -eq 0 ] ; then
        cat /tmp/my-key >> /root/.ssh/authorized_keys
        chmod 700 /root/.ssh/authorized_keys
        rm /tmp/my-key
fi
```

This can be applied to any user account; you do not need to restrict it to root.

Note

Rebundling an instance based on this image includes the key with which it was launched. To prevent the key's inclusion, you must clear out (or delete) the `authorized_keys` file or exclude this file from rebundling.

Disabling sshd DNS Checks (optional)

Disabling sshd DNS checks slightly weakens your sshd security. However, if DNS resolution fails, SSH logins will still work. If you do not disable sshd checks, DNS resolution failures prevent all logins.

To disable sshd DNS checks

1. Open the `/etc/ssh/sshd_config` file with a text editor and locate the following line:

```
#UseDNS yes
```

2. Change the line to:

```
UseDNS no
```

Note

The location of this configuration file can differ for your distribution or if you are not running OpenSSH. If this is the case, consult the relevant documentation.

Identify Yourself

Currently, there is no easy way to know who provided a shared AMI because each AMI is represented by an account ID.

We recommend that you post a description of your AMI, and the AMI ID, in the Amazon EC2 developer forum. This provides a convenient central location for users who are interested in trying new shared AMIs. You can also post the AMI to the Amazon Machine Images (AMIs) page.

Protect Yourself

The previous sections described how to make your shared AMIs safe, secure, and usable for the users who launch them. This section describes guidelines to protect yourself from the users of your AMI.

We recommend against storing sensitive data or software on any AMI that you share. Users who launch a shared AMI might be able to rebundle it and register it as their own. Follow these guidelines to help you to avoid some easily overlooked security risks:

- Always delete the shell history before bundling. If you attempt more than one bundle upload in the same image, the shell history contains your secret access key.
- Bundling a running instance requires your private key and X.509 certificate. Put these and other credentials in a location that is not bundled (such as the instance store).
- Exclude the ssh authorized keys when bundling the image. The Amazon public images store the public key used to launch an instance with its ssh authorized keys file.

Note

Unfortunately, it is not possible for this list of guidelines to be exhaustive. Build your shared AMIs carefully and take time to consider where you might expose sensitive data.

Protect Paid AMIs

The simplest way to prevent users from rebundling paid AMIs that you create is to not provide root access to the AMI and to pay attention to security announcements that involve privilege escalations. Amazon EC2 requires you to have root access on any AMI that you rebundle.

If you must provide root access to an AMI, Amazon EC2 tools are designed to protect the product code. Although this is effective, it is not guaranteed and users might create AMIs using other tools.

To ensure users cannot rebundle your paid AMIs, we recommend that you configure your application to check the instance metadata to verify that the product code is intact.

Sharing AMIs

Amazon EC2 enables you to share your AMIs with other AWS accounts. This section describes how to share AMIs using the Amazon EC2 command line tools.

Note

Before proceeding, make sure to read the security considerations of sharing AMIs in the Protecting a Shared AMI (Linux/UNIX) (p. 47) section.

AMIs have a `launchPermission` property that controls which AWS accounts, besides the owner's, are allowed to launch instances of that AMI. By modifying an AMI's `launchPermission` property, you can allow all AWS accounts to launch the AMI (make the AMI public) or only allow a few specific accounts to launch the AMI.

The `launchPermission` attribute is a list of accounts and *launch groups*. *Launch permissions* can be granted by adding or removing items from the list. Explicit launch permissions for accounts are granted or revoked by adding or removing their AWS account IDs. The only launch group currently supported is the `all` group, which makes the AMI public. The rest of this section refers to launch groups simply as groups. Launch groups are not the same as security groups and the two should not be confused. An AMI can have both public and explicit launch permissions.

Note

You are not billed when your AMI is launched by other AWS accounts. The accounts launching the AMI are billed.

Making an AMI Public

To make an AMI public

- Add the `all` group to the AMI's `launchPermission`.

```
PROMPT>  ec2-modify-image-attribute <ami_id> --launch-permission -a all
```

The <*ami_id*> parameter is the ID of the AMI.

To check the launch permissions of an AMI

- Enter the following command, where <*ami_id*> is the ID of the AMI.

```
PROMPT> ec2-describe-image-attribute <ami_id> -l
```

To make an AMI private again

- Remove the `all` group from its launch permissions, where <*ami_id*> is the ID of the AMI.

```
PROMPT> ec2-modify-image-attribute <ami_id> -l -r all
```

This will not affect any explicit launch permissions for the AMI or any running instances of the AMI.

Example

This example makes the ami-2bb65342 AMI public.

```
PROMPT>  ec2-modify-image-attribute ami-2bb65342  --launch-permission -a all
launchPermission        ami-2bb65342    ADD     group   all
```

This examples displays the launch permissions of the ami-2bb65342 AMI.

```
PROMPT> ec2-describe-image-attribute ami-2bb65342 -l
launchPermission        ami-2bb65342    group   all
```

This example removes the `all` group from the permissions of the ami-2bb65342 AMI, making it private.

```
PROMPT> ec2-modify-image-attribute ami-2bb65342 -l -r all
launchPermission        ami-2bb65342    REMOVE  group   all
```

Sharing an AMI with Specific AWS Accounts

You can share an AMI with specific AWS accounts without making the AMI public. All you need is the account ID.

To grant explicit launch permissions

- Enter the following command:

```
PROMPT> ec2-modify-image-attribute <ami_id> -l -a <user_id>
```

The <*ami_id*> is the ID of the AMI and <*user_id*> is the account ID, without hyphens.

To remove launch permissions for an account

- Enter the following command:

```
PROMPT> ec2-modify-image-attribute <ami_id> -l -r <user_id>
```

The <*ami_id*> is the ID of the AMI and <*user_id*> is the account ID, without hyphens.

To remove all launch permissions

- Enter the following command to remove all public and explicit launch permissions:

```
PROMPT> ec2-reset-image-attribute <ami_id> -l
```

The <*ami_id*> is the ID of the AMI.

Note

The AMI owner always has rights to the AMI and is unaffected by this command.

Example

The following example grants launch permissions to the AWS account with ID 999988887777 for the ami-2bb65342 AMI:

```
PROMPT> ec2-modify-image-attribute ami-2bb65342 -l -a 999988887777
launchPermission        ami-2bb65342    ADD     userId  999988887777
```

The following example removes launch permissions from the AWS account with ID 999988887777 for the ami-2bb65342 AMI:

```
PROMPT> ec2-modify-image-attribute ami-2bb65342 -l -r 999988887777
launchPermission        ami-2bb65342    REMOVE  userId  999988887777
```

The following example removes all public and explicit launch permissions from the ami-2bb65342 AMI:

```
PROMPT> ec2-reset-image-attribute ami-2bb65342 -l
launchPermission        ami-2bb65342    RESET
```

Publishing Shared AMIs

After you create a shared AMI, you can publish information about it in the Amazon EC2 Resource Center.

To publish your AMI

1. Post it in the Public AMIs Folder of the Amazon Web Services Resource Center, and include the following information:

   - AMI ID
   - AMI name (for Amazon EBS-backed AMIs) or AMI manifest (for Amazon EC2 instance store-backed AMIs)
   - Publisher
   - Publisher URL
   - OS / Distribution
   - Key Features
   - Description
   - Daemons / Services
   - Release Notes

2. If you want to, you can paste the following information into the document. You must be in HTML edit mode.

```
<strong>AMI&nbsp;ID: </strong>[ami-id]<br />
<strong>AMI&nbsp;Manifest: </strong>[bucket/image.manifest.xml]<br />
<h2>About this &AMI;</h2>
<ul>

    <li>Published by [Publisher] (<a href="http://www.mysite.com">[ht
tp://www.mysite.com]</a>).<br />
            </li>
    <li>[Key Features] <br />
            </li>
    <li>[Description]</li>
    <li>This image contains the following daemons / services:
```

```
    <ul>

        <li>[Daemon 1]</li>
        <li>[Daemon 2]</li>
    </ul>
            </li>
</ul>
<h2><strong>What&#39;s New?</strong></h2>The following changes were made on
 [Date].<br />
<ul>

    <li>[Release Notes 1]</li>
</ul>
<span style="font-size: x-small; font-family: courier new,couri
er">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;-&nbsp;[Note 1]</span><br />
<span style="font-size: x-small; font-family: courier new,couri
er">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;-&nbsp;[Note 2]</span><br />
<span style="font-size: x-small; font-family: courier new,couri
er">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;-&nbsp;[Note 3]</span><br />
<ul>
```

Amazon Linux AMI Basics

The Amazon Linux AMI is a supported and maintained Linux image provided by Amazon Web Services (AWS) for use on Amazon Elastic Compute Cloud (Amazon EC2). It is designed to provide a stable, secure, and high-performance execution environment for applications running on Amazon EC2. It also includes several packages that enable easy integration with AWS, including launch configuration tools and many popular AWS libraries and tools.

Amazon Linux AMI is similar to CentOS in file system layout and package management. Applications built for CentOS should run without recompilation on the Amazon Linux AMI. The AMIs come pre-installed with the Extra Packages for Enterprise Linux (EPEL) repositories. However, these repositories are disabled by default. AWS provides ongoing security and maintenance updates to all instances running the Amazon Linux AMI. The Amazon Linux AMI is configured for use only in Amazon EC2 and is provided at no additional charge to Amazon EC2 users.

Finding the Amazon Linux AMI

For a list of the latest Amazon Linux AMIs, go to Amazon Linux AMI.

Launching and Connecting to an Amazon Linux Instance

After locating your desired AMI, note its AMI ID. You can use the AMI ID to launch and then connect to your instance.

The Amazon Linux AMI does not allow remote root SSH by default. Also, password authentication is disabled to prevent brute-force password attacks. To enable SSH logins to a running Amazon Linux AMI, you must provide your key pair to the instance at launch. You must also set the security group used to launch your instance to allow SSH access. By default, the only account that can log in remotely via SSH is ec2-user. The ec2-user has sudo privileges. If you want to enable remote root log in, please be aware that it is less secure than relying on key pairs and a secondary user.

For information on launching and using your Amazon Linux instance, go to Launching and Using Instances. For information on connecting to your Amazon Linux instance, go to Connecting to Linux/UNIX Instances from Linux/UNIX.

Identifying Amazon Linux AMI Images

Each image contains a unique `/etc/image-id` that identifies the AMI. This file contains information about the image.

Following is an example of the `/etc/image-id` file:

```
# cat /etc/image-id
image_name="amzn-ami"
image_version="2011.09"
image_arch="x86_64"
image_file="amzn-ami-2011.09.1.x86_64.ext4"
image_stamp="610f-3a9e"
image_date="20110921183121"
recipe_name="amzn ami"
recipe_id="2aaca5dd-b34f-4a46-8281-d2471ce38631"
```

The `image_name`, `image_version`, and `image_arch` items come from the build recipe that Amazon used to construct the image. The `image_stamp` is simply a unique random hex value generated during image creation. The `image_date` item is in YYYYMMDDhhmmss format, and is the UTC time of image creation. The `recipe_name` and `recipe_id` refer to the name and ID of the build recipe Amazon used to construct the image, which identifies the current running version of the Amazon Linux AMI. This file will not change as you install updates from the yum repository.

Amazon linux AMIs contain a `/etc/system-release` file that specifies the current release that is installed. This file is updated through yum and is part of the system-release `rpm`.

Following is an example of `/etc/system-release` file:

```
# cat /etc/system-release
Amazon Linux AMI release 2011.09
```

An Amazon Linux AMI also contains a machine readable version of the `/etc/system-release` file found in `/etc/system-release-cpe` and follows the CPE specification from MITRE (CPE).

Included AWS Command Line Tools

The following popular command line tools for AWS integration and usage have been included in the Amazon Linux AMI:

- aws-apitools-common
- aws-apitools-mon
- aws-apitools-ec2
- aws-apitools-elb
- aws-apitools-rds
- aws-apitools-as
- aws-apitools-iam
- aws-apitools-ses

- aws-apitools-cfn

To simplify the configuration of these tools, a simple script has been included to prepare AWS_CREDENTIAL_FILE, JAVA_HOME, AWS_PATH, PATH, and product-specific environment variables after a credential file has been installed.

Also, to allow the installation of multiple versions of the API and AMI tools, we have placed symlinks to the desired versions of these tools in `/opt/aws`, as described here:

- /opt/aws/bin—Symlink farm to /bin directories in each of the installed tools directories.
- /opt/aws/{apitools|amitools}—Products are installed in directories of the form [name]-version and symlink [name] attached to the most recently installed version.
- /opt/aws/{apitools|amitools}/[name]/environment.sh—Used by /etc/profile.d/aws-apitools-common.sh to set product-specific environment variables (EC2_HOME, etc.).

Cloud-init

Cloud-init is an open source application built by Canonical that is used to bootstrap Linux images in a cloud computing environment such as Amazon EC2. The Amazon Linux AMI contains a customized version of Cloud-init. It enables you to specify actions that should happen to your instance at boot time. You can pass desired actions to Cloud-init through the user data fields when launching an instance. This means you can use common AMIs for many use cases and configure them dynamically at startup. The Amazon Linux AMI also uses Cloud-init to perform initial configuration of the ec2-user account.

For more information about Cloud-init, go to https://-help.ubuntu.com/-community/-CloudInit.

The Amazon Linux AMIs use the following Cloud-init actions (configurable in /etc/sysconfig/cloudinit):

- action: INIT (always runs)
  - Setting a default locale.
  - Setting the hostname.
  - Parsing and handling user data.
- action: CONFIG_SSH
  - Generating host private SSHkeys.
  - Adding user's public SSHkeys to .ssh/authorized_keys for easy login and administration.
- action: PACKAGE_SETUP
  - Preparing yum repo.
  - Handles package actions defined in user data.
- action: RUNCMD
  - Runs a shell command.
- action: RUN_USER_SCRIPTS
  - Executes user scripts found in user data.
- action: CONFIG_MOUNTS
  - Mounts ephemeral drives.

Supported User-Data Formats

Cloud-init supports user-data handling of a variety of formats:

- Gzip
  - If user-data is gzip compressed, Cloud-init will decompress the data and handle as appropriate.
- MIME multipart
  - Using a MIMI multipart file you can specify more than one type of data. For example, you could specify both a user-data script and a cloud-config type. Each part of the multipart file can be handled by Cloud-init if it is one of the supported formats.
- Base64 decoding
  - If user-data is base64 encoded, Cloud-init determines if it can understand the decoded data as one of the supported types. If it understands the decoded data, it will decode the data and handle as appropriate. If not, it returns the base64 data intact.
- User-Data script
  - Begins with "#!" or "Content-Type: text/x-shellscript".
  - The script will be executed by "/etc/init.d/Cloud-init-user-scripts" level during first boot. This occurs late in the boot process (after the initial configuration actions were performed).
- Include file
  - Begins with "#include" or "Content-Type: text/x-include-url".
  - This content is an include file. The file contains a list of URLs, one per line. Each of the URLs will be read, and their content will be passed through this same set of rules. The content read from the URL can be gzipped, MIME-multi-part, or plain text.
- Cloud Config Data
  - Begins with "#cloud-config" or "Content-Type: text/cloud-config".
  - This content is cloud-config data. See the examples for a commented example of supported config formats.
- Cloud Boothook
  - Begins with "#cloud-boothook" or "Content-Type: text/cloud-boothook".
  - This content is boothook data. It is stored in a file under /var/lib/cloud and then executed immediately.
  - This is the earliest "hook" available. Note that there is no mechanism provided for running it only once. The boothook must take care of this itself. It is provided with the instance ID in the environment variable INSTANCE_ID. Use this variable to provide a once-per-instance set of boothook data.

Repository Configuration

Amazon Linux AMI is configured with both main and update repositories enabled. When a new Amazon Linux AMI is released, the repository configuration follows the latest release, making it possible for the instance to be upgraded to the latest release with a simple `yum upgrade`. If you wish to prevent your instance from upgrading to the latest release, remove or comment out the `releasever` variable in `/etc/yum.conf`. You can also set the `releasever` variable to a specific AMI version to prevent the upgrade from taking place and to lock in to a specific version. For example, `releasever=2011.09`.

Adding Packages

In addition to the packages included in the Amazon Linux AMI, Amazon provides a yum repository consisting of common Linux applications for use inside of Amazon EC2. The Amazon Linux AMI is configured to point to this repository by default for all yum actions. The packages can be installed by issuing yum commands. For example:

```
#  sudo yum install httpd
```

The packages available from the Amazon Linux AMI yum repository in Amazon EC2 are designed to work with the Amazon Linux AMI. In addition, the Amazon Linux AMI has been built to be binary-compatible with the CentOS series of releases, and therefore packages built to run on CentOS should also run on the Amazon Linux AMI.

Important

Instances running in an Amazon Virtual Private Cloud (VPC) need to have an Internet Gateway attached to the VPC to contact the yum repository. For more information on Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

If you find that the Amazon Linux AMI does not contain an application you need, you can simply install the application directly on your Amazon Linux AMI instance. The Amazon Linux AMI uses RPM and yum for package management, and that will likely be the simplest way to install new applications. You should always check to see if an application is available in our central Amazon Linux AMI repository first because many applications are available there. These applications can easily be added to your AMI instance.

To upload your applications onto an Amazon Linux running instance, use `scp` or `sftp` and then configure the application by logging on to your instance. Your applications can also be uploaded during the instance launch by using the `PACKAGE_SETUP` action from built-in Cloud-init package. For more information, see Cloud-init.

Accessing Source Packages for Reference

You can view the source of packages you have installed inside Amazon EC2 for reference purposes by using tools provided in the Amazon Linux AMI. Source packages are available for all of the packages included in the Amazon Linux AMI and the online package repository. Simply determine the package name for the source package you want to install and use the get_reference_source command to view source within your running instance. For example:

```
# get_reference_source -p  httpd
```

Following is a sample response:

```
Requested package: httpd
Found package from local RPM database: httpd-2.2.20-1.16.amzn1
Corresponding source RPM to found package : httpd-2.2.20-1.16.amzn1.src.rpm

Are these parameters correct? Please type 'yes' to continue: yes
Source RPM downloaded to: /usr/src/srpm/debug/httpd-2.2.20-1.16.amzn1.src.rpm
```

The source RPM will be placed in the /usr/src/srpm/debug directory of your running Amazon EC2 instance. From there it can be unpacked, and, for reference, you can view the source tree using standard RPM tools. After you finish debugging, the package will be available for use in Amazon EC2.

Important

Instances running in an Amazon Virtual Private Cloud (Amazon VPC) need to have an Internet Gateway attached to the VPC to contact the yum repository. For information on Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

Developing Applications

A full set of Linux development tools is provided in the yum repository for the Amazon Linux AMI. To develop applications on the Amazon Linux AMI, simply select the development tools you need with yum. Alternatively, many applications developed on CentOS and other similar distributions should run on the Amazon Linux AMI.

Instance Store Access

The instance store drive ephemeral0 is mounted in /media/ephmeral0 only on Amazon instance store-backed AMIs. This is different than many other images that mount the instance store drive under /mnt.

Product Life Cycle

You can get started by launching the latest Amazon Linux AMI in Amazon EC2. The Amazon Linux AMI will be updated regularly with security and feature enhancements. If you do not need to preserve data or customizations on your running Amazon Linux AMI instances, you can simply relaunch new instances with the latest updated Amazon Linux AMI. If you do need to preserve data or customizations on your running Amazon Linux AMI instances, you can maintain those instances through the Amazon Linux AMI yum repositories. As the Amazon Linux AMI is updated, all of the updated packages will also be provided through these repositories, and you can chose to apply these updates to your running instances. Older versions of the AMI and update packages will continue to be available for launch in Amazon EC2, even as new Amazon Linux AMI versions are released. However, in some cases, if you're seeking support for an older version of the Amazon Linux AMI through Amazon Premium Support, we might ask you to move to newer versions as part of the support process.

⚠ **Important**

Instances running in an Amazon Virtual Private Cloud (Amazon VPC) need to have an Internet Gateway attached to the VPC to contact the yum repository. For information on adding an Internet Gateway to your Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

Security Updates

Security updates are provided via the Amazon Linux AMI yum repositories and updated Amazon Linux AMIs. Security alerts will be published in the Amazon Linux AMI Security Center. For more information on AWS security policies or to report a security problem, visit the AWS Security Center.

Amazon Linux AMIs are configured to download and install security updates at launch time. This is controlled via a Cloud-init setting called `repo_upgrade`. The following snippet of cloud-init configuration shows how you can change the settings in the user data text you pass to your instance initialization:

```
# cloud-config
repo_upgrade:security
```

The possible values for the `repo_upgrade` setting are as follows:

- security
  - Apply outstanding updates that Amazon marks as security updates.
- bugfix
  - Apply updates that Amazon marks as bug fixes. Bug fixes are a larger set of updates, which include security updates and fixes for various other minor bugs.

- all
  - Apply all applicable available updates, regardless of their classification.
- none
  - Do not apply any updates to the instance on startup.

The default setting for `repo_upgrade` is security. That is, if you don't specify a different value in your user data, by default the Amazon Linux AMI will perform the security upgrades at launch for any packages installed at that time. Amazon Linux AMI will also notify you of any updates to the installed packages by listing the number of available updates upon login using the `motd` . To install these updates, you will need to run `sudo yum upgrade` on the instance.

Important

Instances running in an Amazon Virtual Private Cloud (Amazon VPC) need to have an Internet Gateway attached to the VPC to contact the yum repository. For information on adding an Internet Gateway to Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

Support

Support for installation and use of the base Amazon Linux AMI is included through subscriptions to AWS Premium Support. For more information, go to Premium Support.

You're encouraged to post any questions you have on using the Amazon Linux AMI to the Amazon EC2 forums.

You can report bugs either to Premium Support or the Amazon EC2 forums.

Enabling Your Own Linux Kernels

Topics

Amazon EC2 allows you to load a paravirtual Linux kernel within an Amazon Machine Image (AMI) or Amazon EBS volume. You have the option to create images that contain a kernel and initrd (initial RAM disk), and behave in a manner that is closer to traditional virtual or physical Linux installations. By enabling you to boot from the kernel within volumes, this feature allows you to seamlessly upgrade the kernel on Amazon EC2 instances. We expect that AMI providers will update their AMIs to use this new feature, and most Amazon EC2 users will be able to begin managing their own kernels when these updated AMIs become available. Your AMI provider can tell you when it plans to support this feature. However, if you want to begin managing your own kernel now, the following section shows how. This process assumes general knowledge of Amazon EC2 AMI bundling and registration, as well as knowledge of how to install kernel packages and configure GRUB on your Linux systems.

PV-GRUB: A New Amazon Kernel Image

To enable user-provided kernels, Amazon has published Amazon Kernel Images (AKIs) that use a system called PV-GRUB. PV-GRUB is a paravirtual “mini-OS” that runs a version of GNU GRUB, the standard Linux boot loader. PV-GRUB selects the kernel to boot by reading /boot/grub/menu.lst from your image. It will load the kernel specified by your image and then shut down the “mini-OS,” so that it no longer consumes any resources. One of the advantages of this solution is that PV-GRUB understands standard grub.conf or menu.lst commands, which allows it to work with most existing Linux distributions.

The following task list describes what you need to do to enable an AMI to use PV-GRUB AKI to run a user-provided kernel.

Enabling an AMI to Run A User-Provided Kernel

1	Install an Amazon EC2-compatible kernel.
2	Generate an initrd.
3	Populate /boot/grub/menu.lst referencing your kernel.
4	Select an appropriate AKI ID from the Amazon Kernel Image IDs section that follows.
5	Bundle the AMI and set the default to your chosen AKI.
6	Upload and register your new AMI.
7	For existing AMIs, you can simply specify the appropriate AKI ID when you call RunInstances or when you use the AWS Management Console.

Note

To update an existing AMI to use a user-provided kernel, re-launch the AMI and follow steps 1 through 6 specified in the preceding task list.

For procedures associated with the preceding task list, see Using the User-Provided Kernel (p. 66).

Configuring the GRUB

In order for PV-GRUB to boot, a GRUB menu.lst file must exist in the image. For most distributions, you have two options for the GRUB configuration:

- **Option 1:** Install GRUB and allow the default kernel installation scripts to handle the installing and updating the GRUB configuration. The steps necessary to install GRUB will vary depending on your Linux distribution, but typically GRUB will be available as a package you can install online.
- **Option 2:** Populate a general /boot/grub/menu.lst. An example of a menu.lst configuration file for booting an AMI with a PV-GRUB AKI follows.

  Important

  Your must modify your own menu.lst for your specific environment.

```
default 0
timeout 3
fallback 1

title Vanilla EC2 Kernel 2.6.32.10
 root (hd0)
 kernel /boot/vmlinux-2.6.32.10-ACME_SYS_EC2 root=/dev/sda1
 initrd /boot/initrd-2.6.32.10-ACME_SYS_EC2

title Ubuntu EC2 2.6.32.302-EC
 root (hd0)
 kernel /boot/ubuntu-ec2 root=/dev/sda1
 initrd /boot/initrd-ec2
```

We recommend that you use option two to control the kernel booting for two reasons. First, Amazon EC2 users don't have interactive control over the boot process because there is no keyboard access. GRUB will proceed without user interaction. Second, and most important for Amazon EC2 instances, you want to protect against distributions that auto-update the default kernel and break your image. By not relying on the auto-update mechanism and explicitly choosing which kernel you run, you reduce the risk of an incompatible kernel becoming the default kernel.

A fallback kernel does not have to be specified in your menu.lst, but we recommend that you have a fallback when you test new new kernels. GRUB can fall back to another kernel in the event that the new kernel fails. Having a fallback kernel allows the instance to boot even if the new kernel is not found.

Amazon EBS Volumes on a PV-GRUB-Enabled AMI

There are two special things you should consider when you use a PV-GRUB-enabled image to mount EBS volumes. First, for Amazon EBS volumes the first partition must be a boot partition. Second, if you plan to use a logical volume manager (LVM) with Amazon EBS volumes, you need a separate boot partition outside of the LVM. Then you can create logical volumes with the LVM. PV-GRUB expects to find the menu.lst in /boot/grub. As a result, if the boot partition is mounted in at /boot, menu.lst will be found in /boot/boot/grub.

Amazon Kernel Image IDs

Several PV-GRUB AKIs are available depending on the type and location of your instance. There are AKIs for 32-bit and 64-bit architecture types, with each having one AKI for partitioned images and another AKI for partitionless images. You must choose an AKI with "hd0" in the name if you want a raw or unpartitioned disk image (most images). Choose an AKI with "hd00" in the name if you want an image that has a partition table.

Most vendors, such as Fedora, Red Hat, Ubuntu, and Novell, use unpartitioned disk images. This means that they use the hd0 variants of PV-GRUB; almost without exception most users will want to use the hd0 variants.

Note

You cannot use the 64-bit version of PV-GRUB to start a 32-bit kernel or vice versa.

The following AKI IDs should be used by users who are either registering new AMIs or who want to launch existing AMIs using PV-GRUB. Each AKI type is available in all Amazon EC2 Regions:

- **US-East-1**

  aki-4c7d9525 ec2-public-images/pv-grub-hd00-V1.01-i386.gz.manifest.xml

  aki-4e7d9527 ec2-public-images/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml

aki-407d9529 ec2-public-images/pv-grub-hd0-V1.01-i386.gz.manifest.xml
aki-427d952b ec2-public-images/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
aki-525ea73b ec2-public-images/pv-grub-hd00_1.02-i386.gz.manifest.xml
aki-8e5ea7e7 ec2-public-images/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
aki-805ea7e9 ec2-public-images/pv-grub-hd0_1.02-i386.gz.manifest.xml
aki-825ea7eb ec2-public-images/pv-grub-hd0_1.02-x86_64.gz.manifest.xml

- **US-West-1**
  aki-9da0f1d8 ec2-public-images-us-west-1/pv-grub-hd00-V1.01-i386.gz.manifest.xml
  aki-9fa0f1da ec2-public-images-us-west-1/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml
  aki-99a0f1dc ec2-public-images-us-west-1/pv-grub-hd0-V1.01-i386.gz.manifest.xml
  aki-9ba0f1de ec2-public-images-us-west-1/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
  aki-87396bc2 ec2-public-images-us-west-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
  aki-81396bc4 ec2-public-images-us-west-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-83396bc6 ec2-public-images-us-west-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-8d396bc8 ec2-public-images-us-west-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
- **US-West-2**
  aki-dee26fee ec2-public-images-us-west-2/pv-grub-hd00-V1.01-i386.gz.manifest.xml
  aki-90e26fa0 ec2-public-images-us-west-2/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml
  aki-dce26fec ec2-public-images-us-west-2/pv-grub-hd0-V1.01-i386.gz.manifest.xml
  aki-ace26f9c ec2-public-images-us-west-2/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
  aki-c0e26ff0 ec2-public-images-us-west-2/pv-grub-hd00_1.02-i386.gz.manifest.xml
  aki-94e26fa4 ec2-public-images-us-west-2/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-c2e26ff2 ec2-public-images-us-west-2/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-98e26fa8 ec2-public-images-us-west-2/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
- **EU-West-1**
  aki-47eec433 ec2-public-images-eu/pv-grub-hd00-V1.01-i386.gz.manifest.xml
  aki-41eec435 ec2-public-images-eu/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml
  aki-4deec439 ec2-public-images-eu/pv-grub-hd0-V1.01-i386.gz.manifest.xml
  aki-4feec43b ec2-public-images-eu/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
  aki-8a6657fe ec2-public-images-eu/pv-grub-hd00_1.02-i386.gz.manifest.xml
  aki-60695814 ec2-public-images-eu/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-64695810 ec2-public-images-eu/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-62695816 ec2-public-images-eu/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
- **AP-SouthEast-1**
  aki-6fd5aa3d ec2-public-images-ap-southeast-1/pv-grub-hd00-V1.01-i386.gz.manifest.xml
  aki-6dd5aa3f ec2-public-images-ap-southeast-1/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml
  aki-13d5aa41 ec2-public-images-ap-southeast-1/pv-grub-hd0-V1.01-i386.gz.manifest.xml
  aki-11d5aa43 ec2-public-images-ap-southeast-1/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
  aki-a0225af2 ec2-public-images-ap-southeast-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
  aki-a6225af4 ec2-public-images-ap-southeast-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-a4225af6 ec2-public-images-ap-southeast-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-aa225af8 ec2-public-images-ap-southeast-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
- **AP-NorthEast-1**
  aki-d209a2d3 ec2-public-images-ap-northeast-1/pv-grub-hd0-V1.01-i386.gz.manifest.xml
  aki-d409a2d5 ec2-public-images-ap-northeast-1/pv-grub-hd0-V1.01-x86_64.gz.manifest.xml
  aki-d609a2d7 ec2-public-images-ap-northeast-1/pv-grub-hd00-V1.01-i386.gz.manifest.xml
  aki-d809a2d9 ec2-public-images-ap-northeast-1/pv-grub-hd00-V1.01-x86_64.gz.manifest.xml

aki-e85df7e9 ec2-public-images-ap-northeast-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
aki-ea5df7eb ec2-public-images-ap-northeast-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
aki-ec5df7ed ec2-public-images-ap-northeast-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
aki-ee5df7ef ec2-public-images-ap-northeast-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml

- **US-East-1**
  aki-825ea7eb ec2-public-images/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-805ea7e9 ec2-public-images/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-8e5ea7e7 ec2-public-images/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-525ea73b ec2-public-images/pv-grub-hd00_1.02-i386.gz.manifest.xml
- **US-West-1**
  aki-8d396bc8 ec2-public-images-us-west-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-83396bc6 ec2-public-images-us-west-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-81396bc4 ec2-public-images-us-west-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-87396bc2 ec2-public-images-us-west-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
- **US-West-2**
  aki-c0e26ff0 ec2-public-images-us-west-2/pv-grub-hd00_1.02-i386.gz.manifest.xml
  aki-94e26fa4 ec2-public-images-us-west-2/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-c2e26ff2 ec2-public-images-us-west-2/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-98e26fa8 ec2-public-images-us-west-2/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
- **EU-West-1**
  aki-62695816 ec2-public-images-eu/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-64695810 ec2-public-images-eu/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-60695814 ec2-public-images-eu/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-8a6657fe ec2-public-images-eu/pv-grub-hd00_1.02-i386.gz.manifest.xml
- **AP-SouthEast-1**
  aki-aa225af8 ec2-public-images-ap-southeast-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-a4225af6 ec2-public-images-ap-southeast-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-a6225af4 ec2-public-images-ap-southeast-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-a0225af2 ec2-public-images-ap-southeast-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
- **AP-NorthEast-1**
  aki-ee5df7ef ec2-public-images-ap-northeast-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-ec5df7ed ec2-public-images-ap-northeast-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-ea5df7eb ec2-public-images-ap-northeast-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-e85df7e9 ec2-public-images-ap-northeast-1/pv-grub-hd00_1.02-i386.gz.manifest.xml
- **SA-East-1**
  aki-cc3ce3d1 ec2-public-images-sa-east-1/pv-grub-hd0_1.02-x86_64.gz.manifest.xml
  aki-bc3ce3a1 ec2-public-images-sa-east-1/pv-grub-hd0_1.02-i386.gz.manifest.xml
  aki-d23ce3cf ec2-public-images-sa-east-1/pv-grub-hd00_1.02-x86_64.gz.manifest.xml
  aki-823ce39f ec2-public-images-sa-east-1/pv-grub-hd00_1.02-i386.gz.manifest.xml

Compatible PV-GRUB Kernels

There are a number of Linux distributions that have compatible Amazon EC2 kernels. The following is a brief, non-comprehensive list of kernels that we have worked with the maintainers to test:

- Fedora 8-9 Xen kernels
- Fedora 13 (2.6.33.6-147 and higher)
- Fedora 14 and later
- SLES/openSUSE 10.x, 11.0, 11.1 Xen
- SLES/openSUSE 11.x EC2 Variant
- Ubuntu EC2 Variant kernels
- Ubuntu 11.04 and later
- RHEL 5.x kernels
- RHEL 6.x kernels
- CentOS 5.x kernels
- CentOS 6.x kernels
- Gentoo (see FAQ)

It is possible that your specific Linux kernel will not boot using the new PV-GRUB method. If you find that to be the case, you should select a different kernel or use a non-PV-GRUB AKI to boot your instance.

Note

Kernels that disable the pv-ops XSAVE hypercall are known to work on all instance types, whereas those that enable this hypercall will fail to launch in some cases. Similarly, non-pv-ops kernels that do not adhere to the Xen 3.0.2 interface might fail to launch in some cases.

PV-GRUB Supported File Systems

Since PV-GRUB is a paravirtual version of GRUB 0.97, it has all the limitations of GRUB. Most importantly, this means that it will not work properly for certain disk layouts or file system types. The following are the /boot file systems that PV-GRUB can boot from:

- EXT2/3/4
- XFS
- ReiserFS
- BTRFS (beta)

Note

These are the /boot file systems that we have tested and verified. Others could boot from PV-GRUB, but haven’t been tested.

Using the User-Provided Kernel

The following procedure gives an example of how to enable a openSUSE AMI to use the PV-GRUB AKI to run a user-provided kernel by rebundling from a running instance.

Important

The specific details of configuring your AMI to use PV-GRUB will vary depending on your exact Linux environment. The following example is for openSUSE 11.2.

To use the PV-GRUB AKI with an openSUSE AMI

1. Install an Amazon EC2-compatible kernel from the command line on your running Linux instance.

```

 # rpm –ivh /tmp/kernel-ec2-2.6.35-rc4.8.1.x86_64.rpm

 warning: /tmp/kernel-ec2-2.6.35-rc4.8.1.x86_64.rpm: Header V3 DSA signature:
 NOKEY, key ID a29f6635
Preparing...                ###########################################
[100%]
   1:kernel-ec2             ###########################################
[100%]

Kernel image:   /boot/vmlinux-2.6.35-rc4-8-ec2
Initrd image:   /boot/initrd-2.6.35-rc4-8-ec2
Root device:    /dev/sda1 (mounted on / as ext3)

Features:       block
14807 blocks
```

2. Generate an initrd on your running Linux instance.

```
# mkinitrd
Kernel image:   /boot/vmlinux-2.6.35-rc4-8-ec2
Initrd image:   /boot/initrd-2.6.35-rc4-8-ec2
Root device:    /dev/sda1 (mounted on / as ext3)
Features:       block
14806 blocks
```

3. Populate /boot/grub/menu.lst referencing your kernel on your running Linux instance.

   Important

   Your must modify your own menu.lst for your specific environment.

```
default 0
timeout 3

title EC2
        root (hd0)
        kernel /boot/vmlinux-ec2 root=/dev/sda1
        initrd /boot/initrd-ec2
```

4. Select an appropriate AKI ID from the Amazon Kernel Image IDs section that follows. For this host, we’ve chosen aki-825ea7eb because we are bundling an AMI.
5. For new AMIs or Amazon EBS volumes, bundle the AMI and set the default to your chosen AKI from your running Linux instance.

```
# ec2-bundle-vol -r x86_64 -d /mnt -p openSUSE-11.2-PV-GRUB -u [AWS-ID] -k
 /mnt/pkey.pem -c /mnt/cert.pem -s 10240 -e /mnt,/root/.ssh  --kernel  aki-
825ea7eb
```

6. Upload the bundle from your running Linux instance to Amazon S3.

```
# ec2-upload-bundle -b MyReallyCoolBucketLocation -m /mnt/openSUSE-11.2-PV
GRUB.manifest.xml -a MyAccessKey -s MySecretKey
```

7. Register the AMI with the AKI (aki-825ea7eb) from your desktop using the Amazon EC2 command line tools.

```
$ ec2-register –-name openSUSE-11.2-PVGRUB MyReallyCoolBucketLoca
tion/openSUSE-11.2-PVGRUB.manifest.xml
```

Amazon Support for PV-GRUB

Amazon supports the use of PV-GRUB to load a kernel of your choice for your AMI. However, we cannot provide support for your kernel itself or failures caused by the use of a kernel that does not meet the requirements of PV-GRUB. To avoid the situation in which a kernel does not work consistently or at all, we recommend that you use a known good kernel, select a non-PV-GRUB AKI, or seek support from your AMI vendor. Unfortunately, due to the wide and varied kernel landscape, it is impossible for Amazon to provide support for all kernel varieties.

Technical Notes for Advanced Users

The following information is provided to assist those who are familiar with compiling kernels. Many Linux distributions provide documentation on how to compile a kernel in kernel source packages. For users compiling 2.6.30+ kernels, Amazon recommends using the PVOps method described below. Amazon is unable to offer support for compiling your own kernel.

openSUSE and Gentoo Kernels

Both openSUSE and Gentoo provide native source code that enables compiling an Amazon EC2 capable kernel. For openSUSE, the configuration options are part of the mainline 11.2 and higher, while Gentoo provides a Xen-compatible in the "xen-sources" portage package. For both distributions, please use the following kernel configuration options:

CONFIG_XEN_COMPAT_00002_AND_LATER=y
CONFIG_XEN_COMPAT=0x030002
CONFIG_HOTPLUG_CPU=y

Option 1: Patching vanilla or other distribution kernels (apply Xen DomU patchset)

The process of patching any kernel (vanilla or distribution) is an advanced topic. For instructions, see “kernel/Documentation/applying-patches.txt” under the kernel source tree for instructions.

The same patchset that is used with openSUSE and Gentoo can be applied to a vanilla kernel.org kernel.

http://-code.google.com/-p/-gentoo-xen-kernel/-downloads/-list
2.6.29: http://-lists.xensource.com/-archives/-html/-xen-users/-2009-04/-msg00004.html
http://-lists.xensource.com/-archives/-html/-xen-devel/-2009-06/-msg00127.html
2.6.30: http://-lists.xensource.com/-archives/-html/-xen-devel/-2009-07/-msg00027.html
2.6.31: http://-lists.xensource.com/-archives/-html/-xen-users/-2009-10/-msg00342.html
2.6.31-9: http://-lists.xensource.com/-archives/-html/-xen-users/-2009-12/-msg00411.html
2.6.31-10: http://-lists.xensource.com/-archives/-html/-xen-users/-2010-01/-msg00032.html
2.6.31-14 and 2.6.32-1 :http://-lists.xensource.com/-archives/-html/-xen-users/-2010-04/-msg00091.html

Option 2: Patching vanilla or other distribution kernels (PVOps)

For PVOps kernels, you can elect to disable the XSAVE hypercall in the guest. The following patch works against 2.6.32 through 2.6.35 kernels.

```
---
 arch/x86/xen/enlighten.c |    1 +
 1 files changed, 1 insertions(+), 0 deletions(-)

diff --git a/arch/x86/xen/enlighten.c b/arch/x86/xen/enlighten.c
index 52f8e19..6db3d67 100644
--- a/arch/x86/xen/enlighten.c
+++ b/arch/x86/xen/enlighten.c
@@ -802,6 +802,7 @@ static void xen_write_cr4(unsigned long cr4)
 {
  cr4 &= ~X86_CR4_PGE;
  cr4 &= ~X86_CR4_PSE;
+ cr4 &= ~X86_CR4_OSXSAVE;

  native_write_cr4(cr4);
 }
--
1.6.6.1
```

Using Instances

Topics

After an AMI is launched, the resulting running system is called an instance. This section gives information about instances and their basic characteristics. It also describes how to launch an instance and connect to it.

Instance Families and Types

Topics

Amazon EC2 instances are grouped into the general families described in the following table.

Family	Description
Standard	Have memory-to-CPU ratios suitable for most general purpose applications

Family	Description
Micro	Provide a small amount of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically (for more information, see Micro Instances (p. 328))
High-CPU	Have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications
High-Memory	Have proportionally more memory resources and are well suited for high throughput applications, such as database and memory caching applications
Cluster Compute	Have a very large amount of CPU coupled with increased networking performance, making them well suited for High Performance Compute (HPC) applications and other demanding network-bound applications (for more information, see Cluster Instance Concepts (p. 335))
Cluster GPU	Provide general-purpose graphics processing units (GPUs), with proportionally high CPU and increased network performance for applications that benefit from highly parallelized processing. They're well suited for HPC applications as well as rendering and media processing applications (for more information, see Cluster Instance Concepts (p. 335))

Tip

One of the advantages of Amazon EC2 is that you pay by the instance hour, which makes it convenient and inexpensive to test the performance of your application on different instance families and types. A good way to determine the most appropriate instance family and instance type is to launch test instances and benchmark your application.

Available Instance Types

When you launch an instance, you specify the instance type (the value in the *Name* column in the following table). We launch an m1.small if you don't specify a particular instance type.

Type	CPU	Memory	Local Storage	Platform	I/O	Name
Small	1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)	1.7 GB	160 GB instance storage (150 GB plus 10 GB root partition)	32-bit	Moderate	m1.small
Large	4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each)	7.5 GB	850 GB instance storage (2 x 420 GB plus 10 GB root partition)	64-bit	High	m1.large
Extra Large	8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each)	15 GB	1690 GB instance storage (4 x 420 GB plus 10 GB root partition)	64-bit	High	m1.xlarge

Type	CPU	Memory	Local Storage	Platform	I/O	Name
Micro	Up to 2 EC2 Compute Units (for short periodic bursts)	613 MB	None (use Amazon EBS volumes for storage)	32-bit or 64-bit	Low	t1.micro
High-CPU Medium	5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)	1.7 GB	350 GB instance storage (340 GB plus 10 GB root partition)	32-bit	Moderate	c1.medium
High-CPU Extra Large	20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each)	7 GB	1690 GB instance storage (4 x 420 GB plus 10 GB root partition)	64-bit	High	c1.xlarge
High-Memory Extra Large	6.5 EC2 Compute Units (2 virtual cores with 3.25 EC2 Compute Units each)	17.1 GB	420 GB instance storage (1 x 420 GB)	64-bit	Moderate	m2.xlarge
High-Memory Double Extra Large	13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each)	34.2 GB	850 GB instance storage (1 x 840 GB plus 10 GB root partition)	64-bit	High	m2.2xlarge
High-Memory Quadruple Extra Large	26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each)	68.4 GB	1690 GB instance storage (2 x 840 GB plus 10 GB root partition)	64-bit	High	m2.4xlarge
Cluster Compute Quadruple Extra Large	33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture)	23 GB	1690 GB instance 64-bit storage (2 x 840 GB plus 10 GB root partition)	64-bit	Very high (10 Gbps Ethernet)	cc1.4xlarge
Cluster Compute Eight Extra Large	88 EC2 compute units (2 x Intel Xeon CPU with 8 cores)	60.5 GB	3370 GB instance (4 x 840 GB plus 10 GB root partition)	64-bit	Very high (10 Gbps Ethernet)	cc2.8xlarge
Cluster GPU Quadruple Extra Large	33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture), plus 2 NVIDIA Tesla M2050 "Fermi" GPUs	22 GB (see note after this table)	1690 GB instance (2 x 840 GB plus 10 GB root partition)	64-bit	Very high (10 Gbps Ethernet)	cg1.4xlarge

Note

The cg1.x4large instance type has 23 GB of memory, with 1 GB reserved for GPU operation. The 22 GB doesn't include the on-board memory of the GPUs, which is 3 GB per GPU for the NVIDIA Tesla M2050.

Windows Instance Types

Amazon EC2 instances can run Microsoft Windows Server 2003, Microsoft Windows Server 2008 or Microsoft Windows Server 2008 R2. The Windows AMIs provide you with all standard Microsoft Windows Server functionality.

Using Amazon EC2 instances running Windows is similar to using instances running Linux/UNIX. The following are the major differences between instances that use Linux/UNIX and Windows:

- **Remote Desktop—**To access Windows instances, you use Remote Desktop instead of SSH.
- **Administrative Password—**To access Windows instances the first time, you must obtain the administrative password (available through the AWS Management Console, the command line tools, or the EC2 API).
- **Bundling—**Amazon instance store-backed Windows instances use different bundling procedures than Amazon instance store-backed Linux/UNIX instances. For more information, go to Creating an Instance Store-Backed Windows AMI in the *Amazon EC2 Microsoft Windows Guide* .

Amazon EC2 currently provides the following Windows AMIs:

- Microsoft Windows Server 2003 (32-bit)
- Microsoft Windows Server 2003 (64-bit)
- Microsoft Windows Server 2008 (32-bit)
- Microsoft Windows Server 2008 (64-bit)
- Microsoft Windows Server 2008 R2 (64-bit)

The Windows public AMIs that Amazon provides are unmodified versions of Windows with the following two exceptions: we added drivers to improve the networking and disk I/O performance and we created the Amazon EC2 configuration service. The Amazon EC2 configuration service performs the following functions:

- Randomly sets the Administrator password on initial launch, encrypts the password with the user’s SSH key, and reports it to the console. This operation happens upon initial AMI launch. If you change the password, AMIs that are created from this instance use the new password.
- Configures the computer name to the internal DNS name. To determine the internal DNS name, see Using Instance IP Addresses (p. 289).
- Sends the last three system and application errors from the event log to the console. This helps developers to identify problems that caused an instance to crash or network connectivity to be lost.

For more information about the EC2 configuration service, see Appendix C: Windows Configuration Service (p. 424).

Compute Resources Measurement

Transitioning to a utility computing model changes how developers are trained to think about CPU resources. Instead of purchasing or leasing a particular processor to use for several months or years, you are renting capacity by the hour. Because Amazon EC2 is built on commodity hardware, over time there might be several different types of physical processors underlying different virtual EC2 instances. Our goal is to provide a consistent amount of CPU capacity regardless of the actual underlying hardware.

Amazon EC2 uses a variety of measures to provide each instance with a consistent and predictable amount of CPU capacity. To make it easy for developers to compare CPU capacity between different instance types, we defined an Amazon EC2 Compute Unit. One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor. This is also the equivalent to an early-2006 1.7 GHz Xeon processor referenced in our original documentation.

Note

We use several internal benchmarks and tests to manage the consistency and predictability of the performance of an Amazon EC2 Compute Unit. For more information, go to the Instance page.

To find out which instance works best for your application, we recommend launching an instance and using your own benchmark application. This helps you determine which instance type works best for your specific use case.

I/O Resources

Amazon EC2 provides virtualized server instances. Whereas some resources like CPU, memory, and instance storage are dedicated to a particular instance, other resources such as the network and the disk subsystem are shared among instances. If each instance on a physical host tries to use as much of one of these shared resources as possible, each receives an equal share of that resource. However, when a resource is under-utilized, you are often able to consume a higher share of that resource while it is available.

The different instance types provide higher or lower minimum performance from the shared resources depending on their size. Each of the instance types has an I/O performance indicator (low, moderate, high, etc.). Instance types with high I/O performance have a larger allocation of shared resources. Allocating larger share of shared resources also reduces the variance of I/O performance. For most applications, moderate I/O performance is more than enough. However, for applications that require greater or more consistent I/O performance, consider instances with higher I/O performance.

Instance Tags

To help categorize and manage your instances, you can assign *tags* of your choice to them. For more information, see Using Tags (p. 275) .

Instance Usage

The instance is your basic computation building block. Amazon EC2 offers multiple instance types from which you can choose. You can run as many or as few instances as you need at any given time.

Note

By default, you can run up to 20 instances. If you need more than 20 instances, please complete the Amazon EC2 Instance Request Form.

For information about available instance types, see Instance Families and Types (p. 69).

Once launched, an instance looks very much like a traditional host. You have complete control of your instances; you have root access to each one and you can interact with them as you would any machine.

Here are some suggestions for making the best use of Amazon EC2 instances:

- Do not rely on an instance's local storage for valuable, long-term data.
  When instances fail, the data on the local disk is lost. Use a replication strategy across multiple instances to keep your data safe, or store your persistent data in Amazon S3, or use Amazon EBS.
- Define images based on the type of work they perform.
  For "Internet applications," you might define one image for database instances and another for web servers. Image creation and storage are cheap and easy operations, so you can individualize and customize as necessary. Specialized images can result in smaller AMI sizes, which boot considerably faster.
- Monitor the health of your instances.
  For more information, go to the Amazon CloudWatch product page.
- Keep your Amazon EC2 firewall permissions as restrictive as possible.
  Only open up permissions that you require. Use separate *security groups* to deal with instances that have different security requirements. Consider using additional security measures inside your instance (such as using your own firewall). If you need to log in interactively (SSH), consider creating a bastion security group that allows external login and keep the remainder of your instances in a group that does not allow external login. For more information about security groups, see Using Security Groups (p. 304)
  .

Launching and Using Instances

Topics

This section describes how to launch *instances* and retrieve instance-specific data from within the instance. It also covers launching *shared AMIs* and security risks associated with running shared AMIs.

Note

If you create an AMI in one Region, you cannot launch it in another Region without migrating it. For information on Regions, see Region and Availability Zone Concepts (p. 321). For information on migrating AMIs, refer to the `ec2-migrate-bundle` section in the Amazon Elastic Compute Cloud Command Line Reference.

Related Topics

- Using AMIs (p. 15)
- Using Instances (p. 69)

Finding a Suitable AMI

This section describes how to find an AMI.

AWS Management Console

To find a suitable AMI

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **AMIs** in the **Navigation** pane.
   The console displays your AMIs and all public AMIs.
3. To reduce the number of displayed AMIs, select options from the **Viewing** list boxes. For example, you might want to display Amazon images.
4. After locating your desired AMI, write down its AMI ID. You can use this to launch instances of the AMI or register your own AMI, using this as a baseline.

Command Line Tools

To find a suitable AMI

1. Use the **ec2-describe-images** command.

   Tip

   You can filter this list to return only certain types of AMIs of interest to you. For more information about how to filter the results, go to ec2-describe-images in the *Amazon Elastic Compute Cloud Command Line Reference*.

   The command lists your AMIs and Amazon's public AMIs. The following example shows only part of the resulting output.

```
PROMPT> ec2-describe-images -o self -o amazon

IMAGE ami-d8699bb1 amazon/ami-vpc-nat-1.0.0-beta.i386-ebs amazon available
public  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-33d88c5f 8
IMAGE ami-c6699baf amazon/ami-vpc-nat-1.0.0-beta.x86_64-ebs amazon available
 public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-57d88c3b 8
IMAGE ami-30f30659 amazon/amzn-ami-0.9.7-beta.i386-ebs amazon available public
  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-d895cdb3 10
IMAGE ami-0af30663 amazon/amzn-ami-0.9.7-beta.x86_64-ebs amazon available
public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-f295cd99 10
IMAGE ami-3ac33653 amazon/amzn-ami-0.9.8-beta.i386-ebs amazon available public
  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-14ba967f 10
IMAGE ami-38c33651 amazon/amzn-ami-0.9.8-beta.x86_64-ebs amazon available
public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-10b9957b 10
IMAGE ami-08728661 amazon/amzn-ami-0.9.9-beta.i386-ebs amazon available public
  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-674a930d 10
IMAGE ami-2272864b amazon/amzn-ami-0.9.9-beta.x86_64-ebs amazon available
public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-8926ffe3 10
IMAGE ami-76f0061f amazon/amzn-ami-2010.11.1-beta.i386-ebs amazon available
public  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-cba692a1 8
IMAGE ami-74f0061d amazon/amzn-ami-2010.11.1-beta.x86_64-ebs amazon available
 public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-ffa69295 8
IMAGE ami-8c1fece5 amazon/amzn-ami-2011.02.1.i386-ebs amazon available public
  i386 machine aki-407d9529   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-22fc264e 8
IMAGE ami-8e1fece7 amazon/amzn-ami-2011.02.1.x86_64-ebs amazon available
public  x86_64 machine aki-427d952b   ebs paravirtual xen
BLOCKDEVICEMAPPING /dev/sda1  snap-a6fc26ca 8
```

   In the following sections, we're going to launch an instance of the AMI with ID `ami-b232d0db`.

API

To find a suitable AMI

- Construct the following Query request, which returns all Amazon-owned AMIs:

```
https://ec2.amazonaws.com/
?Action=DescribeImages
&User.1=amazon
&...auth parameters...
```

Following is an example response.

```
<DescribeImagesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <imagesSet>
        <item>
          <imageId>ami-8c1fece5</imageId>
         <imageLocation>amazon/amzn-ami-2011.02.1.i386-ebs</imageLocation>

          <imageState>available</imageState>
          <imageOwnerId>137112412989</imageOwnerId>
          <isPublic>true</isPublic>
          <architecture>i386</architecture>
          <imageType>machine</imageType>
          <kernelId>aki-407d9529</kernelId>
          <imageOwnerAlias>amazon</imageOwnerAlias>
          <name>amzn-ami-2011.02.1.i386-ebs</name>
          <description>Amazon Linux AMI i386 EBS</description>
          <rootDeviceType>ebs</rootDeviceType>
          <rootDeviceName>/dev/sda1</rootDeviceName>
          <blockDeviceMapping>
            <item>
              <deviceName>/dev/sda1</deviceName>
              <ebs>
                <snapshotId>snap-22fc264e</snapshotId>
                <volumeSize>8</volumeSize>
                <deleteOnTermination>true</deleteOnTermination>
              </ebs>
            </item>
          </blockDeviceMapping>
          <virtualizationType>paravirtual</virtualizationType>
          <hypervisor>xen</hypervisor>
        </item>
  </imagesSet>
```

Getting an SSH Key Pair

Topics

- How to Have AWS Create the Key Pair for You (p. 80)

Public AMI instances have no password, and you need a public/private key pair to log in to them. The public key half of this pair is embedded in your instance, allowing you to use the private key to log in securely without a password. After you create your own AMIs, you can choose other mechanisms to securely log in to your new instances.

You can have multiple key pairs, and each key pair requires a name. Be sure to choose a name that is easy to remember.

You have two options for getting a key pair:

- Generate it yourself.

  You can use a third-party tool such as OpenSSH, and then import the public key to AWS using either the `ec2-import-keypair` command or the `ImportKeyPair` action.

- Have AWS generate it for you.

  You can use the AWS Management Console, the `ec2-add-keypair` command, or the `CreateKeyPair` action.

AWS doesn't store a copy of the private key for either option. Amazon EC2 only stores the public key, and associates it with a friendly name that you specify for the key pair.

Note

If you are using PuTTY in Windows, you must convert the private key to PuTTY's format. For more information on using PuTTy with Amazon EC2, see Connecting to Linux/UNIX Instances from Windows Using PuTTY (p. 117).

How to Generate Your Own Key and Import It to AWS

This section describes how to import a public key to AWS from a key pair you've created with a third-party tool.

You can easily create an RSA key pair on Windows or Linux using the `ssh-keygen` command line tool (provided with the standard OpenSSH installation). Java, Ruby, Python, and many other programming languages provide standard libraries for RSA key pair creation.

EC2 accepts the following formats:

- OpenSSH public key format (e.g., the format in ~/.ssh/authorized_keys)
- Base64 encoded DER format
- SSH public key file format as specified in RFC4716

EC2 does not accept DSA keys. Make sure your key generator is set up to create RSA keys.

Supported lengths: 1024, 2048, and 4096.

Command Line Tools

To import a public key

1. Generate the key pair with a third-party tool of your choice.

2. Use `ec2-import-keypair` to import the public key file to AWS. The following example names the key pair `gsg-keypair`. The response displays the MD5 public key fingerprint as specified in section 4 of RFC4716.

```
PROMPT>  ec2-import-keypair gsg-keypair --public-key-file C:\keys\mykey.ppk
KEYPAIR  gsg-keypair
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
```

API

To import the public key

1. Generate the key pair with the third-party tool of your choice.
2. Use `ImportKeyPair` to import the public key file to AWS. The following Query example names the key pair `gsg-keypair`. You must base64 encode the public key material before sending it to AWS.

```
https://ec2.amazonaws.com/?Action=ImportKeyPair
&KeyName=gsg-keypair
&PublicKeyMaterial=LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tDQpNSUlDZHp
DQ0FlQ2dBd0lCQWdJR0FQalRyR3pQ
TUEwR0NTcUdTSWIzRFFFQkJRVUFNRk14Q3pBSkJnTlZCQVlUDQpBbFZUTVJNd0VRWURWUVFLRXdw
QmJXRjZiMjR1WTI5dE1Rd3dDZ1lEVlFRTEV3TkJWMU14SVRBZkJnTlZCQU1UDQpHRUZYVXlCTWFX
MXBkR1ZrTFVGemMzVnlZVzVqWlNCRFFUQWVGdzB3T1RBM016RXlNVFEzTXpWYUZ3MHhNREEzDQpN
ekV5TVRRM016VmFNRkl4Q3pBSkJnTlZCQVlUQWxWVE1STXdFUVlEVlFRS0V3cEJiV0Y2YjI0dVky
OXRNUmN3DQpGUVlEVlFRTEV3NUJWMU10UkdWMlpXeHZjR1Z5Y3pFVk1CTUdBMVVFQXhNTWJUSnVi
RGhxZW00MWVHUjFNSUdmDQpNQTBHQ1NxR1NJYjNEUUVCQVFVQUE0R05BRENCaVFLQmdRQ1dOazBo
QytrcExBRnp2YkFQc3U1TDU5bFMwUnI0DQprZEpaM0RFak1pL0IwV2ZDSzhpS2hWYWt1WitHSnJt
NDdMUHZCaFVKWk9IeHVUU0VXakFDNmlybDJzKzlSWXVjDQpFZXg0TjI4ZlpCZGpORlAzdEgwZ2Nu
WjdIbXZ4aFBrTEtoRTdpZmViNmNGWUhRdHpHRnRPQ0ZQTmdUSE92VDE5DQoyR3lZb1VyU3BDVGFC
UUlEQVFBQm8xY3dWVEFPQmdOVkhROEJBZjhFQkFNQ0JhQXdGZ1lEVlIwbEFRSC9CQXd3DQpDZ1lJ
S3dZQkJRVUhBd0l3REFZRFZSMFRBUUgvQkFJd0FEQWRCZ05WSFE0RUZnUVU1RVNuTUZZZUzdyTDNX
TUdLDQpqejMxVXZ5TThnMHdEUVlKS29aSWh2Y05BUUVGQlFBRGdZRUFnWjdDZ1lJWHR1WFM1NHVq
bU5jOTR0NWRNc3krDQpCM0Z3WVVNdUd4WUI2eGQvSUVWMTFLRVEyZ0hpZUdMU21jUWg4c2JXTTdt
KzcrYm9UNmc2U2hLbU1jblkzWkRTDQpWRVFZZ25qcEt1aEZRd2pmaVpTUEc1UG5SVENhdkVqS3lT
TUpDVGxpdTdTTjMrR2J3cFU5Uzg3K21GM2tsMGRmDQpZNlIrbEl5SWcrU3ROOTg9DQotLS0tLUVO
RCBDRVJUSUZJQ0FURS0tLS0tEXAMPLE
&AuthParams
```

The response includes the MD5 public key fingerprint as specified in section 4 of RFC4716.

```
<ImportKeyPairResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>7a62c49f-347e-4fc4-9331-6e8eEXAMPLE</requestId>
   <keyName>gsg-keypair</keyName>
   <keyFingerprint>
     1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
   </keyFingerprint>
</ImportKeyPairResponse>
```

How to Have AWS Create the Key Pair for You

AWS Management Console

To generate a key pair

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Key Pairs** in the **Navigation** pane.

   The console displays a list of key pairs associated with your account.
3. Click **Create Key Pair**.

   The **Key Pair** dialog box appears.
4. Enter a name for the new key pair in the **Key Pair Name** field and click **Create**.

   You are prompted to download the key file.
5. Download the key file and keep it in a safe place. You will need it to access any instances that you launch with this key pair.

Command Line Tools

To generate a key pair

1. Use `ec2-add-keypair`. The following example names the resulting key pair `gsg-keypair`.

```
PROMPT>  ec2-add-keypair gsg-keypair
```

Amazon EC2 returns a private key, similar to the one in the following example.

```
KEYPAIR gsg-keypair
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
-----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQBuLFg5ujHrtm1jnutSuoO8Xe56LlT+HM8v/xkaa39EstM3/aFxTHgElQiJLChp
HungXQ29VTc8rc1bW0lkdi23OH5eqkMHGhvEwqa0HWASUMll4o3o/IX+0f2UcPoKCOVUR+jx71Sg
5AU52EQfanIn3ZQ8lFW7Edp5a3q4DhjGlUKToHVbicL5E+g45zfB95wIyywWZfeW/UUF3LpGZyq/
ebIUlq1qTbHkLbCC2r7RTn8vpQWp47BGVYGtGSBMpTRP5hnbzzuqj3itkiLHjU39S2sJCJ0TrJx5
i8BygR4s3mHKBj8l+ePQxG1kGbF6R4yg6sECmXn17MRQVXODNHZbAgMBAAECggEAY1tsiUsIwDl5
91CXirkYGuVfLyLflXenxfI50mDFms/mumTqloHO7tr0oriHDR5K7wMcY/YY5YkcXNo7mvUVD1pM
ZNUJs7rw9gZRTrf7LylaJ58kOcyajw8TsC4e4LPbFaHwS1d6K8rXh64o6WgW4SrsB6ICmr1kGQI7
3wcfgt5ecIu4TZf0OE9IHjn+2eRlsrjBdeORi7KiUNC/pAG23I6MdDOFEQRcCSigCj+4/mciFUSA
SWS4dMbrpb9FNSIcf9dcLxVM7/6KxgJNfZc9XWzUw77Jg8x92Zd0fVhHOux5IZC+UvSKWB4dyfcI
tE8C3p9bbU9VGyY5vLCAiIb4qQKBgQDLiO24GXrIkswF32YtBBMuVgLGCwU9h9HlO9mKAc2m8Cm1
jUE5IpzRjTedc9I2qiIMUTwtgnw42auSCzbUeYMURPtDqyQ7p6AjMujp9EPemcSVOK9vXYL0Ptco
xW9MC0dtV6iPkCN7gOqiZXPRKaFbWADp16p8UAIvS/a5XXk5jwKBgQCKkpHi2EISh1uRkhxljyWC
iDCiK6JBRsMvpLbc0v5dKwP5alo1fmdR5PJaV2qvZSj5CYNpMAy1/EDNTY5OSIJU+0KFmQbyhsbm
rdLNLDL4+TcnT7c62/aH01ohYaf/VCbRhtLlBfqGoQc7+sAc8vmKkesnF7CqCEKDyF/dhrxYdQKB
gC0iZzzNAapayz1+JcVTwwEid6j9JqNXbBc+Z2YwMi+T0Fv/P/hwkX/ypeOXnIUcw0Ih/YtGBVAC
DQbsz7LcY1HqXiHKYNWNvXgwwO+oiChjxvEkSdsTTIfnK4VSCvU9BxDbQHjdiNDJbL6oar92UN7V
rBYvChJZF7LvUH4YmVpHAoGAbZ2X7XvoeEO+uZ58/BGKOIGHByHBDiXtzMhdJr15HTYjxK7OgTZm
gK+8zp4L9IbvLGDMJO8vft32XPEWuvI8twCzFH+CsWLQADZMZKSsBasOZ/h1FwhdMgCMcY+Qlzd4
JZKjTSu3i7vhvx6RzdSedXEMNTZWN4qlIx3kR5aHcukCgYA9T+Zrvm1F0seQPbLknn7EqhXIjBaT
P8TTvW/6bdPi23ExzxZn7KOdrfclYRph1LHMpAONv/x2xALIf91UB+v5ohy1oDoasL0gij1houRe
2ERKKdwz0ZL9SWq6VTdhr/5G994CK72fy5WhyERbDjUIdHaK3M849JJuf8cSrvSb4g==
-----END RSA PRIVATE KEY-----
```

You must save the private key to a local file so that you can use it later.

2. Create a file named `id_rsa-gsg-keypair` and paste the entire key generated in step 1, including the following lines.

```
"-----BEGIN RSA PRIVATE KEY-----"
"-----END RSA PRIVATE KEY-----"
```

3. Confirm that the file contents looks similar to the following and save the file.

   You can save the file in any directory, but if you do not put it in your current directory, you should specify the full path when using commands that require the key pair.

```
 -----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQBuLFg5ujHrtm1jnutSuoO8Xe56LlT+HM8v/xkaa39EstM3/aFxTHgElQiJLChp
HungXQ29VTc8rc1bW0lkdi23OH5eqkMHGhvEwqa0HWASUMll4o3o/IX+0f2UcPoKCOVUR+jx71Sg
5AU52EQfanIn3ZQ8lFW7Edp5a3q4DhjGlUKToHVbicL5E+g45zfB95wIyywWZfeW/UUF3LpGZyq/
ebIUlq1qTbHkLbCC2r7RTn8vpQWp47BGVYGtGSBMpTRP5hnbzzuqj3itkiLHjU39S2sJCJ0TrJx5
i8BygR4s3mHKBj8l+ePQxG1kGbF6R4yg6sECmXn17MRQVXODNHZbAgMBAAECggEAY1tsiUsIwDl5
91CXirkYGuVfLyLflXenxfI50mDFms/mumTqloHO7tr0oriHDR5K7wMcY/YY5YkcXNo7mvUVD1pM
ZNUJs7rw9gZRTrf7LylaJ58kOcyajw8TsC4e4LPbFaHwS1d6K8rXh64o6WgW4SrsB6ICmr1kGQI7
3wcfgt5ecIu4TZf0OE9IHjn+2eRlsrjBdeORi7KiUNC/pAG23I6MdDOFEQRcCSigCj+4/mciFUSA
SWS4dMbrpb9FNSIcf9dcLxVM7/6KxgJNfZc9XWzUw77Jg8x92Zd0fVhHOux5IZC+UvSKWB4dyfcI
tE8C3p9bbU9VGyY5vLCAiIb4qQKBgQDLiO24GXrIkswF32YtBBMuVgLGCwU9h9HlO9mKAc2m8Cm1
jUE5IpzRjTedc9I2qiIMUTwtgnw42auSCzbUeYMURPtDqyQ7p6AjMujp9EPemcSVOK9vXYL0Ptco
xW9MC0dtV6iPkCN7gOqiZXPRKaFbWADp16p8UAIvS/a5XXk5jwKBgQCKkpHi2EISh1uRkhxljyWC
iDCiK6JBRsMvpLbc0v5dKwP5alo1fmdR5PJaV2qvZSj5CYNpMAy1/EDNTY5OSIJU+0KFmQbyhsbm
rdLNLDL4+TcnT7c62/aH01ohYaf/VCbRhtLlBfqGoQc7+sAc8vmKkesnF7CqCEKDyF/dhrxYdQKB
gC0iZzzNAapayz1+JcVTwwEid6j9JqNXbBc+Z2YwMi+T0Fv/P/hwkX/ypeOXnIUcw0Ih/YtGBVAC
DQbsz7LcY1HqXiHKYNWNvXgwwO+oiChjxvEkSdsTTIfnK4VSCvU9BxDbQHjdiNDJbL6oar92UN7V
rBYvChJZF7LvUH4YmVpHAoGAbZ2X7XvoeEO+uZ58/BGKOIGHByHBDiXtzMhdJr15HTYjxK7OgTZm
gK+8zp4L9IbvLGDMJO8vft32XPEWuvI8twCzFH+CsWLQADZMZKSsBasOZ/h1FwhdMgCMcY+Qlzd4
JZKjTSu3i7vhvx6RzdSedXEMNTZWN4qlIx3kR5aHcukCgYA9T+Zrvm1F0seQPbLknn7EqhXIjBaT
P8TTvW/6bdPi23ExzxZn7KOdrfclYRph1LHMpAONv/x2xALIf91UB+v5ohy1oDoasL0gij1houRe
2ERKKdwz0ZL9SWq6VTdhr/5G994CK72fy5WhyERbDjUIdHaK3M849JJuf8cSrvSb4g==
-----END RSA PRIVATE KEY-----
```

4. If you're using OpenSSH (or another SSH client), you should set the permissions of this file so it is only readable by you.

   On Linux and UNIX, enter the information in the following example.

```
$  chmod 400 id_rsa-gsg-keypair ; ls -l id_rsa-gsg-keypair
```

   You receive output similar to the following example.

```
-r--------  1 fred flintstones 1701 Jun 19 17:57 id_rsa-gsg-keypair
```

API

To generate a key pair

1. Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateKeyPair
&KeyName=gsg-keypair
&...auth parameters...
```

Following is an example response.

```
<CreateKeyPairResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <keyName>gsg-keypair</keyName>
  <keyFingerprint>
     1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
  </keyFingerprint>
  <keyMaterial>-----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQBuLFg5ujHrtm1jnutSuoO8Xe56LlT+HM8v/xkaa39EstM3/aFxTHgElQiJLChp
HungXQ29VTc8rc1bW0lkdi23OH5eqkMHGhvEwqa0HWASUMll4o3o/IX+0f2UcPoKCOVUR+jx71Sg
5AU52EQfanIn3ZQ8lFW7Edp5a3q4DhjGlUKToHVbicL5E+g45zfB95wIyywWZfeW/UUF3LpGZyq/
ebIUlq1qTbHkLbCC2r7RTn8vpQWp47BGVYGtGSBMpTRP5hnbzzuqj3itkiLHjU39S2sJCJ0TrJx5
i8BygR4s3mHKBj8l+ePQxG1kGbF6R4yg6sECmXn17MRQVXODNHZbAgMBAAECggEAY1tsiUsIwDl5
91CXirkYGuVfLyLflXenxfI50mDFms/mumTqloHO7tr0oriHDR5K7wMcY/YY5YkcXNo7mvUVD1pM
ZNUJs7rw9gZRTrf7LylaJ58kOcyajw8TsC4e4LPbFaHwS1d6K8rXh64o6WgW4SrsB6ICmr1kGQI7
3wcfgt5ecIu4TZf0OE9IHjn+2eRlsrjBdeORi7KiUNC/pAG23I6MdDOFEQRcCSigCj+4/mciFUSA
SWS4dMbrpb9FNSIcf9dcLxVM7/6KxgJNfZc9XWzUw77Jg8x92Zd0fVhHOux5IZC+UvSKWB4dyfcI
tE8C3p9bbU9VGyY5vLCAiIb4qQKBgQDLiO24GXrIkswF32YtBBMuVgLGCwU9h9HlO9mKAc2m8Cm1
jUE5IpzRjTedc9I2qiIMUTwtgnw42auSCzbUeYMURPtDqyQ7p6AjMujp9EPemcSVOK9vXYL0Ptco
xW9MC0dtV6iPkCN7gOqiZXPRKaFbWADp16p8UAIvS/a5XXk5jwKBgQCKkpHi2EISh1uRkhxljyWC
iDCiK6JBRsMvpLbc0v5dKwP5alo1fmdR5PJaV2qvZSj5CYNpMAy1/EDNTY5OSIJU+0KFmQbyhsbm
rdLNLDL4+TcnT7c62/aH01ohYaf/VCbRhtLlBfqGoQc7+sAc8vmKkesnF7CqCEKDyF/dhrxYdQKB
gC0iZzzNAapayz1+JcVTwwEid6j9JqNXbBc+Z2YwMi+T0Fv/P/hwkX/ypeOXnIUcw0Ih/YtGBVAC
DQbsz7LcY1HqXiHKYNWNvXgwwO+oiChjxvEkSdsTTIfnK4VSCvU9BxDbQHjdiNDJbL6oar92UN7V
rBYvChJZF7LvUH4YmVpHAoGAbZ2X7XvoeEO+uZ58/BGKOIGHByHBDiXtzMhdJr15HTYjxK7OgTZm
gK+8zp4L9IbvLGDMJO8vft32XPEWuvI8twCzFH+CsWLQADZMZKSsBasOZ/h1FwhdMgCMcY+Qlzd4
JZKjTSu3i7vhvx6RzdSedXEMNTZWN4qlIx3kR5aHcukCgYA9T+Zrvm1F0seQPbLknn7EqhXIjBaT
P8TTvW/6bdPi23ExzxZn7KOdrfclYRph1LHMpAONv/x2xALIf91UB+v5ohy1oDoasL0gij1houRe
2ERKKdwz0ZL9SWq6VTdhr/5G994CK72fy5WhyERbDjUIdHaK3M849JJuf8cSrvSb4g==
-----END RSA PRIVATE KEY-----</keyMaterial>
</CreateKeyPairResponse>
```

You must save the private key to a local file so that you can use it later.

2. Create a file named `id_rsa-gsg-keypair` and paste the entire key generated in step 1, including the following lines.

```
"-----BEGIN RSA PRIVATE KEY-----"
"-----END RSA PRIVATE KEY-----"
```

3. Confirm that the file contents looks similar to the following and save the file.

   You can save the file in any directory, but if you do not put it in your current directory, you should specify the full path when using commands that require the key pair.

```
 -----BEGIN RSA PRIVATE KEY-----
MIIEoQIBAAKCAQBuLFg5ujHrtm1jnutSuoO8Xe56LlT+HM8v/xkaa39EstM3/aFxTHgElQiJLChp
HungXQ29VTc8rc1bW0lkdi23OH5eqkMHGhvEwqa0HWASUMll4o3o/IX+0f2UcPoKCOVUR+jx71Sg
5AU52EQfanIn3ZQ8lFW7Edp5a3q4DhjGlUKToHVbicL5E+g45zfB95wIyywWZfeW/UUF3LpGZyq/
ebIUlq1qTbHkLbCC2r7RTn8vpQWp47BGVYGtGSBMpTRP5hnbzzuqj3itkiLHjU39S2sJCJ0TrJx5
```

```
i8BygR4s3mHKBj8l+ePQxG1kGbF6R4yg6sECmXn17MRQVXODNHZbAgMBAAECggEAY1tsiUsIwDl5
91CXirkYGuVfLyLflXenxfI50mDFms/mumTqloHO7tr0oriHDR5K7wMcY/YY5YkcXNo7mvUVD1pM
ZNUJs7rw9gZRTrf7LylaJ58kOcyajw8TsC4e4LPbFaHwS1d6K8rXh64o6WgW4SrsB6ICmr1kGQI7
3wcfgt5ecIu4TZf0OE9IHjn+2eRlsrjBdeORi7KiUNC/pAG23I6MdDOFEQRcCSigCj+4/mciFUSA
SWS4dMbrpb9FNSIcf9dcLxVM7/6KxgJNfZc9XWzUw77Jg8x92Zd0fVhHOux5IZC+UvSKWB4dyfcI
tE8C3p9bbU9VGyY5vLCAiIb4qQKBgQDLiO24GXrIkswF32YtBBMuVgLGCwU9h9HlO9mKAc2m8Cm1
jUE5IpzRjTedc9I2qiIMUTwtgnw42auSCzbUeYMURPtDqyQ7p6AjMujp9EPemcSVOK9vXYL0Ptco
xW9MC0dtV6iPkCN7gOqiZXPRKaFbWADp16p8UAIvS/a5XXk5jwKBgQCKkpHi2EISh1uRkhxljyWC
iDCiK6JBRsMvpLbc0v5dKwP5alo1fmdR5PJaV2qvZSj5CYNpMAy1/EDNTY5OSIJU+0KFmQbyhsbm
rdLNLDL4+TcnT7c62/aH01ohYaf/VCbRhtLlBfqGoQc7+sAc8vmKkesnF7CqCEKDyF/dhrxYdQKB
gC0iZzzNAapayz1+JcVTwwEid6j9JqNXbBc+Z2YwMi+T0Fv/P/hwkX/ypeOXnIUcw0Ih/YtGBVAC
DQbsz7LcY1HqXiHKYNWNvXgwwO+oiChjxvEkSdsTTIfnK4VSCvU9BxDbQHjdiNDJbL6oar92UN7V
rBYvChJZF7LvUH4YmVpHAoGAbZ2X7XvoeEO+uZ58/BGKOIGHByHBDiXtzMhdJr15HTYjxK7OgTZm
gK+8zp4L9IbvLGDMJO8vft32XPEWuvI8twCzFH+CsWLQADZMZKSsBasOZ/h1FwhdMgCMcY+Qlzd4
JZKjTSu3i7vhvx6RzdSedXEMNTZWN4qlIx3kR5aHcukCgYA9T+Zrvm1F0seQPbLknn7EqhXIjBaT
P8TTvW/6bdPi23ExzxZn7KOdrfclYRph1LHMpAONv/x2xALIf91UB+v5ohy1oDoasL0gij1houRe
2ERKKdwz0ZL9SWq6VTdhr/5G994CK72fy5WhyERbDjUIdHaK3M849JJuf8cSrvSb4g==
-----END RSA PRIVATE KEY-----
```

4. If you're using OpenSSH (or another SSH client), you should set the permissions of this file so it is readable only by you.

   On Linux and UNIX, enter the information in the following example.

   ```
   $  chmod 400 id_rsa-gsg-keypair ; ls -l id_rsa-gsg-keypair
   ```

   You receive output similar to the following example.

   ```
   -r--------  1 fred flintstones 1701 Jun 19 17:57 id_rsa-gsg-keypair
   ```

Adding Rules to the Default Security Group

Before you can log in to an instance, you must authorize access.

This section describes how to add rules that allow HTTP access on port 80, SSH access on port 22, and Remote Desktop (RDP) access on port 3389. This enables the instance to be reached on port 80 from the Internet and enables you to administer the instance over SSH or RDP.

AWS Management Console

To authorize access to your instance

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. Select the **default** security group.
   Its rules appear on the **Inbound** tab in the lower pane.
4. To add the HTTP rule:

   a. Select `HTTP` from the **Create a new rule** menu.
   b. Click **Add Rule**.

The rule is added to the list of rules on the right. However, the rule isn't applied to the group until you click **Apply Rule Changes** (which you'll do after you've added all the rules). Notice that the rules are highlighted in blue, and there's an asterisk on the **Inbound** tab. These signs indicate that you haven't yet applied the rule changes.

5. To add the SSH rule:

   a. Select `SSH` from the **Create a new rule** menu.
   b. In the **Source** field, enter your public IP address (e.g., 192.0.2.1/32).
   c. Click **Add Rule**.
      The rule is added to the list of rules.

6. To add the RDP rule:

   a. Select `RDP` from the **Create a new rule** menu.
   b. In the **Source** field, enter your public IP address (e.g., 192.0.2.1/32).
   c. Click **Add Rule**.
      The rule is added to the list of rules.

7. Click **Apply Rule Changes**.

   The new rules now apply to the default security group. Notice that the rules are no longer highlighted in blue, and the asterisk no longer appears on the **Inbound** tab.

Command Line Tools

To authorize access to your instance

- Enter the `ec2-authorize` commands.

```
PROMPT>  ec2-authorize default -p 22 -s your-local-system's-public-ip-ad
dress/32
PERMISSION     default  ALLOWS  tcp     22      22      FROM    CIDR   your-
local-system's-public-ip-address/32
PROMPT>  ec2-authorize default -p 3389 -s your-local-system's-public-ip-ad
dress/32
PERMISSION     default  ALLOWS  tcp     3389      3389      FROM    CIDR
your-local-system's-public-ip-address/32
PROMPT>  ec2-authorize default -p 80
PERMISSION     default  ALLOWS  tcp     80      80      FROM    CIDR
0.0.0.0/0
```

Because we didn't specify otherwise, your instance was launched in your `default` group. The first command authorizes network access from your local system to instances in your default group on the standard SSH port (22). The second command authorizes RDP access (port 3389) from your local system to instances in the default security group. Similarly, the third command opens up the standard HTTP port (80).

API

To authorize access to your instance

- Construct the following Query requests. The first two of the following requests give your local system the ability to use SSH (port 22) or Remote Desktop (port 3389) to connect to any instance in the "default" security group. The third command allows all port 80 traffic into all instances in the "default" security group.

```
https://ec2.amazonaws.com/
?Action=AuthorizeSecurityGroupIngress
&GroupName=default
&IpPermissions.1.IpProtocol=tcp
&IpPermissions.1.FromPort=22
&IpPermissions.1.ToPort=22
&IpPermissions.1.IpRanges.1.CidrIp=your-local-system's-public-ip-address/32
&AUTHPARAMS

https://ec2.amazonaws.com/
?Action=AuthorizeSecurityGroupIngress
&GroupName=default
&IpPermissions.1.IpProtocol=tcp
&IpPermissions.1.FromPort=3389
&IpPermissions.1.ToPort=3389
&IpPermissions.1.IpRanges.1.CidrIp=your-local-system's-public-ip-address/32
&AUTHPARAMS

https://ec2.amazonaws.com/
?Action=AuthorizeSecurityGroupIngress
&GroupName=default
&IpPermissions.1.IpProtocol=tcp
&IpPermissions.1.FromPort=80
&IpPermissions.1.ToPort=80
&IpPermissions.1.IpRanges.1.CidrIp=0.0.0.0/0
&AUTHPARAMS
```

Following is an example response.

```
<AuthorizeSecurityGroupIngressResponse xmlns="http://ec2.amazon
aws.com/doc/2011-12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <return>true</return>
</AuthorizeSecurityGroupIngressResponse>
```

Running an Instance

This section describes how to run an instance.

AWS Management Console

To launch an instance

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Instances** in the **Navigation** pane.

The console displays a list of running instances.

3. Click **Launch Instance**.
   The Launch Instance wizard opens with the **Quick Start** tab displayed.
4. If you are launching a Linux/UNIX instance, locate the **Basic 32-bit Amazon Linux AMI** and click its **Select** button. If you are launching a Windows instance, locate a Windows AMI and click its **Select** button.

   Note

   We recommend launching basic AMIs for this tutorial, but you can launch any AMI.

5. Confirm the following settings and click **Continue**.

   - The **Number of Instances** is set to 1.
   - The **Availability Zone** is set to `No Preference`.
   - The **Instance Type** is set to `Small (m1.small)`.
   - The button for **Launch Instances** is selected.

   The second **Instance Details** pages is displayed with advanced instance options.
6. Click **Continue** to accept the default settings for the advanced options.
   The third **Instance Details** page displays. Here you can specify tags for the instance (for more information, see Using Tags (p. 275)).
7. Click **Continue** to accept the default tag settings.
   The **Create Key Pair** page is displayed.
8. Select **Choose from your existing Key Pairs**, and then select a key pair you've created (see Getting an SSH Key Pair (p. 77)).

   The **Configure Firewall** page is displayed.
9. Select **Choose one or more of your existing Security Groups**, and select `default`, because you've already configured the `default` security group (see Adding Rules to the Default Security Group (p. 83)).

   Note

   After an instance is running, you can't change which security groups it belongs to.

10. Click **Continue**, and then **Launch** to begin launching your instance.

Command Line Tools

To launch an instance

1. Use the `ec2-run-instances` command.

```
PROMPT>  ec2-run-instances ami-3ac33653 -k gsg-keypair
```

Amazon EC2 returns output similar to the following example.

```
RESERVATION     r-f25e6f9a      999988887777    default
INSTANCE       i-85b435ee      ami-3ac33653          pending     gsg-keypair
     0 m1.small 2010-03-30T08:01:36+0000        us-east-1a      aki-407d9529
          monitoring-disabled           ebs
```

2. Look for the instance ID in the second field and write it down.
   You use it to manipulate this instance (including terminating it when you are finished).

   It takes a few minutes for the instance to launch.

3. The following command displays the launch status of the instance.

```
PROMPT>  ec2-describe-instances i-85b435ee
RESERVATION r-f25e6f9a 999988887777   default
INSTANCE i-85b435ee ami-3ac33653 ec2-67-202-28-13.compute-1.amazonaws.com
domU-12-31-39-00-78-93.compute-1.internal running gsg-keypair 0 m1.small 2010-
03-30T08:01:36+0000 us-east-1a aki-407d9529 monitoring-disabled 67.202.28.13
 10.254.127.97 ebs
BLOCKDEVICE /dev/sda1 vol-02a2a46b 2010-03-30T08:01:44.000Z
```

⚠ **Important**

After launching an instance, you are billed hourly for running time. When you are finished, make sure to terminate any instances that you started.

When the instance state in the field just before the key pair name reads "running", the instance has started booting. There might be a short time before it is accessible over the network, however. The first DNS name is your instance's external DNS name, i.e. the one that can be used to contact it from the Internet. The second DNS name is your instance's local DNS name, and is only contactable by other instances within the Amazon EC2 network. The DNS names of your instances are different than those shown in the preceding example and you should use yours instead. The examples in this guide use the public DNS name.

If the instance's state immediately goes to "terminated" instead of "running", you can get information about why the instance didn't launch. For more information, see What to Do If an Instance Immediately Terminates (p. 107).

API

To launch an instance

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=RunInstances
&ImageId=ami-3ac33653
&MaxCount=1
&MinCount=1
&KeyName=gsg-keypair
&Placement.AvailabilityZone=us-east-1a
&...auth parameters...
```

  Following is an example response.

```
<RunInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <reservationId>r-47a5402e</reservationId>
  <ownerId>999988887777</ownerId>
  <groupSet>
    <item>
      <groupId>default</groupId>
    </item>
  </groupSet>
  <instancesSet>
    <item>
      <instanceId>i-2ba64342</instanceId>
      <imageId>ami-3ac33653</imageId>
      <instanceState>
        <code>0</code>
        <name>pending</name>
      </instanceState>
      <privateDnsName></privateDnsName>
      <dnsName></dnsName>
      <keyName>gsg-keypair</keyName>
      <amiLaunchIndex>0</amiLaunchIndex>
      <instanceType>m1.small</instanceType>
      <launchTime>2007-08-07T11:51:50.000Z</launchTime>
      <placement>
        <availabilityZone>us-east-1a</availabilityZone>
      </placement>
      <monitoring>
        <enabled>false</enabled>
      </monitoring>
    </item>
  </instancesSet>
</RunInstancesResponse>
```

To view the status of a launched instance

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeInstances
&InstanceId=i-2ba64342
&...auth parameters...
```

Following is an example response

```
<DescribeInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

  <reservationId>r-47a5402e</reservationId>
  <ownerId>999988887777</ownerId>
  <groupSet>
    <item>
      <groupId>default</groupId>
    </item>
  </groupSet>
  <instancesSet>
    <item>
```

```
      <instanceId>i-2ba64342</instanceId>
      <imageId>ami-3ac33653</imageId>
      <instanceState>
        <code>16</code>
        <name>running</name>
      </instanceState>
      <privateDnsName></privateDnsName>
      <dnsName></dnsName>
      <keyName>gsg-keypair</keyName>
      <amiLaunchIndex>0</amiLaunchIndex>
      <instanceType>m1.small</instanceType>
      <launchTime>2007-08-07T11:51:50.000Z</launchTime>
      <placement>
        <availabilityZone>us-east-1a</availabilityZone>
      </placement>
      <monitoring>
        <enabled>false</enabled>
      </monitoring>
    </item>
  </instancesSet>
</DescribeInstancesResponse>
```

If the instance's state immediately goes to "terminated" instead of "running", you can get information about why the instance didn't launch. For more information, see What to Do If an Instance Immediately Terminates (p. 107).

Stopping and Starting Instances

This section describes how to stop and start instances that use Amazon EBS volumes as their root devices.

When an instance is stopped, it is shut down, and any data stored in RAM is not preserved. You're not billed for hourly usage or data transfer, but you're billed for any Amazon EBS volume storage. You can start the instance at any time with a start request. Each time you transition an instance from stopped to started, we charge a full instance hour, even if transitions happen multiple times within a single hour.

Note

If you try to stop an instance that uses an instance store as its root device, you receive an error.

When you start a stopped instance, the IP address is likely to change. By default, the Ec2ConfigService service (Windows) changes the instance hostname to match the new IP and initiates a reboot.

Important

If an instance reboots (intentionally or unintentionally), the data on the instance store will survive.

However, under the following circumstances the data in the instance store will be lost:

- Failure of an underlying drive.
- Running an instance on degraded hardware.
- Stopping an Amazon EBS-backed instance.
- Terminating an instance.

AWS Management Console

To stop and start an instance

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Instances** in the **Navigation** pane.
   The console displays a list of running instances.
3. Select an instance, select **Instance Actions**, and click **Stop Instance**.
   A confirmation dialog box appears.
4. Click the **Yes, Stop Instance**.
   The instance is stopped and saved as a snapshot.
5. To restart the stopped instance, select **Stopped Instances** from the **Viewing** list box.
   The newly stopped instance appears in the list.
6. Select the instance, select **Instance Actions**, and click **Start Instance**.
   The instance begins restarting.

Command Line Tools

To stop and start an instance

1. Use the `ec2-stop-instances` command.

```
PROMPT>  ec2-stop-instances i-10a64379
```

Amazon EC2 returns output similar to the following example.

```
IMAGE i-10a64379 running stopping
```

2. Use the `ec2-start-instances` command.

```
PROMPT>  ec2-start-instances i-10a64379
```

Amazon EC2 returns output similar to the following example.

```
IMAGE i-10a64379 stopped pending
```

API

To stop an instance

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=StopInstances
&InstanceId.1=i-10a64379
&...auth parameters...
```

Following is an example response.

```
<StopInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <instancesSet>
     <item>
        <instanceId>exampleinstanceid</instanceId>
        <currentState>
           <code>64</code>
           <name>stopping</name>
        </currentState>
        <previousState>
           <code>16</code>
           <name>running</name>
        </previousState>
  </instancesSet>
</StopInstancesResponse>
```

To start an instance

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=StartInstances
&InstanceId.1=i-10a64379
&...auth parameters...
```

Following is an example response.

```
<StartInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <instancesSet>
      <item>
         <instanceId>exampleinstanceid</instanceId>
         <currentState>
            <code>0</code>
            <name>pending</name>
         </currentState>
         <previousState>
            <code>80</code>
            <name>stopped</name>
         </previousState>
      </item>
   </instancesSet>
</StartInstancesResponse>
```

Ensuring Idempotency

When you launch an instance through the API or the command line tools (i.e., the API tools), you can optionally provide a client token to ensure the request is idempotent. If timeouts or connection errors occur, you can repeat the request and be sure you haven't launched more instances than you intended. The token must be a unique, case-sensitive string of up to 64 ASCII characters.

If you repeat the request with the same client token, the same response is returned for each repeated request. The only information that might vary in the response is the state of the instance (e.g., pending). The client token is included in the response when you describe the instance.

If you repeat the request with the same client token, but change another request parameter, Amazon EC2 returns an `IdempotentParameterMismatch` error.

The client token is valid for at least 24 hours after the termination of the instance. You should not reuse a client token for another call later on.

You can use the same client token for the same request across different Regions. For example, if you send an idempotent request to launch an instance in the us-east-1 Region, and you use the same exact call (with the same client token) in each of the other Regions, the result is a different running instance in each Region.

The following table shows common response codes and the recommended course of action.

Code	Retry	Comments
200 (OK)	No effect	After you receive a 200, the request has succeeded and any further retries have no effect.
400 (Client Error)	Not recommended	A 400 response typically indicates the request will never succeed (for example when a specified parameter value is not valid). In certain specific cases relating to resources that are transitioning between states, a repeat of the request could possibly succeed (e.g., launching an instance against an Amazon EBS volume that is currently transitioning to the available state).
500 (Server Internal Error)	Recommended	These errors are generally transient. Repeat the request with an appropriate back-off strategy.
503 (Server Unavailable)	Recommended	These errors can occur in times of extreme load. Repeat the request with an appropriate back-off strategy.

Command Line Tools

To send an idempotent request to launch instances

- Add the `--client-token` option to your request with a unique, case-sensitive string of up to 64 ASCII characters.

```
PROMPT>  ec2-run-instances ami-b232d0db -k gsg-keypair --client-token
550e8400-e29b-41d4-a716-446655440000
```

API

To send an idempotent request to launch instances

- Add the *`ClientToken`* parameter to your Query request with a unique, case-sensitive string of up to 64 ASCII characters.

```
https://ec2.amazonaws.com/
?Action=RunInstances
&ImageId=ami-3ac33653
...
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&...auth parameters...
```

Enabling Termination Protection for an Instance

Note

The following information applies to instances of Amazon EBS-backed AMIs or Amazon EC2 instance store-backed AMIs.

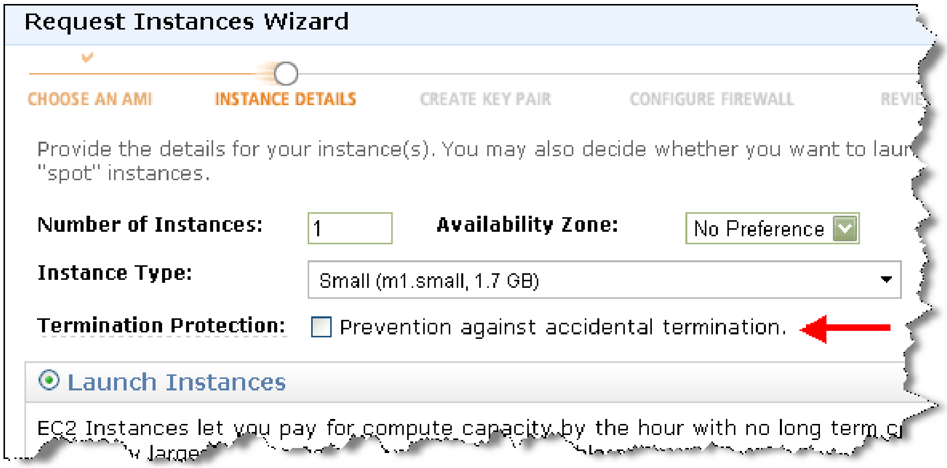
By default, you can terminate any instances you launch. If you want to prevent accidental termination of the instance, you can enable *termination protection* for the instance. The instance's `DisableApiTermination` attribute controls whether the instance is protected. A value of `true` means the instance is protected and can't be terminated; a value of `false` means it's not protected and can be terminated. You can specify whether an instance is protected either at launch time or after the instance is running. You can modify this attribute while the instance is running or stopped (for Amazon EBS-backed instances).

AWS Management Console

By default, termination protection is disabled for an instance at launch time.

To enable termination protection for an instance at launch time

- In the **Request Instances Wizard**, select the check box for **Termination Protection**.



Provide the details for your instance(s). You may also decide whether you want to laun
"spot" instances.

Number of Instances: 1
Availability Zone: No Preference

Instance Type: Small (m1.small, 1.7 GB)

Termination Protection: □ Prevention against accidental termination.

○ Launch Instances

EC2 Instances let you pay for compute capacity by the hour with no long term c
w large

To enable termination protection for an instance after it's launched

1. In your list of instances, right-click the instance and select **Change Termination Protection**.

   A confirmation dialog box appears with the current status of termination protection.

2. Click **Yes, Enable**.

   Now the instance cannot be terminated until you disable termination protection.

You can use the same procedure to disable termination protection for the instance. You can enable and disable this attribute of an instance as often as you want.

Command Line Tools

By default, termination protection is disabled for an instance at launch time.

To enable termination protection for an instance at launch time

- Add `--disable-api-termination` to the `ec2-run-instances` command.

```
PROMPT>  ec2-run-instances ami_id --disable-api-termination ...
```

To enable termination protection for an instance after it's launched

1. View the current value for the attribute with the following command.

```
PROMPT>  ec2-describe-instance-attribute instance_id --disable-api-termination
```

   Following is a sample response.

```
disableApiTermination   i-87ad5eec      false
```

2. Enable termination protection with the following command.

```
PROMPT>  ec2-modify-instance-attribute instance_id --disable-api-termination
 true
```

   Following is a sample response.

```
disableApiTermination   i-87ad5eec      true
```

To disable termination protection for an instance

- Disable termination protection for the instance with the following command.

```
PROMPT>  ec2-modify-instance-attribute instance_id --disable-api-termination
 false
```

  Following is a sample response.

```
disableApiTermination   i-87ad5eec      false
```

You can enable and disable this attribute of an instance as often as you want.

API

By default, termination protection is disabled for an instance at launch time.

To enable termination protection for an instance at launch time

- Issue a Query request for `RunInstances` similar to the following example and include `DisableApiTermination=true`.

  ```
  https://ec2.amazonaws.com/?Action=RunInstances
  &ImageId=ami-60a54009
  &MaxCount=3
  &MinCount=1
  &DisableApiTermination=true
  &Placement.AvailabilityZone=us-east-1b
  &Monitoring.Enabled=true
  &AUTHPARAMS
  ```

  For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

To enable termination protection for an instance after it's launched

1. Issue the following Query request to get the current value of the `disableApiTermination` attribute.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstanceAttribute
   &InstanceId=i-87ad5eec
   &Attribute=disableApiTermination
   &AUTHPARAMS
   ```

   Following is an example response.

   ```
   <DescribeInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
   12-01/">
      <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
      <instanceId>i-87ad5eec</instanceId>
      <disableApiTermination>
         <value>false</value>
      </disableApiTermination>
   </DescribeInstanceAttributeResponse>
   ```

2. Issue the following Query request to modify the `disableApiTermination` attribute.

   ```
   https://ec2.amazonaws.com/
   ?Action=ModifyInstanceAttribute
   &InstanceId=i-87ad5eec
   &DisableApiTermination.Value=true
   &AUTHPARAMS
   ```

   Following is an example response.

```
<ModifyInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <return>true</return>
</ModifyInstanceAttributeResponse>
```

To disable termination protection for an instance

- Issue the following Query request to modify the `disableApiTermination` attribute.

```
https://ec2.amazonaws.com/
?Action=ModifyInstanceAttribute
&InstanceId=i-87ad5eec
&DisableApiTermination.Value=false
&AUTHPARAMS
```

Following is an example response.

```
<ModifyInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
   <return>true</return>
</ModifyInstanceAttributeResponse>
```

You can enable and disable this attribute of an instance as often as you want.

Using Instance Metadata

Topics

Amazon EC2 instances can access instance-specific metadata as well as data supplied when launching the instances. This data can be used to build more generic AMIs that can be modified by configuration files supplied at launch time.

For example, if you run web servers for various small businesses, they can all use the same AMI and retrieve their content from the Amazon S3 bucket you specify at launch. To add a new customer at any time, simply create a bucket for the customer, add their content, and launch your AMI.

Metadata is divided into categories. For a list of the categories, see Appendix B: Metadata Categories (p. 422).

Data Retrieval

An instance retrieves the data by querying a web server using a Query API. The base URI of all requests is `http://169.254.169.254/latest/`.

Security of Launch Data

Although only your specific instance can access launch data, the data is not protected by cryptographic methods. You should take suitable precautions to protect sensitive data (such as long lived encryption keys).

Note

You are not billed for HTTP requests used to retrieve metadata and user-supplied data.

Metadata Retrieval

Requests for a specific metadata resource returns the appropriate value or a `404` HTTP error code if the resource is not available. All metadata is returned as text (content type `text/plain`).

Requests for a general metadata resource (i.e. an URI ending with a `/`) return a list of available resources or a `404` HTTP error code if there is no such resource. The list items are on separate lines terminated by line feeds (ASCII 10).

Examples

The following examples list HTTP GET requests and responses. You can use a tool such as curl or wget to make these types of requests.

This example gets the available versions of the metadata. These versions do not necessarily correlate with an EC2 API version.

```
GET http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
...
```

This example gets the top-level metadata items. Some of these items are available only for instances in a VPC. For more information about each of these items, see Appendix B: Metadata Categories (p. 422).

```
GET http://169.254.169.254/latest/meta-data/
amiid
ami-launch-index
ami-manifest-path
block-device-mapping/
hostname
instance-action
instance-id
instance-type
kernel-id
local-hostname
local-ipv4
mac
network/
```

```
placement/
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
```

This example gets the value of some of the metadata items from the preceding example.

```
GET http://169.254.169.254/latest/meta-data/ami-manifest-path
my-amis/spamd-image.manifest.xml
GET http://169.254.169.254/latest/meta-data/ami-id
ami-2bb65342
GET http://169.254.169.254/latest/meta-data/reservation-id
r-fea54097
GET http://169.254.169.254/latest/meta-data/hostname
ec2-67-202-51-223.compute-1.amazonaws.com
```

This example gets the list of available public keys.

```
GET http://169.254.169.254/latest/meta-data/public-keys/
0=my-public-key
```

This example shows the formats in which public key 0 is available.

```
GET http://169.254.169.254/latest/meta-data/public-keys/0/
openssh-key
```

This example gets public key 0 (in the OpenSSH key format).

```
GET http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa AAAA.....wZEf my-public-key
```

This example gets the product code.

```
GET http://169.254.169.254/latest/meta-data/product-codes
774F4FF8
```

This example gets an instance's Media Access Control (MAC) address.

```
GET http://169.254.169.254/latest/meta-data/mac
02:29:96:8f:6a:2d
```

This example shows the network information available for a VPC instance.

```
GET http://169.254.169.254/latest/meta-data/network/inter
faces/macs/02:29:96:8f:6a:2d/
local-hostname
local-ipv4s
mac
public-ipv4s
```

```
security-group-ids
subnet-id
subnet-ipv4-cidr-block
vpc-id
vpc-ipv4-cidr-block
```

This example gets a VPC instance's subnet ID.

```
GET http://169.254.169.254/latest/meta-data/network/inter
faces/macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

This example shows the network information available for an EC2 instance (one not running in a VPC).

```
GET http://169.254.169.254/latest/meta-data/network/inter
faces/macs/03:15:28:7g:5b:8a/
local-hostname
local-ipv4s
mac
public-ipv4s
public-hostname
```

User Data Retrieval

When you launch an instance, you can specify *user data*, which is available for all instances in the reservation to retrieve. You can also add (or modify) user data to Amazon EBS-backed instances when they're stopped. Requests for the user data returns the data as-is (content type `application/x-octetstream`). Many people use user data to configure an instance during launch or even run a configuration script.

Note

All user-supplied data is treated as opaque data; what you give us is what you get back. It is the responsibility of the instance to interpret this data appropriately.

Example

This shows an example of returning comma-separated user-supplied data.

```
GET http://169.254.169.254/latest/user-data
1234,fred,reboot,true | 4512,jimbo, | 173,,,
```

This shows an example of returning line-separated user-supplied data.

```
GET http://169.254.169.254/latest/user-data
[general]
instances: 4

[instance-0]
s3-bucket: fred

[instance-1]
reboot-on-error: yes
```

You can modify the user data for an Amazon EBS-backed instance while the instance is stopped. For more information, see Modifying Attributes of a Stopped Instance (p. 190).

Use Case: AMI Launch Index Value

In this example, Alice wants to launch four instances of her favorite database AMI with the first acting as master and the remainder acting as replicas.

The master database configuration specifies various database parameters (e.g., the size of store) while the replicas' configuration specifies different parameters, such as the replication strategy. Alice decides to provide this data as an ASCII string with a pipe symbol (|) delimiting the data for the various instances:

```
store-size=123PB backup-every=5min | replicate-every=1min | replicate-every=2min
 | replicate-every=10min | replicate-every=20min
```

The `store-size=123PB backup-every=5min` defines the master database configuration, `replicate-every=1min` defines the first replicant's configuration, `replicate-every=2min` defines the second replicant's configuration, and so on.

Alice launches four instances.

```
PROMPT> ec2-run-instances ami-2bb65342 -n 4 -d "store-size=123PB backup-
every=5min | replicate-every=1min | replicate-every=2min | replicate-every=10min
 | replicate-every=20min"

RESERVATION     r-fea54097      598916040194    default
INSTANCE i-3ea74257 ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-31a74258 ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-31a74259 ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-31a7425a ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
```

After they're launched, all instances have a copy of the user data and the common metadata shown here:

- AMI id: ami-2bb65342
- Reservation ID: r-fea54097
- Public keys: none
- Security group names: default
- Instance type: m1.small

However, each instance has certain unique metadata.

Instance 1

Metadata	Value
instance-id	i-3ea74257
ami-launch-index	0
public-hostname	ec2-67-202-51-223.compute-1.amazonaws.com
public-ipv4	67.202.51.223

Metadata	Value
local-hostname	ip-10-251-50-35.ec2.internal
local-ipv4	10.251.50.35

Instance 2

Metadata	Value
instance-id	i-31a74258
ami-launch-index	1
public-hostname	ec2-67-202-51-224.compute-1.amazonaws.com
public-ipv4	67.202.51.224
local-hostname	ip-10-251-50-36.ec2.internal
local-ipv4	10.251.50.36

Instance 3

Metadata	Value
instance-id	i-31a74259
ami-launch-index	2
public-hostname	ec2-67-202-51-225.compute-1.amazonaws.com
public-ipv4	67.202.51.225
local-hostname	ip-10-251-50-37.ec2.internal
local-ipv4	10.251.50.37

Instance 4

Metadata	Value
instance-id	i-31a7425a
ami-launch-index	3
public-hostname	ec2-67-202-51-226.compute-1.amazonaws.com
public-ipv4	67.202.51.226
local-hostname	ip-10-251-50-38.ec2.internal
local-ipv4	10.251.50.38

Therefore, an instance can determine its portion of the user-supplied data through the following process.

Metadata Discovery Process

1	Determine the index in the launch group.
	`GET http://169.254.169.254/latest/meta-data/ami-launch-index`
	`1`

2	Retrieve the user data.
	`GET http://169.254.169.254/latest/user-data`
	`store-size=123PB backup-every=5min | replicate-every=1min | replicate-every=2min | replicate-every=10min | replicate-every=20min`

3	Extract the appropriate part of the user data. `user_data.split('|')[ami_launch_index]`

Using Shared AMIs

This section describes how to find and safely use shared AMIs. One of the easiest ways to get started with Amazon EC2 is to use a shared AMI that has the components you need and add custom content.

Find Shared AMIs

To find shared AMIs

- Enter the **ec2-describe-images** command (or the abbreviated **ec2dim** command) with a flag to filter the results.

Example

The following command displays a list of all public AMIs. The `-x all` flag shows AMIs executable by all AWS accounts (i.e., AMIs with public launch permissions). This includes AMIs you own with public launch permissions.

```
PROMPT> ec2dim -x all
```

The following command displays a list of AMIs for which you have explicit *launch permissions*. AMIs that you own are excluded from the list.

```
PROMPT> ec2dim -x self
```

The following command displays a list of AMIs owned by Amazon.

```
PROMPT> ec2dim -o amazon
```

The following command displays a list of AMIs owned by a particular AWS account.

```
PROMPT> ec2dim -o <target_uid>
```

The `<target_uid>` is the account ID that owns the AMIs you're looking for.

Tip

You can filter this list to return only certain types of AMIs of interest to you. For more information about how to filter the results, go to ec2-describe-images in the *Amazon Elastic Compute Cloud Command Line Reference*.

Safe Use of Shared AMIs

You launch AMIs at your own risk. Amazon cannot vouch for the integrity or security of AMIs shared by other EC2 users. Therefore, you should treat shared AMIs as you would any foreign code that you might consider deploying in your own data center and perform the appropriate due diligence.

Ideally, you should get the AMI ID from a trusted source (a web site, another EC2 user, etc). If you do not know the source of an AMI, we recommend that you search the forums for comments on the AMI before launching it. Conversely, if you have questions or observations about a shared AMI, feel free to use the AWS forums to ask or comment.

Amazon's public images have an aliased owner and display `amazon` in the userId field. This allows you to find Amazon's public images easily.

Note

Users cannot alias an AMI's owner.

If you plan to use a shared AMI, review the following table to confirm the instance is not doing anything malicious.

Launch Confirmation Process

1	Check the ssh authorized keys file. The only key in the file should be the key you used to launch the AMI.
2	Check open ports and running services.
3	Change the root password if it is not randomized on startup. For more information on randomizing the root password on startup, see Disable Password-Based Logins for Root (p. 48).
4	Check if SSH allows root password logins. See Disable Password-Based Logins for Root (p. 48) for more information on disabling root based password logins.
5	Check whether there are any other user accounts that might allow backdoor entry to your instance. Accounts with super user privileges are particularly dangerous.
6	Verify that all cron jobs are legitimate.

Paying for AMIs

Topics

Amazon EC2 integrates with Amazon DevPay, allowing developers to charge other EC2 users for the use of their AMIs or to provide support for instances. To learn more about Amazon DevPay go to the Amazon DevPay Developer Guide. For more information about charging for the use of your AMIs, or providing support, see Creating Paid AMIs (p. 38).

This section describes how to discover paid AMIs, launch paid AMIs, and launch instances with a support product code. Paid AMIs are AMIs you can purchase from other developers.

Note

All paid AMIs are backed by Amazon instance store.

Find Paid AMIs

There are several ways you can determine what paid AMIs are available for you to purchase. You can look for information about them on the Amazon EC2 resource center and forums. Alternatively, a developer might give you information about a paid AMI directly.

You can also tell if an AMI is a paid AMI by describing the image with the **ec2-describe-images** command. This command lists the product code associated with an AMI (see the following example). If the AMI is a paid AMI, it has a product code. Otherwise, it does not. You can then go to the Amazon EC2 resource center and forums, which might have more information about the paid AMI and where you can sign up to use it.

Note

You must sign up for a paid AMI before you can launch it.

To check if an AMI is paid

- Enter the following command:

  ```
  PROMPT> ec2-describe-images <ami_id>
  ```

  The <*ami_id*> is the AMI ID.

  The command returns numerous fields that describe the AMI. If a product code (e.g. `D6F6052A)` is present in the output, the AMI is a paid AMI.

Example

This example shows an **ec2-describe-images** call describing a paid AMI. The product code is ACD42B6F.

```
PROMPT> ec2-describe-images ami-a5bf59cc
IMAGE   ami-a5bf59cc    cloudmin-2.6-paid/image.manifest.xml    541491349868
  available public   ACD42B6F        i386    machine
instance-store
```

Purchase a Paid AMI

You must sign up for (purchase) the paid AMI before you can launch it.

Typically a seller of a paid AMI presents you with information about the AMI, its price, and a link where you can buy it. When you click the link, you're first asked to log in with an Amazon.com login, and then you are taken to a page where you see the paid AMI's price and you confirm you want to purchase the AMI.

Important

You don't get the discount from Amazon EC2 Reserved Instances with paid AMIs. That is, if you purchase Reserved Instances, you don't get the lower price associated with them when you launch a paid AMI. You always pay the price that the seller of the paid AMI specified. For more information about Reserved Instances, see Reserved Instances.

Launch a Paid AMIs

This section describes how to launch paid AMIs and launch instances with a support product code.

After you purchase a paid AMI, you can launch instances of it. Launching a paid AMI is the same as launching any other AMI. No additional parameters are required. The instance will be charged according to the rates set by the owner of the AMI (which will be more than the base Amazon EC2 rate).

To launch a paid AMI

- Enter the following command:

  ```
  PROMPT> ec2-run-instances <ami_id>
  ```

The <*ami_id*> is the AMI ID.

Note

The owner of a paid AMI will be able to confirm if a particular instance was launched using their paid AMI.

Example

This example shows the command used to launch the ami-2bb65342 AMI.

```
PROMPT> ec2-run-instances ami-2bb65342
RESERVATION r-a034c7c9 999988887777 default
INSTANCE i-31a7425a ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
```

Using Paid Support

The paid AMI feature also allows developers to offer support for software (or derived AMIs). Developers can create support products that you can sign up to use. With this model, the developer provides you with a product. During sign-up for the product, the developer gives you a product code for that product, which you must then associate with your own AMI. This allows the developer to confirm that your instance is eligible for support. It also ensures that when you run instances of the product, you are charged according to the developer's terms for the product.

Important

If you've purchased Amazon EC2 Reserved Instances, you can't use them with supported AMIs. That is, if you associate a product code with one of your AMIs, you don't get the lower price associated with your Reserved Instances when you launch that AMI. You always pay the price that the seller of the support product specified. For more information about Reserved Instances, see Reserved Instances.

To associate the product code with your AMI

- Enter the **ec2-modify-image-attribute** command:

```
PROMPT>  ec2-modify-image-attribute <ami_id> --product-code <product_code>
```

The <*ami_id*> is the AMI ID and <*product_code*> is the product code.

Important

Once set, the product code attribute cannot be changed or removed.

To launch a paid AMI, no additional parameters are required for `ec2-run-instances`. The instance is charged according to the rates set by the AMI owner.

Example

The following command associates the *ami-2bb65342* AMI with the *774F4FF8* product code.

```
PROMPT>  ec2-modify-image-attribute ami-2bb65342 --product-code 774F4FF8
productCodes        ami-2bb65342            productCode   774F4FF8
```

The following command launches the *ami-2bb65342* paid AMI.

```
PROMPT> ec2-run-instances ami-2bb65342
RESERVATION r-a034c7c9 999988887777 default
INSTANCE i-31a7425a ami-2bb65342 pending 0 m1.small 2010-03-19T13:59:03+0000
us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
```

Bills for Paid and Supported AMIs

At the end of each month, you receive an email with the amount your credit card has been charged for using the paid or supported AMIs during the month. This bill is separate from your regular Amazon EC2 bill.

At any time, you can view the usage information for your paid and supported AMIs (go to http://www.amazon.com/dp-applications).

What to Do If an Instance Immediately Terminates

We recommend that when you launch an instance, you immediately check its status to confirm that it goes to the *running* status and not *terminated*.

Why would an Amazon EBS-backed instance immediately terminate? Following are a few reasons why:

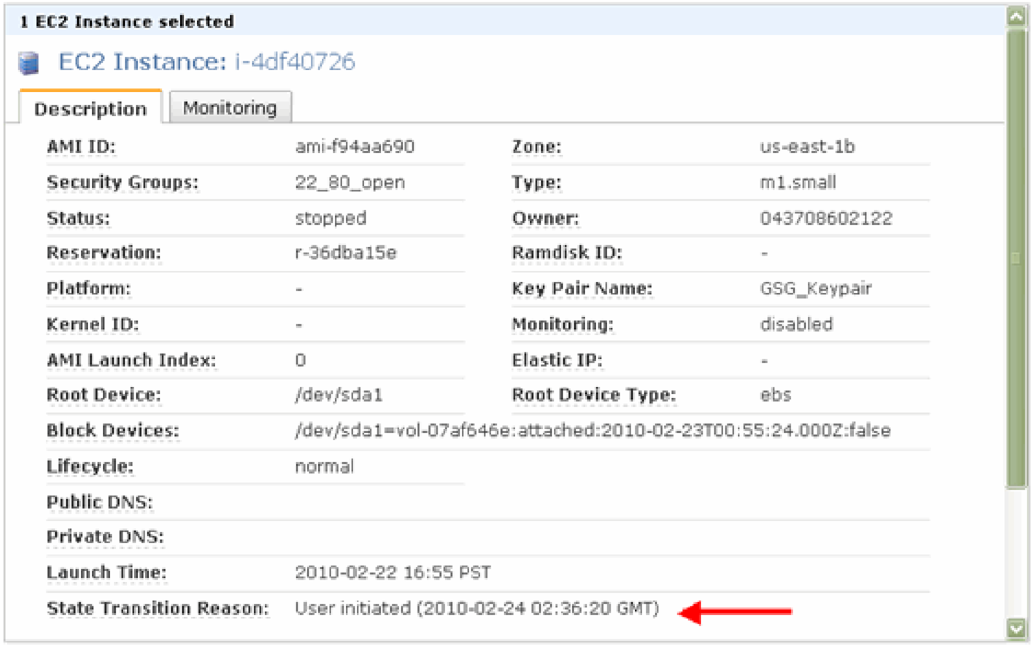
- You've reached your volume limit. Your account is limited to 5000 volumes, or 20 TiB in total volume storage, whichever you reach first. You can request to increase your volume limit by completing the Amazon EBS Volume Limit Request Form.
- The AMI is missing a required part.
- The snapshot is corrupt.

This section describes how to get information about why the instance terminated.

AWS Management Console

To get information about why the instance terminated

1. Go to the **Instances** page in the console and view the instance details.



1 EC2 Instance selected

EC2 Instance: i-4df40726

AMI ID:	ami-f94aa690	Zone:	us-east-1b
Security Groups:	22_80_open	Type:	m1.small
Status:	stopped	Owner:	043708602122
Reservation:	r-36dba15e	Ramdisk ID:	-
Platform:	-	Key Pair Name:	GSG_Keypair
Kernel ID:	-	Monitoring:	disabled
AMI Launch Index:	0	Elastic IP:	-
Root Device:	/dev/sda1	Root Device Type:	ebs
Block Devices:	/dev/sda1=vol-07af646e:attached:2010-02-23T00:55:24.000Z:false		
Lifecycle:	normal		
Public DNS:			
Private DNS:			
Launch Time:	2010-02-22 16:55 PST		
State Transition Reason:	User initiated (2010-02-24 02:36:20 GMT)		

2. Look at the **State Transition Reason**.

   The field shows information about why the instance terminated. For a normal instance, the field is usually blank or shows *User initiated* if you've requested to stop or start the instance.

Command Line Tools

To get information about why the instance terminated

1. Use the **ec2-describe-instances** command in verbose mode:

```
PROMPT>  ec2-describe-instances instance_id -v
```

   When you use verbose mode, the response includes the underlying SOAP request and the SOAP XML response.

2. In the XML response that's displayed, locate the `stateReason` element. It looks similar to the following example.

```
<stateReason>
   <code>Client.UserInitiatedShutdown</code>
   <message>Client.UserInitiatedShutdown: User initiated shutdown</message>
</stateReason>
```

   The preceding example shows what's displayed for a normal instance that you've stopped. For the instance that terminated immediately, the `code` and `message` elements will describe the reason for the termination (e.g., `VolumeLimitExceeded`).

API

To get information about why the instance terminated

1. Construct the following Query request. For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

```
https://ec2.amazonaws.com/
?Action=DescribeInstances
&...auth parameters...
```

2. In the XML response that's displayed, locate the `stateReason` element. It looks similar to the following example.

```
<stateReason>
   <code>Client.UserInitiatedShutdown</code>
  <message>Client.UserInitiatedShutdown: User initiated shutdown</message>
</stateReason>
```

   The preceding example shows what's displayed for a normal instance that you've stopped. For the instance that terminated immediately, the `code` and `message` elements will describe the reason for the termination (e.g., `VolumeLimitExceeded`).

Getting Console Output and Rebooting Instances

Console output is a valuable tool for problem diagnosis. It is especially useful for troubleshooting kernel problems and service configuration issues that could cause an instance to terminate or become unreachable before its SSH daemon can be started.

Similarly, the ability to reboot instances that are otherwise unreachable is valuable for both troubleshooting and general instance management.

Amazon EC2 instances do not have a physical monitor through which you can view their console output. They also lack physical controls that allow you to power up, reboot, or shut them down. To allow these actions, we provide them through the Amazon EC2 API and the command line tools.

Console Output

For Linux/UNIX instances, the Amazon EC2 instance console output displays the exact console output that would normally be displayed on a physical monitor attached to a machine. This output is buffered because the instance produces it and then posts it to a store where the instance's owner can retrieve it.

For Windows instances, the Amazon EC2 instance console output displays the last three system event log errors.

The posted output is not continuously updated; only when it is likely to be of the most value. This includes shortly after instance boot, after reboot, and when the instance terminates.

Note

Only the most recent 64 KB of posted output is stored, which is available for at least 1 hour after the last posting.

You can retrieve the console output for an instance using `GetConsoleOutput`. For more information, go to the Amazon Elastic Compute Cloud API Reference or Amazon Elastic Compute Cloud Command Line Reference.

Note

Only the instance owner can access the console output.

Instance Reboot

Just as you can reset a machine by pressing the reset button, you can reset Amazon EC2 instances using `RebootInstances`. For more information, go to the Amazon Elastic Compute Cloud API Reference or Amazon Elastic Compute Cloud Command Line Reference.

Caution

For Windows instances, this operation performs a hard reboot that might result in data corruption.

Connecting to Instances

Topics

This section describes how to connect to instances that you launched and how to transfer files between your local machine and your Amazon EC2 instance. For information on launching instances, see Launching and Using Instances (p. 75).

Prerequisites

- **Enable SSH/RDP traffic—**Open the instance's SSH or RDP port
  Before you try to connect, ensure that your Amazon EC2 instance accepts incoming traffic on the proper port. For Linux/UNIX instances, open port 22 for SSH access. For Windows instances, open port 3389 for RDP access. For more information, see Authorize Network Access to Your Instances (p. 111).
- **SSH/RDP client—**Install an SSH or RDP client
  Most Linux and UNIX machines include an SSH client by default. You can check for an SSH client by typing `ssh` at the command line. If your machine doesn't recognize the command, the OpenSSH project provides a free implementation of the full suite of SSH tools. For more information, go to http://www.openssh.org. Likewise, most Windows machines include an RDP client. For more information, go to the Microsoft website.
- **Instance ID—**Get the ID of your Amazon EC2 instance
  Retrieve the Instance ID of the Amazon EC2 instance you want to access. The Instance ID for all your instances are available in the AWS Management Console or through the CLI command ec2-describe-instances.
- **Private key—**Get the path to your private key
  You'll need the fully qualified path of the private key file associated with your instance. For more information on key pairs, see Getting an SSH Key Pair (p. 77).

To connect to an Amazon EC2 Linux/UNIX instance, use SSH. To connect to an Amazon EC2 Windows instance, use the Remote Desktop Protocol (RDP). The following sections provide more information on how to connect to your instance with these protocols.

Authorize Network Access to Your Instances

By default, Amazon EC2 instances do not permit access on any ports. To access your instance with SSH or RDP, your instance must allow incoming traffic on port 22 or 3389, respectively. To open a port for incoming traffic, add a security group rule to a security group that includes your instance. You can use the AWS Management Console or the command line tools (i.e., API tools). If you use the command line tools, use them on your local system, not on the instance itself.

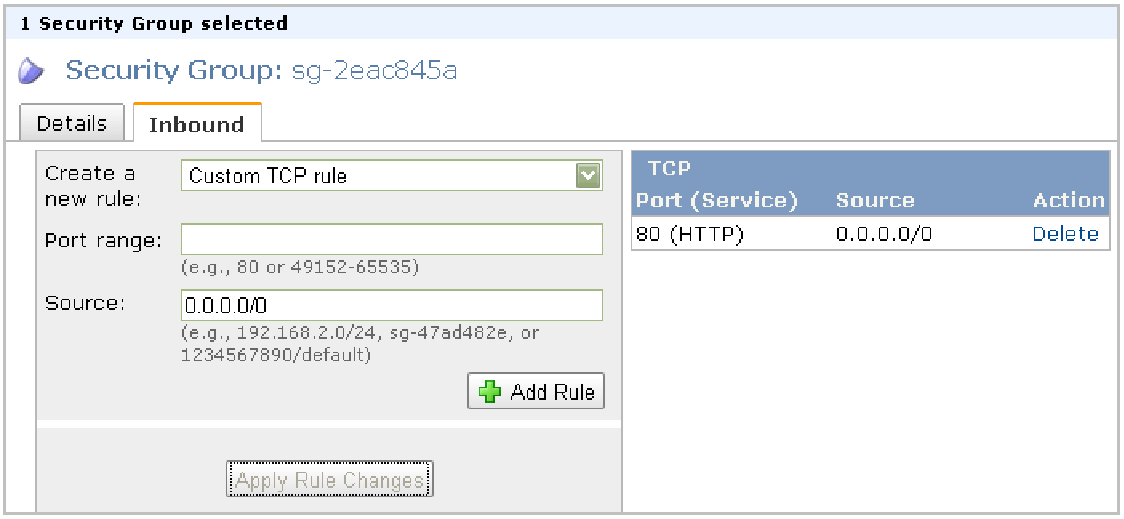
The following instructions authorize incoming SSH or RDP traffic for your instance, but only from your local system's public IP address. If your IP address is dynamic, you must authorize access each time it changes. To allow additional IP address ranges, add a new security group rule for each range.

> **Important**
>
> Get the public IP address of your local machine by going to a search engine (Google, Yahoo, Bing, etc.). Enter "what is my IP address" and use one of the provided services.

AWS Management Console

To add a rule to a security group forSSH or RDP access

1. Go to the Amazon EC2 console in the AWS Management Console.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. Select an EC2 security group that includes your instance.
   Its rules appear on the **Inbound** tab in the lower pane.



1 Security Group selected

Security Group: sg-2eac845a

Create a new rule: Custom TCP rule

Port range:
(e.g., 80 or 49152-65535)

Source: 0.0.0.0/0
(e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

TCP Port (Service)	Source	Action
80 (HTTP)	0.0.0.0/0	Delete

4. To add a rule for SSH or RDP:
   a. From the **Create a new rule:** drop-down list, select**SSH** or **RDP**.
   b. In the **Source** field, specify your local system's public IP address in CIDR notation. For example, if your IP address is 203.0.113.0, enter 203.0.113.0/32.
5. Click **Add Rule**.
   An asterisk appears on the **Inbound** tab.
6. Click **Apply Rule Changes**.
   The new rule is created and applied to all instances that belong to the security group.

Command Line Interface

To add a rule to a security group for SSH access

- Enter the `ec2-authorize` command to open port 22 (SSH port) to your IP address.
  The following example adds a rule to the default security group that allows incoming traffic on port 22 from your IP address.

```
PROMPT>  ec2-authorize default -p 22 -s your_ip_address/32
GROUP default
PERMISSION default ALLOWS tcp 22 22 FROM CIDR your_ip_address/32
```

To add a rule to a security group for RDP access

- Enter the ec2-authorize command to open port 3389 (SSH port) to your IP address.
  The following example adds a rule to the default security group that allows incoming traffic on port 22 from your IP address.

  ```
  PROMPT>  ec2-authorize default -p 3389 -s your_ip_address/32
  GROUP default
  PERMISSION default ALLOWS tcp 3389 3389 FROM CIDR your_ip_address/32
  ```

Connecting to Linux/UNIX Instances from Linux/UNIX

Topics

This section describes how to connect to Linux and UNIX instances using SSH and SCP on a Linux/UNIX machine.

Connect to Linux/UNIX Instances from Linux/UNIX with SSH

Prerequisites

- **Enable SSH traffic—**Open the instance's SSH port
  Before you try to connect, ensure that your Amazon EC2 instance accepts incoming SSH traffic (usually on port 22). For more information, see Authorize Network Access to Your Instances (p. 111).
- **SSH client—**Install an SSH client
  Most Linux and UNIX machines include an SSH client by default. You can check for an SSH client by typing `ssh` at the command line. If your machine doesn't recognize the command, the OpenSSH project provides a free implementation of the full suite of SSH tools. For more information, go to http://www.openssh.org.
- **Instance ID—**Get the ID of your Amazon EC2 instance
  Retrieve the Instance ID of the Amazon EC2 instance you want to access. The Instance ID for all your instances are available in the AWS Management Console or through the CLI command ec2-describe-instances.
- **Instance's public DNS—**Get the public DNS of your Amazon EC2 instance
  Retrieve the public DNS of the Amazon EC2 instance you want to access. You can find the public DNS for your instance using the AWS Management Console or by calling the CLI command `ec2-describe-instances`. The format of an instance's public DNS is ec2-w-x-y-z-compute-1.amazonaws.com where w, x, y, and z each represents a number between 0 and 255 inclusive.
- **Private key—**Get the path to your private key
  You'll need the fully qualified path of the private key file associated with your instance. For more information on key pairs, see Getting an SSH Key Pair (p. 77).

To use SSH to connect

1. If you've launched a public AMI that you have not rebundled, run the `ec2-get-console-output` command on your local system (not on the instance), and locate the `SSH HOST KEY FINGERPRINTS` section. For more information, go to `ec2-get-console-output` in the *Amazon Elastic Compute Cloud Command Line Reference*.

```
PROMPT>  ec2-get-console-output instance_id

...
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 2048 bc:89:29:c6:45:4b:b3:e2:c1:41:81:22:cb:3c:77:54
/etc/ssh/ssh_host_key.pub
ec2: 2048 fc:8d:0c:eb:0e:a6:4a:6a:61:50:00:c4:d2:51:78:66
/etc/ssh/ssh_host_rsa_key.pub
ec2: 1024 b5:cd:88:6a:18:7f:83:9d:1f:3b:80:03:10:17:7b:f5
/etc/ssh/ssh_host_dsa_key.pub
ec2: -----END SSH HOST KEY FINGERPRINTS-----
...
```

Note the fingerprints so that you can compare them to the fingerprints of the instance.

2. In a command line shell, change directories to the location of the private key file that you created when you launched the instance.
3. Use the `chmod` command to make sure your private key file isn't publicly viewable. For example, if your private key file were `My_Keypair.pem`, you would enter:

```
chmod 400 My_Keypair.pem
```

4. Connect to your instance using the instance's public DNS name (available through the AWS Management Console or the `ec2-describe-instances` command). For example, if the key file is `My_Keypair.pem` and the instance's DNS name is ec2-184-72-204-112.compute-1.amazonaws.com, use the following command.

```
ssh -i /example/My_Keypair.pem root@ec2-184-72-204-112.compute-1.amazonaws.com
```

Note

Some AMIs let you log in as root, but some require that you log in with the username ec2-user. For log in information for your chosen AMI, contact your AMI provider directly or go to Amazon Machine Images(AMIs) page, then locate and click your AMI on the list.

You'll see a response like the following.

```
The authenticity of host 'ec2-184-72-204-112.compute-1.amazonaws.com
(10.254.142.33)'
can't be established.
RSA key fingerprint is fc:8d:0c:eb:0e:a6:4a:6a:61:50:00:c4:d2:51:78:66.
Are you sure you want to continue connecting (yes/no)? yes
```

Important

If you've launched a public AMI, verify that the fingerprint matches the fingerprint from the output of the `ec2-get-console-output` command. If it doesn't, someone might be attempting a "man-in-the-middle" attack.

5. Enter **yes**.

You'll see a response like the following.

```
Warning: Permanently added 'ec2-184-72-204-112.compute-1.amazonaws.com'
(RSA)
to the list of known hosts.
```

Transfer Files to Linux/UNIX Instances from Linux/UNIX with SCP

One way to transfer files between your local machine and a Linux/UNIX instance is to use Secure Copy (SCP). This section describes how to transfer files with SCP. The procedure is very similar to the procedure for connecting to an instance with SSH.

Prerequisites

- **Enable SSH traffic—**Open the instance's SSH port
  Before you try to connect, ensure that your Amazon EC2 instance accepts incoming SSH traffic (usually on port 22). For more information, see Authorize Network Access to Your Instances (p. 111).
- **SCP client—**Install an SCP client
  Most Linux and UNIX machines include an SCP client by default. If yours doesn't, the OpenSSH project provides a free implementation of the full suite of SSH tools, including an SCP client. For more information, go to http://www.openssh.org.
- **Instance ID—**Get the ID of your Amazon EC2 instance
  Retrieve the Instance ID of the Amazon EC2 instance you want to access. The Instance ID for all your instances are available in the AWS Management Console or through the CLI command ec2-describe-instances.
- **Instance's public DNS—**Get the public DNS of your Amazon EC2 instance
  Retrieve the public DNS of the Amazon EC2 instance you want to access. You can find the public DNS for your instance using the AWS Management Console or by calling the CLI command `ec2-describe-instances`. The format of an instance's public DNS is ec2-w-x-y-z-compute-1.amazonaws.com where w, x, y, and z each represents a number between 0 and 255 inclusive.
- **Private key—**Get the path to your private key
  You'll need the fully qualified path of the private key file associated with your instance. For more information on key pairs, see Getting an SSH Key Pair (p. 77).

The following procedure steps you through using SCP to transfer a file. If you've already connected to the instance with SSH and have verified its fingerprints, you can start with the step that contains the SCP command (step 4).

To use SCP to transfer a file

1. If you've launched a public AMI that you have not rebundled, run the `ec2-get-console-output` command on your local system (not on the instance), and locate the `SSH HOST KEY FINGERPRINTS` section. For more information, go to `ec2-get-console-output` in the *Amazon Elastic Compute Cloud Command Line Reference*.

```
PROMPT>  ec2-get-console-output instance_id

...
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 2048 bc:89:29:c6:45:4b:b3:e2:c1:41:81:22:cb:3c:77:54
/etc/ssh/ssh_host_key.pub
ec2: 2048 fc:8d:0c:eb:0e:a6:4a:6a:61:50:00:c4:d2:51:78:66
```

```
/etc/ssh/ssh_host_rsa_key.pub
ec2: 1024 b5:cd:88:6a:18:7f:83:9d:1f:3b:80:03:10:17:7b:f5
/etc/ssh/ssh_host_dsa_key.pub
ec2: -----END SSH HOST KEY FINGERPRINTS-----
...
```

Note the fingerprints so that you can compare them to the fingerprints of the instance.

2. In a command line shell, change directories to the location of the private key file that you created when you launched the instance.
3. Use the `chmod` command to make sure your private key file isn't publicly viewable. For example, if your file were `My_Keypair.pem`, you would enter:

```
chmod 400 My_Keypair.pem
```

4. Transfer a file to your instance using the instance's public DNS name (available through the AWS Management Console or the `ec2-describe-instances` command). For example, if the key file is `My_Keypair.pem`, the file to transfer is samplefile.txt, and the instance's DNS name is ec2-184-72-204-112.compute-1.amazonaws.com, use the following command to copy the file to the ec2-user home directory.

```
scp -i My_Keypair.pem samplefile.txt ec2-user@ec2-184-72-204-112.compute-
1.amazonaws.com:~
```

Note

Some AMIs let you log in as root, but some require that you log in with the username ec2-user. For log in information for your chosen AMI, contact your AMI provider directly or go to Amazon Machine Images(AMIs) page, then locate and click your AMI on the list.

You'll see a response like the following.

```
The authenticity of host 'ec2-184-72-204-112.compute-1.amazonaws.com
(10.254.142.33)'
can't be established.
RSA key fingerprint is fc:8d:0c:eb:0e:a6:4a:6a:61:50:00:c4:d2:51:78:66.
Are you sure you want to continue connecting (yes/no)? yes
```

Important

If you've launched a public AMI, verify that the fingerprint matches the fingerprint from the output of the `ec2-get-console-output` command. If it doesn't, someone might be attempting a "man-in-the-middle" attack.

5. Enter **yes**.

You'll see a response like the following.

```
Warning: Permanently added 'ec2-184-72-204-112.compute-1.amazonaws.com'
(RSA)
to the list of known hosts.
Sending file modes: C0644 20 samplefile.txt
```

```
Sink: C0644 20 samplefile.txt
samplefile.txt                                100%   20     0.0KB/s   00:00
```

To transfer files in the other direction, i.e., from your Amazon EC2 instance to your local machine, simply reverse the order of the host parameters. For example, to transfer the samplefile.txt file from your Amazon EC2 instance back to the home directory on your local machine as samplefile2.txt, use the following command on your local machine.

```
scp -i My_Keypair.pem ec2-user@ec2-184-72-204-112.compute-1.amazon
aws.com:~/samplefile.txt ~/samplefile2.txt
```

Connecting to Linux/UNIX Instances from Windows Using PuTTY

Topics

- Getting PuTTY (p. 117)
- Converting Your Private Key (p. 118)
- Connecting Using PuTTY SSH (p. 119)
- Transferring Files with PSCP (p. 122)

To connect to your Linux/UNIX instance from a Windows machine, use an SSH client. The following instructions explain how to use PuTTY, a free SSH client for Windows machines.

Prerequisites

- **Enable SSH traffic—**Open the instance's SSH port
  Before you try to connect, ensure that your Amazon EC2 instance accepts incoming SSH traffic (usually on port 22). For more information, see Authorize Network Access to Your Instances (p. 111).
- **Instance ID—**Get the ID of your Amazon EC2 instance
  Retrieve the Instance ID of the Amazon EC2 instance you want to access. The Instance ID for all your instances are available in the AWS Management Console or through the CLI command ec2-describe-instances.
- **Instance's public DNS—**Get the public DNS of your Amazon EC2 instance
  Retrieve the public DNS of the Amazon EC2 instance you want to access. You can find the public DNS for your instance using the AWS Management Console or by calling the CLI command `ec2-describe-instances`. The format of an instance's public DNS is ec2-w-x-y-z-compute-1.amazonaws.com where w, x, y, and z each represents a number between 0 and 255 inclusive.
- **Private key—**Get the path to your private key
  You'll need the fully qualified path of the private key file associated with your instance. For more information on key pairs, see Getting an SSH Key Pair (p. 77).

Getting PuTTY

To download and install PuTTY

- Go to http://www.chiark.greenend.org.uk/~sgtatham/putty/ and follow the instructions there.

Note

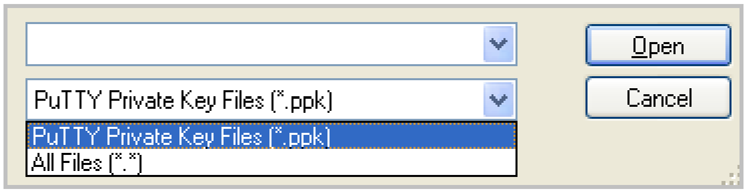
Other tools in the PuTTY suite are PuTTYgen, a key generation program, and pscp, a secure copy command line tool. The different PuTTY tools are separate applications. You can install them separately or install the entire suite with a simple Windows installer. The following instructions assume you've installed the entire suite and can access all the components from the Windows Start menu.

Converting Your Private Key

PuTTY does not natively support the private key format generated by Amazon EC2. Fortunately, PuTTY has a tool called PuTTYgen, which can convert keys to the required PuTTY format.

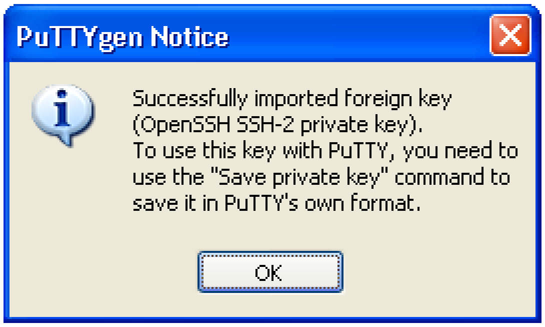
To convert your private key

1. Start PuTTYgen (e.g., from the **Start** menu, click **All Programs > PuTTY > PuTTYgen**).
2. Click **Load** and browse to the location of the private key file that you want to convert (e.g., `GSG_Keypair.pem`). By default, PuTTYgen displays only files with extension `.ppk`; you'll need to change that to display files of all types in order to see your `.pem` key file. The private key file must end with a newline character or PuTTYgen cannot load it correctly.


Open
Cancel
PuTTY Private Key Files (*.ppk)
PuTTY Private Key Files (*.ppk)
All Files (*.*)

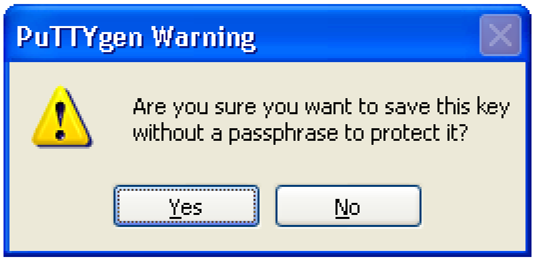
3. Select your `.pem` key file and click **Open**.

   PuTTYgen displays the following message.



Successfully imported foreign key (OpenSSH SSH-2 private key).
To use this key with PuTTY, you need to use the "Save private key" command to save it in PuTTY's own format.

   When you click OK, PuTTYgen displays a dialog box with information about the key you loaded, such as the public key and the fingerprint. The keys that Amazon EC2 generates are 1024-bit SSH-2 RSA keys.
4. Click **Save private key** to save the key in PuTTY's format.
   PuTTYgen asks if you want to save the key without a passphrase.



Are you sure you want to save this key without a passphrase to protect it?

5. Click **Yes**.

   Note

   A passphrase on a private key is an extra layer of protection, so even if your private key is discovered, it will not be usable without the passphrase. The downside to using a passphrase is that it makes automation harder because human intervention is needed to log on to an instance, or copy files to an instance. For this exercise, we're not using a passphrase.

6. Name the key with the same name you used for the key pair (e.g., GSG_Keypair). PuTTY automatically adds the `.ppk` file extension.

Your private key is now in the correct format for use with PuTTY. You can now connect to your instance using PuTTY's SSH client.

Connecting Using PuTTY SSH

You'll connect by starting a PuTTY SSH session.

To use SSH to connect

1. If you've launched a public AMI that you have not rebundled, run the `ec2-get-console-output` command on your local system (not on the instance), and locate the `SSH HOST KEY FINGERPRINTS` section. For more information, go to `ec2-get-console-output` in the *Amazon Elastic Compute Cloud Command Line Reference*.

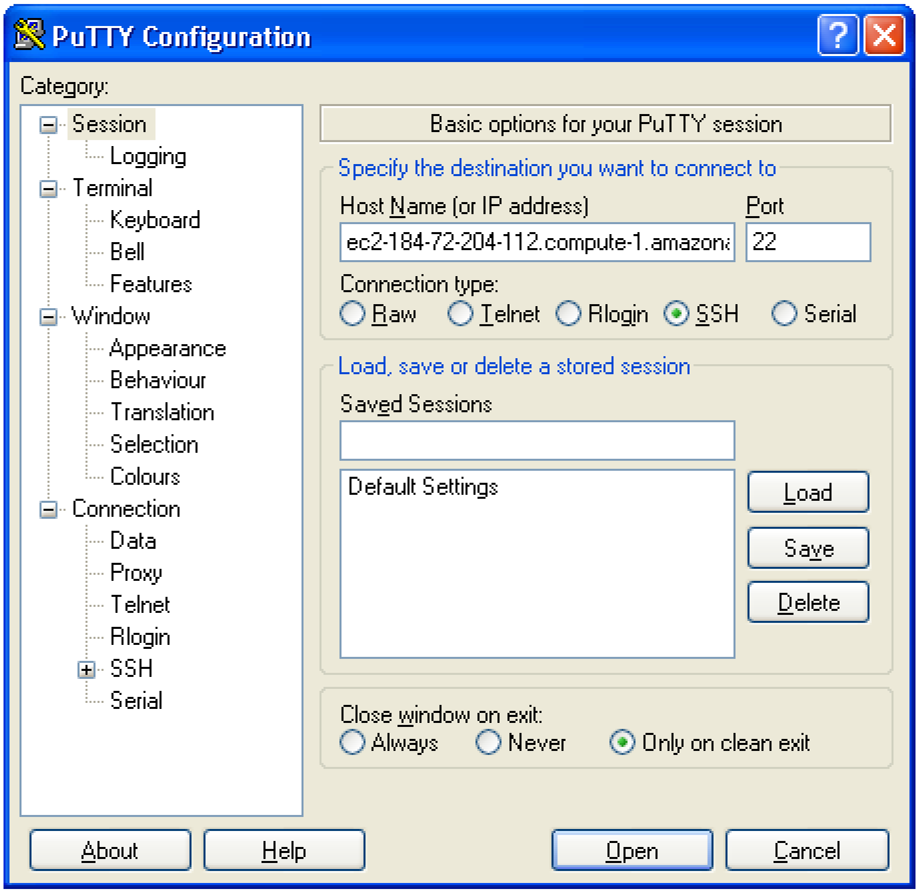
```
PROMPT>  ec2-get-console-output instance_id

...
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 2048 bc:89:29:c6:45:4b:b3:e2:c1:41:81:22:cb:3c:77:54
/etc/ssh/ssh_host_key.pub
ec2: 2048 fc:8d:0c:eb:0e:a6:4a:6a:61:50:00:c4:d2:51:78:66
/etc/ssh/ssh_host_rsa_key.pub
ec2: 1024 b5:cd:88:6a:18:7f:83:9d:1f:3b:80:03:10:17:7b:f5
/etc/ssh/ssh_host_dsa_key.pub
ec2: -----END SSH HOST KEY FINGERPRINTS-----
...
```

   Note the fingerprints so that you can compare them to the fingerprints of the instance.

2. Start PuTTY (e.g., from the **Start** menu, click **All Programs > PuTTY > PuTTY**).
   A dialog box opens with a **Category** menu on the left side. On the right side, the basic options for your PuTTY session are displayed.

3. In the **Host Name** field, enter the public DNS name of your instance (available through the AWS Management Console or the `ec2-describe-instances` command). You can optionally prefix the DNS name with `ec2-user@` to automatically log in with superuser privileges when the session opens.



Host Name (or IP address): ec2-184-72-204-112.compute-1.amazon‹
Port: 22

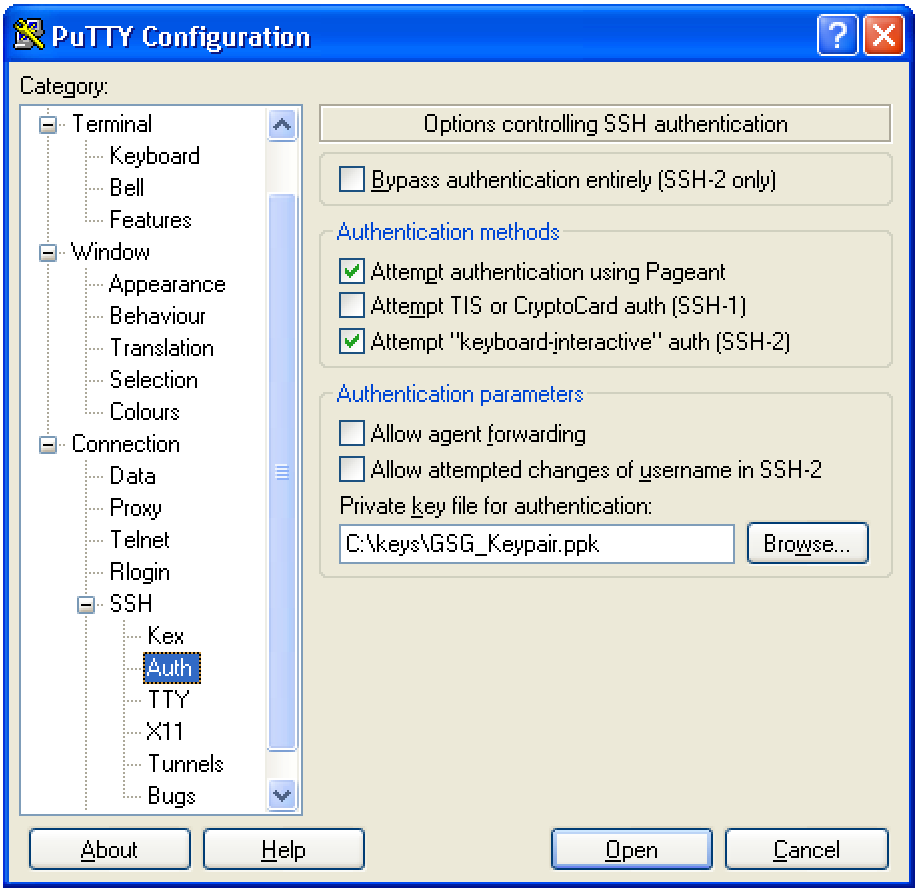
Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Saved Sessions:

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

4. In the **Category** menu, under **Connection**, click **SSH**, and then **Auth**.
   The options controlling SSH authentication are displayed.
5. Click **Browse** and navigate to the PuTTY private key file you generated in the preceding section.



Options controlling SSH authentication

□ Bypass authentication entirely (SSH-2 only)

Authentication methods

☑ Attempt authentication using Pageant
□ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding
□ Allow attempted changes of username in SSH-2

Private key file for authentication: C:\keys\GSG_Keypair.ppk

6. Click **Open**.
   An SSH session window opens and PuTTY displays a security alert asking if you trust the host you're connecting to.

   Important

   If you've launched a public AMI, verify that the fingerprint in the security alert matches the fingerprint from the output of the `ec2-get-console-output` command. If it doesn't, someone might be attempting a "man-in-the-middle" attack.

7. Click **Yes**.
8. In the SSH session window, log in as root (or ec2-user) if you didn't as part of starting the SSH session.

   Note

   Some AMIs let you log in as root, but some require you to log in with the username ec2-user. For log in information for your chosen AMI, contact your AMI provider directly or go to Amazon Machine Images(AMIs) page, then locate and click your AMI on the list.

   Note

   If you specified a passphrase when you converted your private key to PuTTY's format, you must provide that passphrase when you log in to the instance.

Transferring Files with PSCP

The PuTTY Secure Copy Client (PSCP) is a command-line tool that lets you transfer files between your Windows machine and your Linux/UNIX instance.

To use PSCP, you'll need the private key you generated in Converting Your Private Key (p. 118). You'll also need the public IP address of your Linux/UNIX instance.

The following example transfers the file `sample_file.txt` from a Windows machine to the `/usr/local` directory on a Linux/UNIX instance:

```
Prompt>pscp -i C:\GSG_Keypair.ppk C:\sample_file.txt root@ec2-184-72-204-
112.compute-1.amazonaws.com:/usr/local/sample_file.txt
```

If you prefer a graphical user interface (GUI), you can use an open source GUI tool named WinSCP. For more information, go to the WinSCP website.

Connecting to Windows Instances

Topics

This section describes how to connect to instances running Windows from local machines running Windows, Linux/UNIX, or Mac OS.

Prerequisites

- **Enable RDP traffic—**Open the instance's RDP port
  Before you try to connect, ensure that your Amazon EC2 instance accepts incoming RDP traffic (usually on port 3389). For more information, see Authorize Network Access to Your Instances (p. 111).
- **RDP client—**Install an RDP client
  Windows machines include an RDP client by default. For Mac OS X, you can use Microsoft's Remote Desktop Client. For Linux/UNIX, you can use rdesktop.
- **Instance ID—**Get the ID of your Amazon EC2 instance
  Retrieve the Instance ID of the Amazon EC2 instance you want to access. The Instance ID for all your instances are available in the AWS Management Console or through the CLI command ec2-describe-instances.
- **Private key—**Get the path to your private key
  You'll need the fully qualified path of the private key file associated with your instance. For more information on key pairs, see Getting an SSH Key Pair (p. 77).

Connect to Windows Instances with RDP

To connect to a Windows instance, you must retrieve the initial administrator password first, and then use it with Remote Desktop. You'll need the contents of the private key file that you created when you launched the instance (e.g., `GSG_Keypair.pem`).

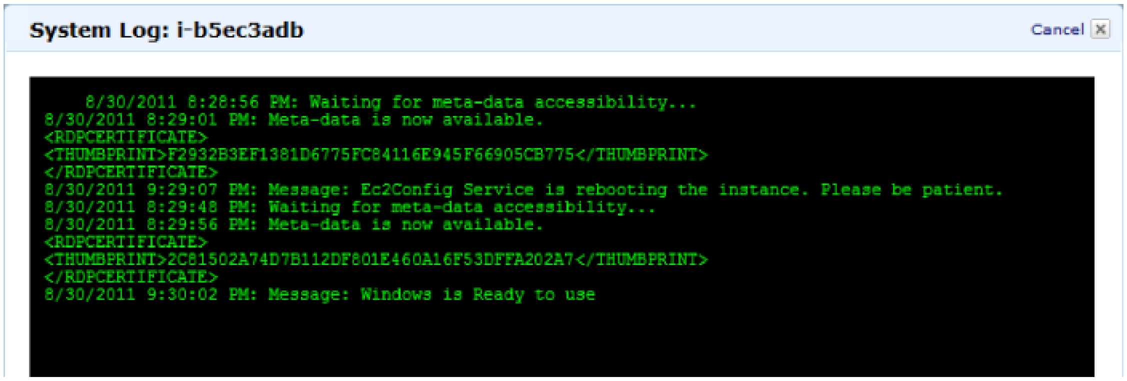
Note

The Windows password is only generated the first time an AMI is launched. It is not generated for rebundled AMIs or after the password is changed on an instance.

The password is encrypted using the key pair that you provided and stored within the <password> tags of the console output.

To connect to your Windows instance

1. If you've launched a public AMI that you have not rebundled, get the instance's RDP certificate.
    a. Go to the AWS Management Console and locate the instance on the **Instances** page.
    b. Right-click the instance and select **Get System Log**.
    The **System Log** dialog box is displayed (it might take a few minutes after the instance is launched before the RDP certificate is available).

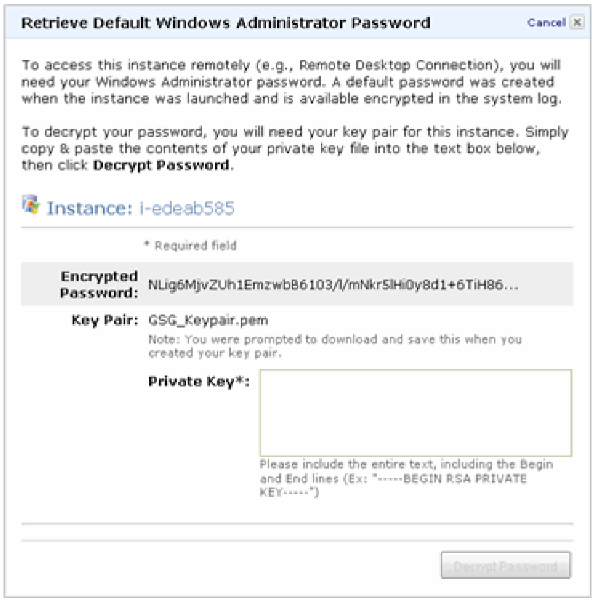


System Log: i-b5ec3adb

```
    8/30/2011 8:28:56 PM: Waiting for meta-data accessibility...
8/30/2011 8:29:01 PM: Meta-data is now available.
<RDPCERTIFICATE>
<THUMBPRINT>F2932B3EF1381D6775FC84116E945F66905CB775</THUMBPRINT>
</RDPCERTIFICATE>
8/30/2011 9:29:07 PM: Message: Ec2Config Service is rebooting the instance. Please be patient.
8/30/2011 8:29:48 PM: Waiting for meta-data accessibility...
8/30/2011 8:29:56 PM: Meta-data is now available.
<RDPCERTIFICATE>
<THUMBPRINT>2C81502A74D7B112DF801E460A16F53DFFA202A7</THUMBPRINT>
</RDPCERTIFICATE>
8/30/2011 9:30:02 PM: Message: Windows is Ready to use
```

A thumbprint is a series of hexadecimal numbers enclosed in a `<THUMBPRINT>` tag. For example, a thumbprint might look like `<THUMBPRINT>2C81502A74D7B112DF801E460A16F53DFFA202A7</THUMBPRINT>`. Note the thumbprints so that you can compare them to the thumbprints of the instance.

2. Retrieve the initial administrator password:
    a. Navigate to the directory where you stored the private key file when you launched the instance.
    b. Open the file in a text editor and copy the entire contents (including the first and last lines, which contain *BEGIN RSA PRIVATE KEY* and *END RSA PRIVATE KEY*).
    c. Go to the AWS Management Console and locate the instance on the **Instances** page.
    d. Right-click the instance and select **Get Windows Password**.
    The **Retrieve Default Windows Administrator Password** dialog box is displayed (it might take a few minutes after the instance is launched before the password is available).



To access this instance remotely (e.g., Remote Desktop Connection), you will need your Windows Administrator password. A default password was created when the instance was launched and is available encrypted in the system log.

To decrypt your password, you will need your key pair for this instance. Simply copy & paste the contents of your private key file into the text box below, then click **Decrypt Password**.

Instance: i-edeab585

* Required field

Encrypted Password: NLig6MjvZUh1EmzwbB6103/l/mNkr5lHi0y8d1+6TiH86...

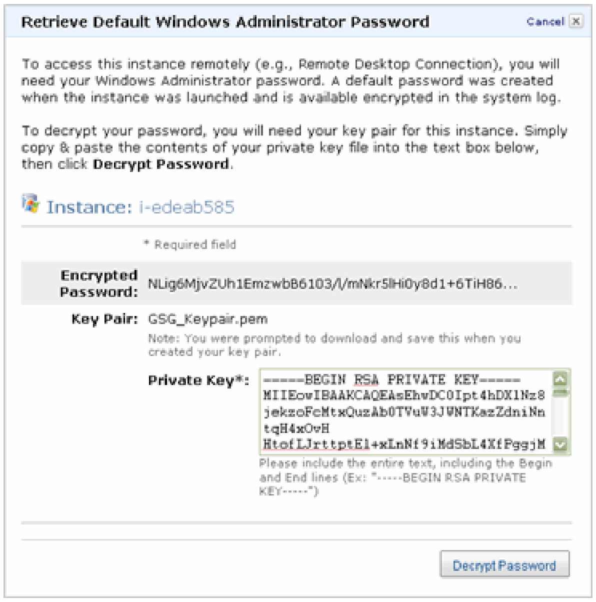
Key Pair: GSG_Keypair.pem

Note: You were prompted to download and save this when you created your key pair.

Private Key*:

Please include the entire text, including the Begin and End lines (Ex: "-----BEGIN RSA PRIVATE KEY-----")

e. Paste the contents of the private key file into the **Private Key** field.



To access this instance remotely (e.g., Remote Desktop Connection), you will need your Windows Administrator password. A default password was created when the instance was launched and is available encrypted in the system log.

To decrypt your password, you will need your key pair for this instance. Simply copy & paste the contents of your private key file into the text box below, then click **Decrypt Password**.

Instance: i-edeab585

* Required field

Encrypted Password: NLig6MjvZUh1EmzwbB6103/l/mNkr5lHi0y8d1+6TiH86...

Key Pair: GSG_Keypair.pem

Note: You were prompted to download and save this when you created your key pair.

Private Key*:

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAsEhvDC0Ipt4hDX1Nz8
jekzoFcMtxQuzAb0TVuW3JWNTKazZdniNn
tqH4xOvH
HtofLJrttptE1+xLnNf9iMdSbL4XfPggjM
```

Please include the entire text, including the Begin and End lines (Ex: "-----BEGIN RSA PRIVATE KEY-----")

f. Click **Decrypt Password**.
The console returns the default administrator password for the instance.

g. Save the password. You will need it to connect to the instance.

3. Connect to the instance using Remote Desktop:

   a. Go to the AWS Management Console and locate the instance on the **Instances** page.
   b. Right-click the instance and select **Connect**.
   c. Click **Download shortcut file**.
   Save the shortcut file to a convenient location on your local machine.

d. Launch the shortcut file.

e. Log in using `Administrator` as the username and the administrator password you got in the previous task as the password.
The Amazon EC2 instance returns a security alert.

f. To verify the instance, click **View Certificate**.
The **Certificate** page appears.

g. Click the **Details** tab.
The **Details** page appears.

h. Select **Thumbprint** and verify its value against the value you wrote down previously.

> **Important**
>
> If you've launched a public AMI, verify that the thumbprint matches a thumbprint from the instance's RDP certificate. If it doesn't, someone might be attempting a "man-in-the-middle" attack.

i. If it matches, click **OK** and then **Yes**.
The Remote Desktop Connection client connects to the instance.

Transfer Files to Windows Instances from Windows

One way to transfer files between an Amazon EC2 Windows instance and your local Windows machine is to use the local file sharing feature of Windows Remote Desktop. If you enable this option in your Windows Remote Desktop Connection software, you can access your local files from your Amazon EC2 Windows instances. You can access local files on hard disk drives, DVD drives, portable media drives, and mapped network drives.

For information about this feature, go to the Microsoft Support website or go to The most useful feature of Remote Desktop I never knew about on the MSDN Blogs website.

Using Storage

Topics

This section is intended to give you information about the primary data storage options supported by Amazon Elastic Compute Cloud. These storage options include Amazon Elastic Block Store (Amazon EBS), Amazon Simple Storage Service (Amazon S3), and Amazon EC2 instance storage. After reading this section, you should have a good idea of how you can use these storage options to meet your specific requirements.

Amazon EC2 provides you with flexible, cost effective, and easy-to-use data storage options for your instances. Each option has a unique combination of performance and durability. These storage options can be used alone or with others to suit the requirements of your application. For example, you can use Amazon Simple Storage Service (Amazon S3) to store backup copies of your data and applications. You can use Amazon Elastic Block Store (Amazon EBS) as a primary storage device for your data requiring frequent and granular updates. Amazon instance storage is ideal for using as temporary storage for constantly changing information, such as caches and buffers.

Amazon Elastic Block Store (Amazon EBS) is a durable, block-level storage volume that you can attach to a single Amazon EC2 running instance. Amazon EBS volumes behave like raw, unformatted, external block devices you can attach. They persist independently from the running life of an Amazon EC2 instance. After an Amazon EBS volume is attached to an instance, you can use it like any other physical hard drive. For more information, see Amazon Elastic Block Store (p. 126).

Amazon Simple Storage Service (Amazon S3) is a repository for Internet data. Amazon S3 provides access to reliable, fast, and inexpensive data storage infrastructure. It is designed to make web-scale computing easy by enabling you to store and retrieve any amount of data, at any time, from within Amazon EC2 or anywhere on the web. For more information, see Amazon Simple Storage Service (p. 150).

Amazon EC2 instance store provides temporary block-level storage for Amazon EC2 instances. When an instance is created from an Amazon Machine Image (AMI), in most cases it comes with a preconfigured block of pre-attached instance store. Instance store is also known as an *ephemeral store*. The data on the instance store volumes persists only during the life of the associated Amazon EC2 instance. For more information, see Amazon EC2 Instance Storage (p. 151).

Everytime you launch an instance from an AMI, a root storage device is created for that instance. The root storage device is created either from a template stored in Amazon S3 or from an Amazon EBS snapshot and contains all the information necessary to boot the instance. The template on Amazon S3 is copied onto the instance store when the instance is launched. All Amazon AMIs are categorized as either *backed by Amazon EC2 instance store* or *backed by Amazon EBS*, depending on where the root device information is stored. There are significant differences between the instances launched from these two types of AMIs. These differences determine the design and the architecture of your applications. For more information, see Root Device Storage (p. 160).

Every AMI and instance has a *block device mapping* structure that specifies the block devices attached to the instance. A block device mapping structure contains one or more items, and each item describes the mapping for a single block device. Each item specifies the device name, and whether the device is mapped to an instance store, an Amazon EBS volume, or to nothing. Block device mapping information is included as part of the AMI details displayed in the AWS Management Console. You can also get the information by describing the AMI with the command line tools or API. For more information, see Block Device Mapping (p. 167).

Amazon Elastic Block Store

Topics

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with Amazon EC2 instances. Amazon EBS volumes are highly available and reliable storage volumes that can be attached to any running instance. The attached Amazon EBS volumes are exposed as storage volumes that persist independently from the life of the instance.

You can attach multiple volumes to the same instance within the limits specified by your AWS account. As per your AWS account, you can have up to 5000 EBS volumes or 20Tib in total volume storage, whichever you reach first. Each EBS volume can be from 1GiB to 1 TiB. If you find the default volume limits not meeting your specific requirements, go to the Amazon EBS Volume Limit Request Form to request for more.

Data Availability

When you create an Amazon EBS volume in an Availability Zone, it is automatically replicated within that zone to prevent data loss due to failure of any single hardware component. After you create a volume, you can attach it to any Amazon EC2 instance in the same Availability Zone. Once attached, the volume appears as a native block device similar to a hard drive or other physical device. At that point, the instance can interact with the volume just as it would with a local drive; the instance can format the EBS volume with a file system such as ext3 (Linux) or NTFS (Windows) and install applications. You use the file system to access the stored files.

An EBS volume can be attached to only one instance at a time within the same Availability Zone. However, multiple volumes can be attached to a single instance. If you attach multiple volumes to a device that you have named, you can stripe data across the volumes for increased I/O and throughput performance.

You can get monitoring data for your Amazon EBS volumes at no additional charge (this includes data for the root device volumes for Amazon EBS-backed instances). For more information, see Monitoring Amazon EBS Volumes (p. 353).

Data Persistence

An Amazon EBS volume is off-instance storage that can persist independently from the life of an instance (An exception to this rule is an EBS volume created and attached to an instance at launch. See the following paragraph for more information). On instance termination, the attached EBS volume automatically detaches from the instance with the data intact. The volume can then be reattached to a new instance, allowing for quick recovery. If you are using an EBS-backed instance, you can stop and restart that instance without affecting the data stored in the attached volume. The volume remains attached throughout the stop-start cycle. This allows you to process and store the data set indefinitely, only using the processing and storage resources when required. The data set persists in the volume until the volume is deleted. After a volume is deleted it cannot be attached to any instance.

Note

You continue to pay for the volume usage as long as the data persists.

EBS volumes that are created and attached to an instance at launch are, by default, deleted when that instance is terminated. This behavior can be modified by changing the value of the flag `DeleteOnTermination` to `false` while launching the instance. The modified value causes the volume to persist after the instance is terminated. This volume becomes available to be attached to another instance. EBS volumes attached to an instance after it is already running are, by default, not deleted on instance termination.

Note

If the EBS volume is created when the instance is created, then the termination of the instance deletes the volumes by default. If the volume is created explicitly, then it must be deleted explicitly.

Snapshots

Amazon EBS provides the ability to create snapshots (backups) of any Amazon EC2 volumes. A snapshot of an EBS volume causes a copy of the data in the volume to be written to Amazon S3, where it is stored redundantly in multiple Availability Zones. After writing data to a volume, you can periodically create a snapshot of the volume to use as a baseline for new volumes. The volume need not be attached to a running instance in order to take a snapshot. The snapshots can be used to create multiple new Amazon EBS volumes, expand the size of a volume, or move volumes across Availability Zones. When a new volume is created using a snapshot, it is an exact replica of the original volume. By optionally specifying a different volume size or a different Availability Zone, you can use this functionality to increase the size of an existing volume or to create duplicate volumes in new Availability Zones. The snapshots can also be shared with specific AWS accounts or made public.

Amazon EBS snapshots are incremental backups, meaning that only the blocks on the device that have changed since your last snapshot will be saved. If you have a device with 100GiB of data, but only 5GiB of data have changed since your last snapshot, only the 5GiB of modified data will be stored back to Amazon S3. Even though snapshots are saved incrementally, the snapshot deletion process is designed so that you need to retain only the most recent snapshot in order to restore the volume.

To help categorize and manage your volumes and snapshots, you can tag them with metadata of your choice. For more information, see Using Tags (p. 275).

Amazon EBS Usage Scenarios

Amazon EBS is used when data changes frequently and requires long-term persistence. Amazon EBS is particularly well-suited for use as the primary storage for a file system, database, or for any applications that require fine granular updates and access to raw, unformatted block-level storage. Amazon EBS is particularly helpful for database-style applications that frequently encounter many random reads and writes across the dataset.

The combination of Amazon EC2 and Amazon EBS makes it easy to design and develop steps to optimize the performance and resource utilization of your servers and storage devices.

The following task lists describe some of the common tasks performed when using Amazon EBS.

Expanding the Storage Space of a Volume

1	Create a snapshot of the volume attached to your running instance.
2	Create a new volume from the snapshot by specifying a larger size than the original volume.
3	Attach the new volume to your instance.
4	Detach and delete the old volume.

Adding Multiple Copies of Your Volume to Your EC2 Instances

1	Create a snapshot of the volume attached to your running instance.
2	Create the new volumes you need using the snapshot.
3	Attach the newly created volumes to your EC2 instances.

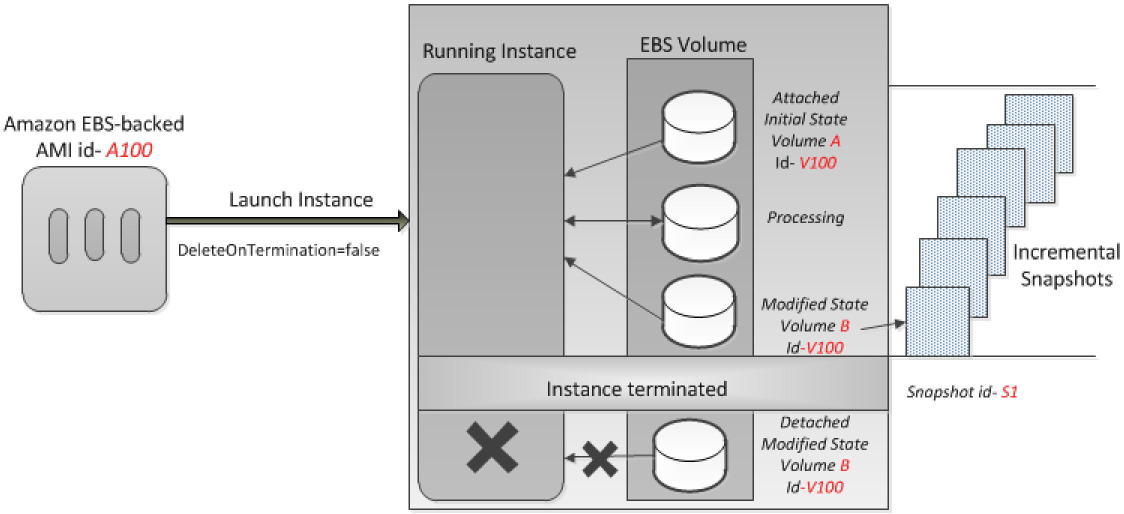
Reducing Use of Storage Resources When Traffic Decreases

1	Identify the volume or volumes that are not currently needed.
2	[optional] Create snapshots of the identified volumes.
3	Detach the volume(s) from the instance. **Note** The data will persist in the detached volume. This volume can be attached at a later time to any running instance.
4	To remove the data from the volume, delete the volume. **Note** The data in this volume is deleted. This volume cannot be attached to any instance.

Data Persistence after Instance Termination

By default, any volumes that were created when the instance launched are automatically deleted when the instance terminates (in other words, their `DeleteOnTermination` flags are set to `true` by default). However, any volumes that you attached *after* the instance was running are not automatically deleted (their `DeleteOnTermination` flags are set to `false` by default). You can change the default behavior by setting the `DeleteOnTermination` flag to `false` when you launch the instance. For an example of how to change the flag at instance launch, see Using Amazon EC2 Root Device Storage (p. 165).

The following diagram and task list describe the different states of a volume from the time it is attached to an instance until the time the instance terminates.



Persisting Data after Instance Termination

1	Launch an instance from an EBS-backed AMI and set the value of `DeleteOnTermination` flag to `false`. **Note** The instance launches with the EBS volume in its initial state.
2	Create snapshots of the volume at regular intervals.
3	Make changes to the data in the volume by doing some processing. **Note** This volume is now in a modified state.
4	Terminate the instance. The volume gets detached from the instance.

The volume, after instance termination, persists in its modified state. This is a new and a different volume from its initial state and is no longer associated with its original AMI.

When you launch an instance of a particular AMI and then terminate the instance, the detached volume will retain its original volume ID. If you launch another instance of the same AMI, the volume associated with the newly launched instance will have a different volume ID.

Using the Modified Volume (Volume B) after Instance Termination

1	Create a new EBS-backed AMI from the snapshot of the detached volume.
2	Launch an instance using the newly created AMI. **Note** This instance is now launched with the modified (Volume B) volume.

Or

Attach the modified volume to another instance.

Using the Initial Volume (Volume A) after Instance Terminates

Launch an instance from EBS-backed AMI and set the value of `DeleteOnTermination` flag to `false`. **Note** The instance launches with the EBS volume (Volume A) in its initial state with a new volume ID.

Data Availability

A single Amazon EBS volume is constrained to the single Availability Zone where it is created and becomes unavailable if the Availability Zone itself is unavailable. However, a snapshot of an Amazon EBS volume is available across all of the Availability Zones within a Region. You can use these snapshots to create volumes and make them available in more than one Availability Zone.

Making Your Data Available in Multiple Availability Zones

1	Create a snapshot of the volume that you want to make available in another Availability Zone.
2	Create a new volume using the snapshot created in the previous step, and specify a different Availability Zone.
3	Launch a new instance in that Availability Zone and attach the new volume.

API and Command Overview

The following table summarizes the available Amazon EBS commands and corresponding API actions for creating and using Amazon EBS volumes. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-create-volume` `CreateVolume`	Creates a new Amazon EBS volume using the specified size or creates a new volume based on a previously created snapshot.
`ec2-delete-volume` `DeleteVolume`	Deletes the specified volume. The action does not delete any snapshots that were created from the volume.
`ec2-describe-volumes` `DescribeVolumes`	Describes all your volumes, including size, source snapshot, Availability Zone, creation time, and status (`available` or `in-use`). If the volume is `in-use`, an attachment line shows the volume ID, the instance ID to which the volume is attached, the device name exposed to the instance, its status (attaching, attached, detaching, detached), and when it attached.
`ec2-attach-volume` `AttachVolume`	Attaches the specified volume to a specified instance, exposing the volume using the specified device name. A volume can be attached to only a single instance at any time. The volume and instance must be in the same Availability Zone. The instance must be in the `running` or `stopped` state.
`ec2-detach-volume` `DetachVolume`	Detaches the specified volume from the instance it's attached to. This action does not delete the volume. The volume can be attached to another instance and will have the same data as when it was detached.
`ec2-create-snapshot` `CreateSnapshot`	Creates a snapshot of the volume you specify. After the snapshot is created, you can use it to create volumes that contain exactly the same data as the original volume.

Command and API Action	Description
`ec2-delete-snapshot` `DeleteSnapshot`	Deletes the specified snapshot. This action does not affect currently running Amazon EBS volumes, regardless of whether they were used to create the snapshot or were derived from the snapshot.
`ec2-describe-snapshots` `DescribeSnapshots`	Describes the specified snapshot. Describes all snapshots, including their source volume, snapshot initiation time, progress (percentage complete), and status (`pending`, `completed`, etc.).
`ec2-modify-snapshot-attribute` `ModifySnapshotAttribute`	Modifies permissions for a snapshot (i.e., who can create volumes from the snapshot). You can specify one or more AWS accounts, or specify `all` to make the snapshot public.
`ec2-describe-snapshot-attribute` `DescribeSnapshotAttribute`	Describes permissions for a snapshot.

Creating an Amazon EBS Volume

Topics

To use Amazon EBS, you first create a volume that can be attached to any Amazon EC2 instance within the same Availability Zone. You can create an Amazon EBS volume with data from a snapshot stored in Amazon S3 or as a new blank volume with no data. You can also create and attach Amazon EBS volumes when you launch instances. For more information on volumes and snapshots, see Amazon Elastic Block Store (p. 126). The following procedure walks you through the process of creating an Amazon EBS volume from a snapshot.

AWS Management Console

To create an Amazon EBS volume

1. Log in to the AWS Management Console and open the **Amazon EC2** console.
2. Click **Volumes** in the **Navigation** pane.

   The console displays a list of current volumes.
3. Click **Create Volume**.

   The **Create Volume** dialog box appears.
4. Configure the following settings and click **Create**.

   - Size of the volume (in GiB)
   - Availability Zone in which to launch the instance
   - The ID of the snapshot from which you are launching the volume (optional)

Note

If you specify both a volume size and a snapshot ID, the size must be equal to or greater than the snapshot size.

Amazon EC2 begins creating the volume.

New volumes created from existing Amazon S3 snapshots load lazily in the background. This means that after a volume is created from a snapshot, there is no need to wait for all of the data to transfer from Amazon S3 to your Amazon EBS volume before your attached instance can start accessing the volume and all of its data. If your instance accesses data that hasn't yet been loaded, the volume will immediately download the requested data from Amazon S3, and then will continue loading the rest of the data in the background.

Accessing data for the first time from Amazon S3 might cause latency during the loading period. To avoid the possibility of an increased latency during the background loading of the data, you first access/read all performance-critical data from the volume (or read the entire volume) to ensure it has been loaded to the EBS volume from Amazon S3 before running the application.

Command Line Tools

To create an Amazon EBS volume

1. Enter the following command.

```
PROMPT>  ec2-create-volume --size 80 --availability-zone us-east-1a
```

   Amazon EC2 returns information about the volume that is similar to the following example.

```
VOLUME  vol-c7f95aae    80              us-east-1a      creating        2010-
03-30T13:54:37+0000
```

2. To check whether the volume is ready, use the following command.

```
PROMPT>  ec2-describe-volumes volume_id
```

   Amazon EC2 returns information about the volume that is similar to the following example.

```
VOLUME  vol-c7f95aae    80              us-east-1a      available       2010-
03-30T13:54:37+0000
```

API

To create an Amazon EBS volume

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateVolume
&Size=size
&AvailabilityZone=zone
&...auth parameters...
```

Following is an example response.

```
<CreateVolumeResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>06eabcdf-95b8-424f-87dc-da8e976f7858</requestId>
  <volumeId>vol-d11fbbb8</volumeId>
  <size>80</size>
  <snapshotId/>
  <availabilityZone>us-east-1a</availabilityZone>
  <status>creating</status>
  <createTime>2010-03-23T09:16:24.000Z</createTime>
</CreateVolumeResponse>
```

Attaching the Volume to an Instance

Topics

This section describes how to attach a volume that you created to an instance.

Note

The volume and instance must be in the same Availability Zone.

The following table lists the devices that are technically possible to connect to, the ones reserved for root device and instance storage, and the ones we recommend for connection. For more information about the instance store, see Amazon EC2 Instance Storage (p. 151). For information about the root device storage, see Root Device Storage (p. 160).

Instance Type	Technically Possible to Connect to	Reserved for Root and Instance Store	Recommended for Connection
Linux / UNIX	/dev/sd[a-z] /dev/sd[a-z][1-15] /dev/hd[a-z] /dev/hd[a-z][1-15]	/dev/sd[a] for root. /dev/sd[b-e] for instance stores.	/dev/sd[f-p] /dev/sd[f-p][1-6] If you're having trouble using these, see the important note after this table.
Windows	xvd[a-p] /dev/sda[1-2] /dev/sd[b-e]	/dev/sda[1-2] for root. xvd[a-e] for instance stores.	xvd[f-p]

After you have connected the device using the device names from the preceding table, the devices on your instance are mapped to either scsi drivers (sd[a-z]) or xvd drivers (xvd[a-p]), depending on the block device driver of your kernel. If the kernel has xvd drivers, then the devices will be mapped to xvd[a-p]. If the kernel does not have xvd drivers, then the devices will be mapped to sd[a-z][1-15].

Base Windows and CentOS HPC (Cluster Compute) images come bundled with xvd drivers, so the devices on Windows and CentOS HPC (Cluster Compute) instances are mapped to xvd[a-p]. The xvd drivers (xvd[a-p]) do not support the use of trailing numbers (xvd[a-p][1-15]). We recommend that you limit the use of the device names to xvd[a-p] on instances that have xvd drivers.

The device mapping does not affect the way you connect to the devices.

Depending on the size of your instance, Amazon EC2 provides instance stores that uses the drives sd[b-e] (on Linux/UNIX) or xvd[a-e] (on Windows and CentOS HPC). Although it is technically possible to connect your Amazon EBS volumes to devices reserved for instance store, we highly recommend that you do not do so. The behavior can be unpredictable.

Important

For Linux/UNIX instance types, we've received reports that some custom kernels might have restrictions that limit use to /dev/sd[f-p] or /dev/sd[f-p][1-6]. If you're having trouble using /dev/sd[q-z] or /dev/sd[q-z][1-6], try switching to /dev/sd[f-p] or /dev/sd[f-p][1-6].

Note

You are limited to total of 12 EBS volume attachments on a Windows instance.

AWS Management Console

To attach an Amazon EBS volume

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Volumes** in the **Navigation** pane.
   The console displays a list of current volumes.
3. Select a volume and click **Attach Volume**.
   The **Attach Volume** dialog box appears.
4. Select the instance to attach the volume to from the **Instance** list box (only instances in the same Availability Zone as the volume are displayed).
5. Select how the device is exposed to the instance from the **Device** list box.
6. Click **Attach**.
   The volume is attached to the instance.

Command Line Tools

To attach an Amazon EBS volume

- Enter the following command.

```
PROMPT>  ec2-attach-volume volume_id -i instance_id -d device
```

Amazon EC2 returns information that is similar to the following.

```
ATTACHMENT volume_id instance_id device attaching date_time
```

This example attaches volume `vol-4d826724` to `instance i-6058a509` in Linux and UNIX and exposes it as device `/dev/sdh`.

```
PROMPT>  ec2-attach-volume vol-4d826724 -i i-6058a509 -d /dev/sdh
```

```
ATTACHMENT vol-4d826724 i-6058a509 /dev/sdh attaching 2010-03-30T13:58:58+0000
```

This example attaches volume `vol-4d826724` to `instance i-6058a509` in Windows and exposes it as device `xvdf`.

```
PROMPT>  ec2-attach-volume vol-4d826724 -i i-6058a509 -d xvdf
```

```
ATTACHMENT vol-4d826724 i-6058a509 xvdf attaching 2010-03-30T13:58:58+0000
```

API

To attach an Amazon EBS volume

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=AttachVolume
&VolumeId=volume-id
&InstanceId=instance-id
&Device=device
&...auth parameters...
```

Following is an example response.

```
<AttachVolumeResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <volumeId>vol-4d826724</volumeId>
  <instanceId>i-6058a509</instanceId>
  <device>/dev/sdh</device>
  <status>attaching</status>
  <attachTime>2008-05-07T11:51:50.000Z</attachTime>
</AttachVolumeResponse>
```

Describing Volumes and Instances

Topics

You can list information about a volume, including the specific instance the volume is attached to.

AWS Management Console

To view information about an Amazon EBS volume

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Volumes** in the **Navigation** pane.
   The console displays a list of current volumes and the instances to which they are attached.
3. To view more information about a volume, select it.
   Information about the volume appears in the lower pane.

Command Line Tools

To describe volumes and list information about all volumes that you own

- Enter the following command.

```
PROMPT>  ec2-describe-volumes
```

Amazon EC2 returns information similar to the following.

```
VOLUME vol-4d826724 us-east-1a 80 in-use 2010-03-30T13:58:58+0000
ATTACHMENT vol-4d826724 i-6058a509 /dev/sdh attached  2010-03-30T13:54:55+0000
VOLUME vol-50957039 13 us-east-1a available  2010-03-24T08:01:44+0000
VOLUME vol-6682670f 1 us-east-1a in-use  2010-03-30T08:11:01+0000
ATTACHMENT vol-6682670f i-69a54000 /dev/sdh attached  2010-03-30T09:21:14+0000
```

This information includes the volume ID, capacity, status (in-use or available), and creation time of each volume. If the volume is attached, an attachment line shows the volume ID, the instance ID to which the volume is attached, the device name exposed to the instance, its status (attaching, attached, detaching, detached), and when it attached.

To describe instances and list volumes that are attached to running instances

- Enter the following command.

```
PROMPT>  ec2-describe-instances
```

Amazon EC2 returns information similar to the following.

```
RESERVATION     r-f25e6f9a      999988887777    default
INSTANCE        i-84b435de      ami-b232d0db    ec2-184-73-201-68.compute-
1.amazonaws.comdomU-12-31-39-00-86-35.compute-1.internal       running gsg-
keypair     0              m1.small 2010-03-30T08:43:48+0000        us-east-
1a      aki-94c527fd    ari-96c527ff    monitoring-disabled     184.73.201.68
   10.254.137.191                  ebs
BLOCKDEVICE     /dev/sda1       vol-cf13b3a6    2010-03-30T08:01:44.000Z
BLOCKDEVICE     /dev/sdh        vol-c7f95aae    2010-03-30T13:58:58.000Z
```

For more information about block device mapping, see Block Device Mapping (p. 167).

Tip

You can filter the results to return only the information about volumes and instances that match the criteria you specify. For more information about how to filter the results, go to ec2-describe-volumes and ec2-describe-instances in the *Amazon Elastic Compute Cloud Command Line Reference*.

API

To describe volumes and list information about all volumes that you own

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeVolumes
&...auth parameters...
```

Following is an example response.

```
<DescribeVolumesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
<volumeSet>
  <item>
    <volumeId>vol-4282672b</volumeId>
    <size>80</size>
    <status>in-use</status>
    <createTime>2008-05-07T11:51:50.000Z</createTime>
    <attachmentSet>
      <item>
        <volumeId>vol-4282672b</volumeId>
        <instanceId>i-6058a509</instanceId>
        <size>80</size>
        <snapshotId>snap-12345678</snapshotId>
        <availabilityZone>us-east-1a</availabilityZone>
        <status>attached</status>
        <attachTime>2008-05-07T12:51:50.000Z</attachTime>
      </item>
    </attachmentSet>
  </item>
  ...
</volumeSet>
```

Tip

You can filter the results to return only the information about volumes and instances that match the criteria you specify. For more information about how to filter the results, go to DescribeVolumes in the *Amazon Elastic Compute Cloud API Reference*.

Making an Amazon EBS Volume Available for Use

Topics

- Linux (p. 139)
- Windows (p. 139)

Inside the instance, the Amazon EBS volume is exposed as a normal block device and can be formatted as any file system and mounted.

After making the Amazon EBS volume available for use, you can take snapshots of it for backup purposes or to use as baselines to launch new volumes.

Linux

This section describes how to make a volume available to the Linux operating system.

Caution

This procedure assumes you want to mount an empty volume. If you're mounting a volume that already has data on it (e.g., a public dataset), don't use `mkfs` before mounting the volume. Otherwise you'll format the volume and delete the existing data.

To create an ext3 file system on the Amazon EBS volume and mount it as /mnt/data-store

1. Enter the following command.

```
$  yes | mkfs -t ext3 /dev/sdh
```

2. Enter the following command.

```
$  mkdir /mnt/data-store
```

3. Enter the following command.

```
$  mount /dev/sdh /mnt/data-store
```

Any data written to this file system is written to the Amazon EBS volume and is transparent to applications using the device.

Note

To enable the instance to reconnect to an Amazon EBS volume on reboot, add the device to the fstab or create a script that automatically mounts the volume on start-up.

Windows

This section describes how to make a volume available to the Windows operating system.

Caution

If you're mounting a volume that already has data on it (e.g., a public dataset), make sure you don't reformat the volume and delete the existing data.

To use an Amazon EBS volume

1. Log in to your instance using Remote Desktop.
2. On the taskbar, click **Start**, and then click **Run**.
3. Type **diskmgmt.msc** and click **OK**. The Disk Management utility opens.
4. Right-click the Amazon EBS volume, select **New Volume**, and follow the on-screen prompts.

Note

If the **New Volume** option does not appear, select **Format**.

Any data written to this file system is written to the Amazon EBS volume and is transparent to applications using the device.

Creating an Amazon EBS Snapshot

Topics

After writing data to an Amazon EBS volume, you can periodically create a snapshot of the volume to use as a baseline for new volumes or for data backup.

Snapshots occur asynchronously and the status of the snapshot is "pending" until the snapshot is complete.

By default, only you can launch volumes from your snapshots. However, you can choose to share your snapshots with specific AWS accounts or make them public. For more information, see Modifying Snapshot Permissions (p. 143).

Note

If you make periodic snapshots of a volume, the snapshots are incremental so that only the blocks on the device that have changed since your last snapshot are saved in the new snapshot. Even though snapshots are saved incrementally, the snapshot deletion process is designed so that you need to retain only the most recent snapshot in order to restore the volume.

AWS Management Console

To create a snapshot

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Snapshots** in the **Navigation** pane.
   The console displays a list of current snapshots.
3. Click **Create Snapshot**.
   The Create Snapshot dialog box appears.
4. Select the volume to create a snapshot for and click **Create**.
   Amazon EC2 begins creating the snapshot.

Command Line Tools

To create a snapshot

- Enter the following command.

```
PROMPT>  ec2-create-snapshot volume_id
```

Amazon EC2 returns information similar to the following example.

```
SNAPSHOT        snap-88fe11e0   vol-c7f95aae    pending 2010-03-
30T14:10:38+0000        999988887777     80
```

When the snapshot is complete, its status will change to `completed`.

API

To create a snapshot

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateSnapshot
&VolumeId=volume-id
&...auth parameters...
```

Following is an example response.

```
<CreateSnapshotResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>fe0d60a3-b804-484e-b558-92518bebe7af</requestId>
  <snapshotId>snap-05b4aa6c</snapshotId>
  <volumeId>vol-d11fbbb8</volumeId>
  <status>pending</status>
  <startTime>2010-03-23T09:43:51.000Z</startTime>
  <progress/>
  <ownerId>999988887777</ownerId>
  <volumeSize>20</volumeSize>
  <description/>
</CreateSnapshotResponse>
```

Describing Snapshots

Topics

This section describes how to view snapshots that you created.

AWS Management Console

To describe snapshots

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Snapshots** in the **Navigation** pane.

   The console displays a list of all snapshots to which you have access and their status.
3. To reduce the list, select an option from the **Viewing** list box. For example, to view only your snapshots, select **Owned by Me**.

   The console displays a new list of snapshots.
4. To view more information about a snapshot, select it.

   Information about the snapshot appears in the lower pane.

Command Line Tools

To describe snapshots

- Enter the following command.

```
PROMPT>  ec2-describe-snapshots snap-78a54011
```

Amazon EC2 returns information about the snapshot.

```
SNAPSHOT snap-78a54011 vol-4d826724 pending 2008-02-15T09:03:58+0000   60%
```

If the snapshot is in the process of being created, the status is `pending` and a percentage complete is displayed (e.g., 60%). Once the snapshot is complete, its status changes to `completed`.

Tip

If you have a large number of snapshots, you can filter the results to return information only about snapshots that match the criteria you specify. For more information about how to filter the results, go to ec2-describe-snapshots in the *Amazon Elastic Compute Cloud Command Line Reference*.

API

To describe snapshots

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeSnapshots
&SnapshotId=snapshot-id
&...auth parameters...
```

Following is an example response.

```
<DescribeSnapshotsResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
```

```
  <requestId>26245bae-2c70-4846-82d3-65784e298820</requestId>
  <snapshotSet>
      <item>
         <snapshotId>snap-05b4aa6c</snapshotId>
         <volumeId>vol-d11fbbb8</volumeId>
         <status>completed</status>
         <startTime>2010-03-23T09:43:52.000Z</startTime>
         <progress>100%</progress>
         <ownerId>999988887777</ownerId>
         <volumeSize>20</volumeSize>
         <description/>
      </item>
  </snapshotSet>
</DescribeSnapshotsResponse>
```

Tip

If you have a large number of snapshots, you can filter the list to return only certain types of interest to you. For more information about how to filter the results, go to DescribeSnapshots in the *Amazon Elastic Compute Cloud API Reference*.

Modifying Snapshot Permissions

Topics

This section describes how to modify permissions for your snapshots so that specific AWS accounts or all Amazon EC2 users can create volumes from them.

Important

When you share a snapshot (whether by sharing it with another AWS account or making it public to all), you are giving others access to all the data on your snapshot. Share snapshots only with people with whom you want to share *all* your snapshot data.

AWS Management Console

To modify snapshot permissions

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Snapshots** in the **Navigation** pane.
   The console displays a list of current snapshots and their status.
3. Select a snapshot and click **Permissions**.
   The **Modify Snapshot Permissions** dialog box appears.
4. Choose whether to make the snapshot public or to share it with select AWS accounts:

   - To make the snapshot public, select **Public** and click **Save**.

Important

This shares all the data on your snapshot with everyone.

- To only expose the snapshot to specific AWS accounts, select **Private**, enter the IDs of the accounts you want to share with, and click **Save**.

The console modifies permissions for the snapshot.

Command Line Tools

To modify snapshot permissions

1. Enter the following command to first describe the snapshot's permissions.

   ```
   PROMPT>  ec2-describe-snapshot-attribute snap_id --create-volume-permission
   ```

   If there are no permissions set on the snapshot, the output is empty.
2. Choose whether to make the snapshot public, or to share it with a specific AWS account:

   - To make the snapshot public, enter the following command.

     Important

     This command shares all the data on your snapshot with everyone.

     ```
     PROMPT>  ec2-modify-snapshot-attribute snap_id -c --add all
     ```

     Amazon EC2 returns permission information for the snapshot.

     ```
     createVolumePermission  snap_id   ADD     group   all
     ```

   - To share the snapshot with a particular AWS account, enter the following command.

     ```
     PROMPT>  ec2-modify-snapshot-attribute snap_id -c --add account_id
     ```

     Amazon EC2 returns permission information for the snapshot.

     ```
     createVolumePermission  snap_id   ADD     account_id
     ```

API

To modify snapshot permissions

1. Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeSnapshotAttribute
&SnapshotId=snapshot-id
&...auth parameters...
```

Following is an example response.

```
<DescribeSnapshotAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <requestId>d0d21738-e3da-4077-947d-c9e48472d831</requestId>
   <snapshotId>snap-05b4aa6c</snapshotId>
   <createVolumePermission/>
</DescribeSnapshotAttributeResponse>
```

2. Construct the following Query request to make the snapshot public.

Important

This shares all the data on your snapshot with everyone.

```
https://ec2.amazonaws.com/
?Action=ModifySnapshotAttribute
&SnapshotId.1=snapshot-id
&Add.1=all
&...auth parameters...
```

Following is an example response.

```
<ModifySnapshotAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <return>true</return>
</ModifySnapshotAttributeResponse>
```

Detaching an Amazon EBS Volume from an Instance

Topics

You can detach an Amazon EBS volume from an instance either by explicitly detaching the volume or terminating the instance. However, a volume must be unmounted inside the instance before being detached. Failure to do so will result in the volume being stuck in the busy state while it is trying to detach, which could possibly damage the file system or the data it contains.

This example explicitly unmounts the volume and then detaches it from the instance. This is useful when you want to terminate an instance or attach a volume to a different instance.

Note

If an Amazon EBS volume is the root device of an instance, it cannot be detached unless the instance is in the stopped state.

AWS Management Console

To detach an Amazon EBS volume

First unmount the volume from your instance drive.

1. For Linux/UNIX, enter the following command.

   ```
   # umount -d /dev/sdh
   ```

   For Windows, open **Disk Management**, right-click the volume to unmount, and select **Change Drive Letter and Path**. Then, select the mount point to remove and click **Remove**.
2. Log in to the AWS Management Console and click the **Amazon EC2** tab.
3. Click **Volumes** in the **Navigation** pane.
   The console displays a list of current volumes.
4. Select a volume and click **Detach Volume**.
   A confirmation dialog box appears.
5. Click **Yes, Detach**.
   The volume is detached from the instance.

Caution

If your volume stays in the *detaching* state, you can force the detachment using the **Force Detach** button. Forcing the detachment can lead to data loss or a corrupted file system. Use this option only as a last resort to detach a volume from a failed instance. The instance will not have an opportunity to flush file system caches or file system metadata. If you use this option, you must perform file system check and repair procedures.

Command Line Tools

To detach an Amazon EBS volume

- Enter the following command.

  ```
  # umount -d /dev/sdh
  PROMPT>  ec2-detach-volume vol-4d826724
  ```

  Amazon EC2 returns information similar to the following example.

  ```
  ATTACHMENT vol-4d826724 i-6058a509 /dev/sdh detaching 2010-03-30T13:58:58+0000
  ```

  Caution

  If your volume stays in the *detaching* state, you can force the detachment using the `--force` option. Forcing the detachment can lead to data loss or a corrupted file system. Use this option only as a last resort to detach a volume from a failed instance. The instance will not have an opportunity to flush file system caches or file system metadata. If you use this option, you must perform file system check and repair procedures. For more information about the `--force` option, go to ec2-detach-volume in the *Amazon Elastic Compute Cloud Command Line Reference*.

To detach an Amazon EBS volume by terminating the instance

- Enter the following command.

```
PROMPT>  ec2-terminate-instances i-6058a509
```

Amazon EC2 returns information similar to the following example.

```
INSTANCE    i-6058a509    running shutting-down
```

API

To detach an Amazon EBS volume

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DetachVolume
&VolumeId=volume-id
&InstanceId=instance-id
&...auth parameters...
```

Following is an example response.

```
<DetachVolumeResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <volumeId>vol-4d826724</volumeId>
  <instanceId>i-6058a509</instanceId>
  <device>/dev/sdh</device>
  <status>detaching</status>
  <attachTime>2008-05-08T11:51:50.000Z</attachTime>
</DetachVolumeResponse>
```

Caution

If your volume stays in the *detaching* state, you can force the detachment using the `Force` option. Forcing the detachment can lead to data loss or a corrupted file system. Use this option only as a last resort to detach a volume from a failed instance. The instance will not have an opportunity to flush file system caches or file system metadata. If you use this option, you must perform file system check and repair procedures. For more information about the `Force` option, go to DetachVolume in the *Amazon Elastic Compute Cloud API Reference*.

To verify the volume is no longer attached to the instance, see Describing Volumes and Instances (p. 136).

Deleting an Amazon EBS Snapshot

Topics

This section describes how to delete a snapshot.

Note

Be aware of the following:

- If you make periodic snapshots of a volume, the snapshots are incremental so that only the blocks on the device that have changed since your last snapshot are saved in the new snapshot. Even though snapshots are saved incrementally, the snapshot deletion process is designed so that you need to retain only the most recent snapshot in order to restore the volume.
- You cannot delete a snapshot of the root device of an EBS volume used by a registered AMI. You must first de-register the AMI before you can delete the snapshot.

AWS Management Console

To delete a snapshot

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Snapshots** in the **Navigation** pane.
   The console displays a list of current snapshots.
3. Select a snapshot and click **Delete Snapshot**.
   A confirmation dialog box appears.
4. Click **Yes, Delete**.
   The snapshot is deleted.

Command Line Tools

To delete a snapshot

- Enter the following command.

```
PROMPT>  ec2-delete-snapshot snapshot_id
```

Amazon EC2 returns information similar to the following example.

```
SNAPSHOT snap-78a54011
```

API

To delete a snapshot

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DeleteSnapshot
&SnapshotId=snapshot-id
&...auth parameters...
```

Following is an example response.

```
<DeleteSnapshotResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <return>true</return>
</DeleteSnapshotResponse>
```

Deleting an Amazon EBS Volume

Topics

After you no longer need a volume, you can delete it. After deletion, its data is gone and the volume can't be attached to any instance. However, before deletion, you can store a snapshot of the volume, which you can later use to recreate the volume.

This section describes how to delete a volume.

AWS Management Console

To delete a volume

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Volumes** in the **Navigation** pane.
   The console displays a list of current volumes.
3. Select a volume and click **Delete Volume**.
   A confirmation dialog box appears.
4. Click **Yes, Delete**.
   The volume is deleted.

Command Line Tools

To delete a volume

- Enter the following command.

```
PROMPT>  ec2-delete-volume vol-4282672b
```

Amazon EC2 returns information similar to the following example.

```
VOLUME vol-4282672b
```

API

To delete a volume

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DeleteVolume
&VolumeId=volume-id
&...auth parameters...
```

Following is an example response.

```
<DeleteVolumeResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <return>true</return>
</DeleteVolumeResponse>
```

Amazon Simple Storage Service

Amazon Simple Storage Service (Amazon S3) is a repository for Internet data. Amazon S3 provides access to reliable, fast, and inexpensive data storage infrastructure. It is designed to make web-scale computing easy by enabling you to store and retrieve any amount of data, at any time, from within Amazon EC2 or anywhere on the web. Amazon S3 stores data objects redundantly on multiple devices across multiple facilities and allows concurrent read or write access to these data objects by many separate clients or application threads. You can use the redundant data stored in Amazon S3 to recover quickly and reliably from instance or application failures.

Objects are the fundamental entities stored in Amazon S3. Objects consist of object data and metadata. The metadata is a set of name-value pairs that describes the object. The data portion is opaque to Amazon S3.

Every object stored in Amazon S3 is contained in a bucket. Buckets organize the Amazon S3 namespace at the highest level and identify the account responsible for that storage. Amazon S3 buckets are similar to Internet domain names. Objects stored in the buckets have a unique key value and are retrieved using a HTTP URL address. For example, if an object with a key value /photos/mygarden.jpg is stored in the johnsmith bucket, then it is addressable using the URL http://johnsmith.s3.amazonaws.com/photos/mygarden.jpg.

For more information on Amazon S3 go to the Amazon Simple Storage Service Developer Guide.

Amazon S3 Usage Scenario

The combination of Amazon S3 and Amazon EC2 provides resizable computing capacity with highly scalable and fast data storage infrastructure.

Amazon EC2 uses Amazon S3 for storing Amazon Machine Images (AMIs). You use AMIs for launching EC2 instances. In case of instance failure, you can use the stored AMI to immediately launch another instance, thereby allowing for fast recovery and business continuity.

Amazon EC2 also uses Amazon S3 to store snapshots (backup copies) of the data volumes. You can use snapshots for recovering data quickly and reliably in case of application or system failures. You can also use snapshots as a baseline to create multiple new data volumes, expand the size of an existing data volume, or move data volumes across multiple Availability Zones, thereby making your data usage highly scalable. For more information on using data volumes and snapshots, see Amazon Elastic Block Store (p. 126).

If you already have your data stored in Amazon S3 and want to move it to Amazon EBS volume, you can do so by mounting an empty Amazon EBS vloume to an Amazon EC2 instance and then using a data transfer application (e.g., FTP/SFTP, SCP) on your running instance.

Amazon S3 stores AMIs and snapshots redundantly on multiple devices across multiple facilities, thus providing an Amazon EC2 instance with highly durable storage infrastructure.

Amazon EC2 Instance Storage

Topics

When an instance is created from an Amazon Machine Image (AMI), in most cases it comes with a preconfigured block of pre-attached disk storage. Within this document, it is referred to as an *instance store*; it is also known as an *ephemeral store*. An instance store provides temporary block-level storage for Amazon EC2 instances. The data on the instance store volumes persists only during the life of the associated Amazon EC2 instance. The amount of this storage ranges from 160GiB to up to 3.3TiB and varies by Amazon EC2 instance type. Larger Amazon EC2 instances have more and larger instance store volumes. The instance store is dedicated to a particular instance, however the disk subsystem is shared among instances.

If you need a permanent storage solution, or if you need to improve storage latency or throughput, we recommend using Amazon EBS. For more information, see Amazon Elastic Block Store (p. 126).

An Amazon EC2 instance store consists of the virtual machine's root partition (forinstance store-backed AMI's only), plus one or more additional volumes that are dedicated to the Amazon EC2 instance (for both EBS-backed AMIs and instance store--backed AMIs). This storage is usable only from a single Amazon EC2 instance during its lifetime. Unlike EBS volumes, the instance store cannot be detached or attached to another instance. For more information on instance store--backed AMIs and Amazon EBS-backed AMIs, see AMI Basics (p. 15).

Important

If an instance reboots (intentionally or unintentionally), the data on the instance store will survive. However, under the following circumstances the data in the instance store will be lost:

- Failure of an underlying drive.

- Running an instance on degraded hardware.
- Stopping an Amazon EBS-backed instance.
- Terminating an instance.

The data on the instance store is not included when you bundle an AMI. For example, if you have an Amazon EC2 instance store-backed Windows instance, the D: drive on that instance is typically an instance store drive by default, and it's not included in an AMI that you bundle from that instance.

Instance Stores Available On Instance Types

Amazon EC2 instances are divided into different instance types depending on the number and size of the instance storage available on each one of them. The following table shows the instance types along with the number and the sizes of the instance store volumes available with each instance type.

Type	Name	Platform	Memory	Instance Storage
Small	m1.small	32-bit	1.7GiB	160GiB instance storage (150GiB plus 10GiB root partition)
Large	m1.large	64-bit	7.5GiB	850GiB instance storage (2 x 420GiB plus 10GiB root partition)
Extra Large	m1.xlarge	64-bit	15GiB	1690GiB instance storage (4 x 420GiB plus 10GiB root partition)
Micro	t1.micro	32-bit or 64-bit	613MiB	None (use Amazon EBS volumes for storage)
High-CPU Medium	c1.medium	32-bit	1.7GiB	350GiB instance storage (340GiB plus 10GiB root partition)
High CPU Extra Large	c1.xlarge	64-bit	7GiB	1690GiB instance storage (4 x 420GiB plus 10GiB root partition)
High-Memory Extra Large	m2.xlarge	64-bit	17.1GiB	420GiB instance storage (1 x 420GiB)
High-Memory Double Extra Large	m2.2xlarge	64-bit	34.2GiB	850GiB instance storage (1 x 840GiB plus 10GiB root partition)

Type	Name	Platform	Memory	Instance Storage
High-Memory Quadruple Extra Large	m2.4xlarge	64-bit	68.4GiB	1690GiB instance storage (2 x 840GiB plus 10GiB root partition)
Cluster Compute Quadruple Extra Large	cc1.4xlarge	64-bit	23GiB	1690GiB instance 64-bit storage (2 x 840GiB plus 10GiB root partition)
Cluster Compute Eight Extra Large	cc2.8xlarge	64-bit	60.5 GiB	3370GiB instance 64-bit storage (4 x 840 plus 10GiB root partition)
Cluster GPU Quadruple Extra Large	cg1.4xlarge	64-bit	22GiB (see note after this table)	1690GiB instance 64-bit storage (2 x 840GiB plus 10GiB root partition)

Note

The cg1.4xlarge instance type has 23GiB of memory, with 1GiB reserved for GPU operation. The 22GiB doesn't include the on-board memory of the GPUs, which is 3GiB per GPU for the NVIDIA Tesla M2050.

Note

When you launch an instance, you specify the instance type (the value in the Name column in the preceding table). We launch an m1.small instance if you don't specify a particular instance type.

Instance Store Device Names

Inside the instance, instance stores are exposed as normal block devices. All instance store volumes are pre-formatted with ext3 file system. However, you can reformat the volumes with any filesystem of your choice after you launch your instance. A Windows instance uses the built-in EC2Config Service tool to reformat all the instance store volumes available on the instance with NTFS file system.

Every AMI and instance has a mapping of the instance store attached to the instance. The mapping consists of the device name and the instance store it is attached to.

Note

Even though the instance storage mapping information exists on both Amazon EC2 instance store-backed and Amazon EBS-backed AMIs and instances, it is visible to you in the descriptions of Amazon EBS-backed AMIs and instances only. However, you can determine which instance stores are mapped from within the running instance itself.

Instance store volumes have to be mounted on the devices before they can be used. On a Linux instance, depending on the instance type, some instance stores are formatted and mounted by default by Amazon build images. A Windows instance uses a built-in tool, EC2Config Service, to format and mount all the instance stores attached to the instance.

The following table lists the devices reserved for instance store on each instance type and the state of the device (formatted, mounted, available). Instance store chunks are specified as ephemeral0, ephemeral1, etc. Basically, you just count up based on your instance type. A small image has an ephemeral0 (ext3, 15GiB) and an ephemeral1 (swap, 1GiB).

Device Name	Description
/dev/sda1	Formatted and mounted as root (/) on all Linux and UNIX instance types. Formatted and mounted as C:\ on all Windows instance types.
/dev/sda2 or xvdb (Windows)	Formatted and mounted as /mnt on m1.small and c1.medium instances. Formatted and mounted on small Windows instance types.
/dev/sda3	Formatted and mounted as /swap on m1.small and c1.medium instances on all Linux and UNIX instance types. Not available on Windows instances.
/dev/sdb or xvdb (Windows)	Formatted and mounted as /mnt on m1.large, m1.xlarge, c1.xlarge, cc1.4xlarge, cc2.8xlarge, m2.xlarge, m2.2xlarge, and m2.4xlarge Linux and UNIX instances. Formatted and mounted on m1.large, m1.xlarge, c1.xlarge, m2.xlarge, and m2.2xlarge Windows instances.
/dev/sdc or xvdc (Windows)	Available on m1.large, m1.xlarge, cc2.8xlarge, cc1.4xlarge, and c1.xlarge Linux and UNIX instances. Formatted and mounted on m1.large, m1.xlarge, c1.xlarge, and m2.4xlarge Windows instances.
/dev/sdd or xvdd (Windows)	Available on m1.xlarge and c1.xlarge, cc2.8xlarge Linux and UNIX instances. Formatted and mounted on m1.xlarge and c1.xlarge Windows instances.
/dev/sde or xvde (Windows)	Available on m1.xlarge and c1.xlarge Linux and UNIX instances. Formatted and mounted on m1.xlarge and c1.xlarge Windows instances.

Multiple instance store volumes can be mapped to the devices on any one instance. However, the number and size of these volumes must not exceed the number and size of the instance storage available for the instance type.

Amazon EC2 Instance Storage Usage Scenarios

Instance store volumes are ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers.

Making Instance Stores Available on Your Instances

Instances that use Amazon EBS for the root device do not, by default, have instance store available at boot time. If you want to use the instance stores with your EBS-backed instance, you have to explicitly map the instance stores with the devices while launching the instance. Examples of mapping the instance stores with the devices are, dev/sdb=ephemeral0, dev/sdc=ephemeral1, and so on. You need to use command line tools for mapping the instance stores with the devices. For more information on device mapping, see Block Device Mapping (p. 167)

The following task list describes what you need to do to launch an EBS-backed *m1.large* Windows instance by mapping the instance stores with the devices.

Accessing Instance Stores on Amazon EBS-backed Windows Instances

1	Locate an Amazon EBS-backed Windows AMI.
2	Using command lines tools, launch an m1.large instance and add the following instance store mapping: /dev/xvdb=ephemeral0 and /dev/xvdc=ephemeral1.
3	Connect to the instance.
4	On the **Start** menu, choose **Computer**.
5	Devices listed: • Local Disk C:/ 9.98GiB • Local Disk D:/ 419GiB • Local Disk E:/ 419GiB
6	Double-click Local Disk C:/. You will see the list of all the installed applications. This is your root drive.
7	Double-click Local Disk D:/ and then double-click Local Disk E:/. These drives are empty. They are the instance stores that come with your large instance, and they are available to use with your applications just like any physical drive.

The following task list describes what you need to do to launch an Amazon EBS-backed *m1.large* Linux instance by mapping the instance stores with the devices.

Accessing Instance Stores on Amazon EBS-backed Linux Instances

1	Locate an Amazon EBS-backed Linux/UNIX AMI.
2	Using command lines tools, launch an *m1.large* instance and add the following instance store mapping: /dev/sdb=ephemeral0 and /dev/sdc=ephemeral1.
3	Connect to the instance.
4	Check out the instance stores currently mounted on the disk.
5	Notice a 10GiB root partition mounted on the root and 420GiB mounted on an ephemeral0 device. Your *m1.large* instance comes with 2x420GiB instance storage. The other 420GiB is available but is unformatted and unmounted.
6	To format and mount the other 420GiB: a. Create a file system of your choice on the device /dev/sdc (requires root privileges). b. Create a directory on which to mount the device. c. Mount the device on the newly created directory.
7	Check to see if the device has been mounted.
8	Optionally, list the files on the root device.

Instance stores can also be mapped to the devices while the AMI image is being created. When you map instance stores to the devices (while creating the AMI image), subsequent instances launched from that AMI will all have instance stores by default. These instance stores are available only for that instance. You will have to use command line tools for mapping the instance stores with the devices. For information on adding instance stores while creating an AMI image, see the section called “Creating Your Own AMIs” (p. 17).

Note

You cannot map the instance stores with the devices after you've launched the instance.

No matter how you get your instance storage device on your instance, any data on those stores is deleted if the instance stops, fails, or terminates (except for data on the Amazon EBS volume being used as a root device).

Tip

Do not rely on your instance's instance storage for valuable, long-term data. Use a replication strategy across multiple instances to keep your data safe, or store your persistent data in Amazon S3, or use an Amazon EBS volume.

Instances that use Amazon S3 for the root device automatically have instance store available on them (for the root partition and other data you want to add). Instance stores on Amazon EC2 instance store-backed instances are visible and available from within a running instance.

The following task list describes what you need to do to access the instance stores from within an Amazon EC2 instance store-backed *m1.large* Windows instance.

Tasks for Accessing Instance Stores on Amazon EC2 instance store-backed Windows Instances

1	Locate an Amazon EC2 instance store-backed Windows AMI.
2	Launch an *m1.large* instance.
3	Connect to the instance.
4	On the **Start** menu, choose **Computer**.
5	Devices listed: • Local Disk C:/ 9.98GiB • Local Disk D:/ 419GiB • Local Disk E:/ 419GiB
6	Double-click Local Disk C:/. You see the list of all the installed applications. This is your root drive.
7	Double-click Local Disk D:/ and then double-click Local Disk E:/. These are empty. They are the instance stores that come with your m1.large instance, and they are available to use with your applications just like any physical drive.

Depending on the instance type, some instance stores on Amazon EC2 instance store-backed Linux and UNIX instances are not mounted. For example, on an m1.large Linux and UNIX instance, the device /dev/sdc, although formatted and available, has to be mounted before it can be used.

The following task list describes what you need to do to access the instance stores from within Amazon EC2 instance store-backed *m1.large* Linux instance.

Accessing Instance Stores on Amazon EC2 instance store-backed Linux Instances

1	Locate an Amazon EC2 instance store-backed Linux/Unix AMI.
2	Launch a *m1.large* instance.
3	Connect to the instance.
4	Check out the file systems currently mounted on the disk.
5	Notice 10GiB root partition mounted on the root and 420GiB mounted on an ephemeral0 device. Your m1.large instance comes with 2x420GiB instance storage. The other 420GiB is available but has to be mounted before it can be used.
6	To mount the other 420GiB: a. Create a directory on which to mount the device. b. Mount the device on the newly created directory.
7	Check to see if the device has been mounted.
8	Optionally, list the files on the root device.

Suppressing Instance Stores at Launch Time

You can use command line tools to prevent a particular instance storage device from attaching to the instance. You can do this for both Amazon EC2 instance store-backed and Amazon EBS-backed AMIs and instances. For example, mapping /dev/sdc=none when launching an instance will cause the device /dev/sdc to not attach to the instance. For more information on device mapping, see see Block Device Mapping (p. 167) and AMI Mapping Override (p. 169).

Using Amazon EC2 Instance Storage

This section describes how to add instance stores while creating an AMI image.

Adding A Default Instance Store

Because Amazon EBS-backed AMIs include a root device volume, they don't include any default instance store (i.e., ephemeral storage) the way Amazon EC2 instance store-backed AMIs do. However, you might want instances launched from your Amazon EBS-backed AMIs to include instance storage by default. This section describes how to create an AMI that includes the instance storage of your choice by default. You must do this with the command line tools or API; there's currently no way to do it through the AWS Management Console.

Command Line Tools

Use the `ec2-register` command and specify a block device mapping that includes the instance storage that you want for the image. For more information about block device mapping, see Block Device Mapping (p. 167).

To add instance storage by default

- Use the `ec2-register` command with the desired block device mapping information.

The following example registers an AMI with an 80GiB root device volume at /dev/sda1 created from the snap-12345678 snapshot. The root volume's `DeleteOnTermination` flag is set to `false`. The second block device mapping in the request maps /dev/sdc to the ephemeral0 instance store.

You can omit the size in the block device mapping if you want to create a volume the size of the snapshot. If you do specify a size, it must be equal to or larger than the snapshot. You can also resize the partition or create a new partition later.

Note

If you're using the command line tools on a Windows machine, you need to put quotation marks around any part of the command that includes an equal sign. For example: `... -b "/dev/sdc=ephemeral0" ...`.

```
PROMPT>  ec2-register -n My_Image_Name -d My_image_description --root-device-
name /dev/sda1  -b /dev/sda1=snap-12345678:80:false -b /dev/sdc=ephemeral0
```

In response, you get the ID for your new AMI.

```
IMAGE     ami-61a54008
```

Then, any instance of the AMI that you launch includes the instance storage you specified. You can view the block device mappings for the instance. Unfortunately, the information that's available doesn't include the instance store mappings. You can confirm that the instance store devices are available from within the instance itself. For more information, see Viewing Block Device Mappings (p. 170).

API

You can use `RegisterImage` and specify a block device mapping that includes the instance storage that you want for the image.

To add instance storage by default

- Issue the following Query request to register the image. The example registers an AMI with an 80 GiB root device volume at /dev/sda1 created from the snap-12345678 snapshot. The root volume's `deleteOnTermination` flag is set to `false`. The second block device mapping in the request maps /dev/sdc to the ephemeral0 instance store.

  Note

  You can omit the size in the block device mapping if you want to create a volume the size of the snapshot. If you do specify a size, it must be equal to or larger than the snapshot.

```
https://ec2.amazonaws.com/
?Action=RegisterImage
&Name=MyImage
&KernelId=aki-f70657b2
&RamdiskId=ari-ff0d5cba
&RootDeviceName=/dev/sda1
&BlockDeviceMapping.1.DeviceName=/dev/sda1
&BlockDeviceMapping.1.Ebs.SnapshotId=snap-12345678
&BlockDeviceMapping.1.Ebs.VolumeSize=80
```

```
&BlockDeviceMapping.1.Ebs.DeleteOnTermination=false
&BlockDeviceMapping.2.DeviceName=/dev/sdc
&BlockDeviceMapping.2.VirtualName=ephemeral0
&...auth parameters...
```

For information about the *auth* parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

Following is an example response.

```
<RegisterImageResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <imageId>ami-61a54008</imageId>
</RegisterImageResponse>
```

Any instance you then launch of the AMI includes the instance storage you specified. You can view the block device mappings for the instance. Unfortunately, the information that's available doesn't include the instance storage mappings. You can confirm the local storage devices are available from within the instance itself.For more information, see Viewing Block Device Mappings (p. 170)).

Disk Performance Optimization

Because of the way that Amazon EC2 virtualizes disks, the first write to any location on an instance's drives performs more slowly than subsequent writes. For most applications, amortizing this cost over the lifetime of the instance is acceptable. However, if you require high disk performance, we recommend that you initialize your drives by writing once to every drive location before production use. If you require further improvements in latency or throughput, we recommend using Amazon EBS.

To initialize the stores, use the following Linux/UNIX dd commands, depending on which store you want to initialize (/dev/sdb, etc.).

Note

Make sure to unmount the drive before performing this command.

Initialization can take a long time (about 8 hours for an extra large instance).

To initialize the stores, use the following commands on the m1.large, m1.xlarge, c1.xlarge, m2.xlarge, m2.2xlarge, and m2.4xlarge instance types:

```
dd if=/dev/zero of=/dev/sdb bs=1M
               dd if=/dev/zero of=/dev/sdc bs=1M
               dd if=/dev/zero of=/dev/sdd bs=1M
               dd if=/dev/zero of=/dev/sde bs=1M
```

For information about the stores that are available for each instance type, see Instance Stores Available On Instance Types (p. 152).

To perform initialization on all drives at the same time, use the following command:

```
dd if=/dev/zero bs=1M|tee /dev/sdb|tee /dev/sdc|tee /dev/sde > /dev/sdd
```

RAID Configuration

Configuring drives for RAID initializes them by writing to every drive location. When configuring software-based RAID, make sure to change the minimum reconstruction speed:

```
echo $((30*1024)) > /proc/sys/dev/raid/speed_limit_min
```

Amazon EC2 Storage Options Quick Reference

Features	Amazon S3	Amazon EBS	Amazon EC2 Instance Storage
Performance	Moderate (single thread) to Very High (multiple threads)	High	High
Durability of the data	High	Moderate	Low
Elasticity/scalability	Automatic	Manual (by adding more volumes)	No
Availability	High	Moderate to High (using EBS snapshots)	Low
Persistence across instantiations	Yes	Yes	No
Cross-instance access	Yes	No	No
Degree of redundancy	Highly redundant across multiple data centers	Redundant within an Availability Zone	Not redundant
Interfaces	HTTP, REST, or SOAP	Access through EC2 OS/file system	Block device map, access through EC2 OS/file system
Size limits	Effectivley unlimited (5TiB per object, unlimited objects per bucket)	1GiB to 1TiB per volume (can use multiple volumes or striping for larger capacities)	160GiB to 3.3TiB (larger instances have both larger volumes and more volumes)

Root Device Storage

Topics

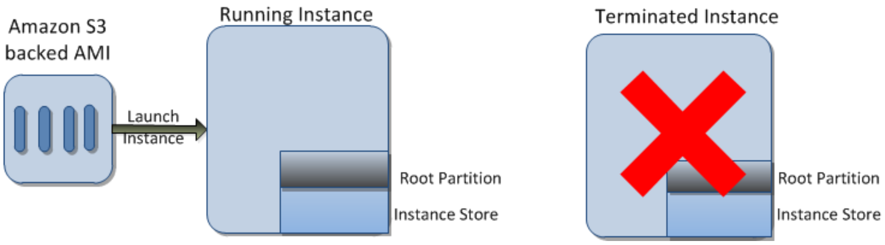
- Amazon EC2 Root Device Storage Concepts (p. 161)
- Amazon EC2 Root Device Storage Usage Scenarios (p. 163)
- Using Amazon EC2 Root Device Storage (p. 165)

When Amazon EC2 was first introduced, all AMIs were backed by Amazon EC2 instance store, which means the root device for an instance launched from the AMI is created from a template stored in Amazon S3. After we introduced Amazon EBS, we also introduced AMIs that are backed by Amazon EBS snapshots. This means that the root device for an instance launched from the AMI is on an Amazon EBS volume created from an Amazon EBS snapshot. You can choose between Amazon EC2 instance store and Amazon EBS as the root device for your AMI. We recommend using AMIs backed by EBS, because they launch faster and use persistent storage.

Amazon EC2 Root Device Storage Concepts

An Amazon EC2 instance can be launched from one of two types of AMIs: an Amazon EC2 instance store-backed AMI or an Amazon EBS-backed AMI. The description of an AMI includes type of AMI it is (you'll see the root device referred to in some places as either `ebs` (for Amazon EBS-backed) or `instance store` (for Amazon EC2 instance store-backed). This is important because there are significant differences between what you can do with each type of AMI. For a discussion of these differences, see Basics of Amazon EBS-Backed AMIs and Instances (p. 177).

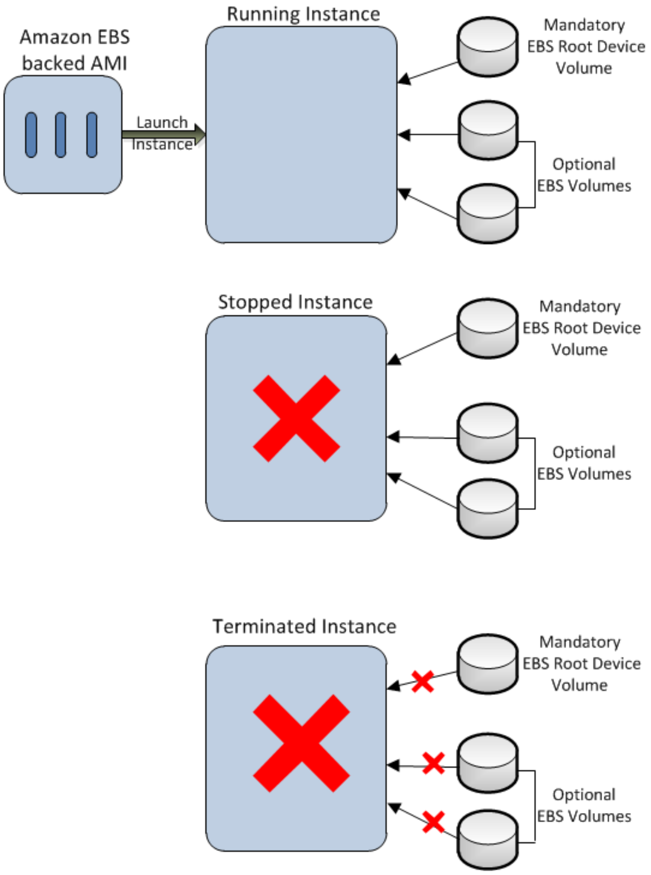
Instances that use Amazon S3 for the root device automatically have instance stores available on them with a separate root partition. When an instance is launched, an Amazon S3 image is copied to the root partition. This image is used to boot the instance. Any data on the instance stores persists as long as the instance is running and is deleted when the instance fails or terminates.



After an EC2 instance store-backed instance fails or terminates, it cannot be restored. If you plan to use Amazon EC2 instance store-backed instances, we highly recommend that you distribute the data on your instance stores across multiple Availibility Zones. You should also have a plan for backing up the data on your instance stores to Amazon S3 on a regular basis.

Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. When an Amazon EBS-backed instance is launched, an EBS volume is created for each EBS snapshot referenced by the AMI. You must have at least one snapshot that denotes the root device; the others are optional and denote additional volumes to be created from other snapshots.

An Amazon EBS-backed instance can be stopped and later restarted without affecting data stored in the attached volumes. There are various instance- and volume-related tasks you can do when an Amazon EBS-backed instance is in a stopped state. For example, you can modify the properties of the instance, you can change the size of your instance or update the kernel it is using, or you can attach your root volume to a different running instance for debugging or any other purpose.



By default, the root device volume and the other volumes created when an Amazon EBS-backed instance is launched are automatically deleted when the instance terminates (in other words, their `DeleteOnTermination` flags are set to `true` by default). You can change the default behavior by setting the `DeleteOnTermination` flag to the value you want when you launch the instance. For an example of how to change the flag at launch time, see Using Amazon EC2 Root Device Storage (p. 165).

However, any volumes that you attach *after* the instance is running (their `DeleteOnTermination` flags are set to `false` by default) are detached with their data intact on instance termination. These volumes can later be reattached to any running instance.

If an Amazon EBS-backed instance fails, you can restore your session by following one of these methods:

- You can stop and then start again.
- You can use `CreateImage` to automatically snapshot all the relevant volumes and create a new AMI. For more information, see Creating Amazon EBS-Backed AMIs (p. 18).
- You can attach the volume to the new instance by following these steps:
  1. Snapshot just the root volume.
  2. Register a new AMI using the snapshot.
  3. Launch a new instance from the new AMI.
  4. Detach the remaining volumes from the old instance.
  5. Reattach the volumes to the new instance.

We recommend using either the first or the second method for failed instances with normal volume size and the third method for failed instances with large volumes.

Amazon EC2 Root Device Storage Usage Scenarios

You can implement Amazon EBS backed AMIs by creating a set of snapshots and registering an AMI that uses those snapshots. The AMI publisher controls the default size of the root device through the size of the snapshot. The default size can be increased up to 1TiB to accommodate the requirements of the application either at the time you register the EBS-backed AMI or while you launch the EBS-backed instance.

Launching an Amazon EBS-backed Instance with Increased Root Device Storage Disk Size

1	Select an Amazon EBS-backed AMI to launch your instance from.
2	Check the root device size and note the AMI ID.
3	Using the command line interface, launch the instance by specifying the AMI ID and the mapping of the root device with the increased size.
4	Connect to the instance.
5	Check the size of the root device on the instance. **Note** The increased size of the root device is not apparent yet. This is because the file system does not recognize the increased size on the root device.
6	Resize the file system.
7	Check the size of the root device.
8	The root device of the newly launched instance now shows the increased size.

Note

You cannot decrease the size of your root device to less than the size of the AMI. To decrease the size of your root device, create your own AMI with the desired size for the root device and then launch an instance from that AMI.

Increasing the Size of the Root Device on an Amazon EBS-Backed Running Instance

1	Get the Amazon EBS volume ID and the Availability Zone of a running instance for which you want to increase the root storage size.
2	Stop the instance.
3	Detach the original volume from the instance.
4	Create a snapshot of the detached volume.
5	Create a new volume from the snapshot by specifying a larger size.

6	Attach the new volume to the stopped instance.
7	Start the instance and get the new IP address/hostname.
8	Connect to the instance using the new IP address/hostname.
9	Resize the root file system to the extent of the new Amazon EBS volume.
10	Check the size of the root device. The root device nows shows the increased size.
11	[Optional] Delete the old EBS volume, if you no longer need it.

The following list describes the tasks for creating a snapshot of the root device of an Amazon EC2 instance store-backed instance. The snapshot is created using an Amazon EBS volume. You can use this snapshot to create a new EBS-backed AMI or to launch another instance.

Creating a Snapshot of the Root Device of an Amazon EC2 instance store-backed Instance

1	Launch an instance from an Amazon EC2 instance store-backed AMI.
2	Create a 10GiB Amazon EBS volume in the same Availability Zone as that of your newly launched instance. **Note** You will use this volume to create a snapshot of the root partition of an Amazon EC2 instance store-backed AMI. The resulting snapshot will be the same size as the root partition; the maximum size of the root partition in an Amazon EC2 instance store-backed AMI is 10GiB.
3	Attach the volume to the running instance using either the AWS Management Console or the command line tools.
4	Format the volume with a file system.
5	[For Linux Users] Create a directory and then mount the volume on the newly created directory.
6	Copy the data on the root storage device to the newly attached EBS volume.
7	Unmount and detach the volume from the instance.
8	Create a snapshot of the volume.

Amazon EC2 instance store-backed AMIs are limited to 10GiB storage for the root device. If you require additional storage on your root device, you will have to first convert the Amazon EC2 instance store-backed AMI to an Amazon EBS-backed AMI and then launch an EBS-backed instance with increased root storage.

Converting an instance store-backed AMI to an Amazon EBS-backed AMI

1	Launch an instance from an Amazon EC2 instance store-backed AMI.

2	Create a 10GiB Amazon EBS volume in the same Availability Zone as that of your newly launched instance. **Note** You will use this volume to create a snapshot of the root partition of the Amazon EC2 instance store-backed AMI. The resulting snapshot will be the same size as the root partition; the maximum size of the root partition in an Amazon EC2 instance store-backed AMI is 10GiB.
3	Attach the volume to the running instance using either the AWS Management Console or the Command Line Tools.
4	Format the volume with a file system.
5	[For Linux Users] Create a directory and then mount the volume on the newly created directory.
6	Copy the data on the root storage device to the newly attached EBS volume.
7	Unmount and detach the volume from the instance.
8	Create a snapshot of the volume.
9	Register the snapshot of the volume as an AMI.

Using Amazon EC2 Root Device Storage

By default, the root device volume for an AMI backed by Amazon EBS is deleted when the instance terminates. This section describes how to change the default behavior of the root device volume.

Changing the Root Volume to Persist

Topics

- Command Line Tools (p. 165)
- API (p. 166)

In this section, we show you how to set the `DeleteOnTermination` flag to `false` in the instance's block device mapping at launch time. You can't currently do this in the AWS Management Console; you must use the command line tools or Amazon EC2 API. You can verify the change in the console, however.

Command Line Tools

To change the DeleteOnTermination flag at launch time

- In your `ec2-run-instances` request, include a block device mapping that sets the `deleteOnTermination` flag for the root device to `false`. Include the `-v` option to run the command in verbose mode.

```
PROMPT>  ec2-run-instances ami_id  -b root_device_name=::false other para
meters... -v
```

  The root device is typically /dev/sda1, or xvda (for Windows). Following is an example.

```
PROMPT>  ec2-run-instances ami-1a2b3c4d  -b /dev/sda1=::false other parameters... -v
```

If you're using the command line tools on a Windows system, you need to put quotation marks around the block device mapping value.

```
PROMPT>  ec2-run-instances ami-1a2b3c4d  -b "xvda=::false" other parameters... -v
```

By running the command in verbose mode, you can see the underlying SOAP request, and confirm that the `deleteOnTermination` value is set to `false`. The following XML snippet is from the SOAP request sent to Amazon EC2.

```
...
<blockDeviceMapping>
   <item>
     <deviceName>/dev/sda1</deviceName>
     <ebs>
       <deleteOnTermination>false</deleteOnTermination>
     </ebs>
   </item>
</blockDeviceMapping>
...
```

You can also verify the setting after the instance launches by viewing the instance's details in the AWS Management Console.

API

To change the DeleteOnTermination flag at launch time

- Issue the following Query request to include a block device mapping that sets the `deleteOnTermination` flag for the root device to `false`. If it's a Windows AMI, use xvda instead of /dev/sda1 as the root device name.

```
https://ec2.amazonaws.com/
?Action=RunInstances
&ImageId=ami-1a2b3c4d
&BlockDeviceMapping.1.DeviceName=/dev/sda1
&BlockDeviceMapping.1.Ebs.DeleteOnTermination=false
&...auth parameters...
```

You can confirm the setting by using `DescribeInstances`. The following example snippet from the XML response shows that the flag is set to `false` for the instance's root device.

```
...
 <rootDeviceName>/dev/sda1</rootDeviceName>
 <blockDeviceMapping>
   <item>
     <deviceName>/dev/sda1</deviceName>
     <ebs>
```

```
       <volumeId>vol-818843e8</volumeId>
       <status>attached</status>
       <attachTime>2010-02-22T20:36:18.000Z</attachTime>
       <deleteOnTermination>false</deleteOnTermination>
     </ebs>
   </item>
 </blockDeviceMapping>
...
```

You can also verify the setting by viewing the instance's details in the AWS Management Console. For more information, see Viewing Block Device Mappings (p. 170).

Block Device Mapping

Topics

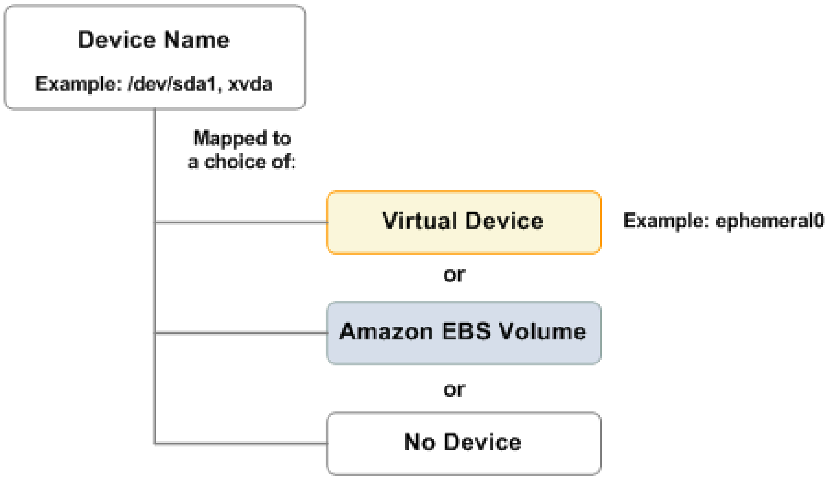
Every AMI and instance has a *block device mapping* structure that specifies the block devices to be attached to the instance. For example, AWS provides some simple Amazon EBS-backed AMIs that you can use to get started with Amazon EC2. By default, when you launch an instance of one of these AMIs, there's a volume attached for the root device, and that's all. There's no instance store attached by default, and no additional Amazon EBS volumes. The block device mapping for that AMI or instance contains information about the one block device that's mapped: the root device volume. That information is included as part of the AMI details displayed in the AWS Management Console. You can also get the information by describing the AMI with the command line tools or API.

Amazon EC2 instance store-backed AMIs also have a block device mapping structure. These AMIs have instance storage available by default; however, there's no information about this storage included in the AMI's block device mapping. Also, you can't set up an Amazon EC2 instance store-backed AMI to automatically attach a volume at launch time (something you can do with Amazon EBS-backed AMIs). Therefore, the block device mapping structure for Amazon EC2 instance store-backed AMIs is effectively empty.

This section describes the parts of a block device mapping and what you can do with an AMI's or instance's block device mapping.

Parts of a Block Device Mapping

A block device mapping structure contains one or more items, and each item describes the mapping for a single block device. As shown in the following figure, each item specifies the device name, and whether the device is mapped to a virtual device (i.e., an instance store), an Amazon EBS volume, or to nothing. You need to use the last option only when you want to suppress a block device from attaching at launch time (for an example, see AMI Mapping Override (p. 169)).



For a device mapped to a virtual device, the block device mapping specifies the name of the store (e.g., ephemeral0).

Note

Even though the virtual device part of the block device mapping exists in the Amazon EC2 system, it isn't included in the block device mapping information that is visible to you in the AWS Management Console or when you describe the AMI. However, you can determine which instance storage devices are mapped from within the running instance itself.

For a device mapped to an Amazon EBS volume (which is available only for Amazon EBS-backed AMIs), the block device mapping specifies the following items:

- Snapshot ID to use for the volume
- Volume size (in GiB)
- `DeleteOnTermination` flag (whether to delete the volume on instance termination)

At a minimum, either a snapshot ID or a volume size is required. If there's no snapshot ID, then an empty volume is created in the size specified. If both a snapshot ID and a size are specified, then a volume of the specified snapshot is created in the specified size. The specified size must be equal to or larger than the size of the snapshot.

The default value for the `DeleteOnTermination` flag is `true` for any volumes that are created when the instance is launched. If you attach any volumes to a running instance, new items are added to the block device mapping for those volumes. The `DeleteOnTermination` flag is `false` by default for any volumes you attach *after* the instance is running. You can change the `DeleteOnTermination` value for any of the volumes in the block device mapping. For an example of how to do this with the root device, see Using Amazon EC2 Root Device Storage (p. 165).

You can view an AMI's block device mapping information in the AWS Management Console or by describing the AMI with the command line tools or API. The available information includes mappings only for volumes; no information about instance store devices is included. For more information, see Viewing Block Device Mappings (p. 170).

Let's say you want to take one of the Amazon EBS-backed AMIs that AWS provides to help you get started with Amazon EC2 and create your own AMI from it with instance storage devices or additional Amazon EBS volumes that automatically attach at launch time. You can do that by specifying a block device mapping with your additions when you register the AMI. For more information, see Adding A Default Instance Store (p. 157) and Automatically Attaching Volumes (p. 187).

Note

When you launch an instance of a particular AMI and then terminate the instance, any remaining volumes are not automatically related to any future instances of that AMI. In other words, a new instance that you launch of that same AMI doesn't attempt to attach to the remaining volume. However, you can manually attach the remaining volume to the new instance if you want.

Instance Block Device Mapping

Just as every AMI has a block device mapping structure that specifies the block devices that are attached to it, so does every instance. You can view the block device mapping for an instance in the AWS Management Console or by describing the instance with the command line tools or API. The available information includes information only for volumes; no information about instance store devices is currently available.

Note

For instances launched before the release of the 2009-10-31 API version, no block device mapping information for any attached volumes is available when you describe the instance. You must detach and reattach the volumes for the information to be available when you describe the instance.

The following information is included for each Amazon EBS volume attached to the instance:

- The device name (e.g., /dev/sda1 or xvda)
- The volume ID (e.g., vol-12345678)
- The status of the attachment (e.g., attached)
- The time when the attachment was initiated
- The `DeleteOnTermination` value (`true` or `false`)

For more information about viewing this information, see Viewing Block Device Mappings (p. 170).

AMI Mapping Override

You can override the block device mapping for an instance that you plan to launch. This will not make a permanent change to the mapping of the AMI by specifying a block device mapping structure when you run the instance. Any differences between the AMI's block device mapping and the one you specify for the instance overwrite or merge with the AMI's original mapping. You can use the override capability with any storage attachment the AMI has, including the root device volume. Exception: You can't change the size of the root device volume with a block device mapping override.

Note

For Amazon EC2 instance store-backed AMIs, you can't specify a block device mapping that includes Amazon EBS volumes to attach at launch time. You can only modify the instance storage (e.g., add or remove an instance storage device).

For example, let's say you have an Amazon EBS-backed AMI that has a block device mapping that includes two additional volumes (beyond the root device volume) called *ExtraSnapshot1* and *ExtraSnapshot2*. You set both of these volumes to and you've set both of persist when the instance terminates. You want to launch a single instance that:

- Includes a larger version of ExtraSnapshot1 that is set to delete on instance termination

- Doesn't include ExtraSnapshot2
- Includes instance storage device

At launch time, you specify a block device mapping that includesthe changes that you want. When the instance launches, its block device mapping includes the original AMI settings that were unaffected by your overrides, plus the new overrides you specified.

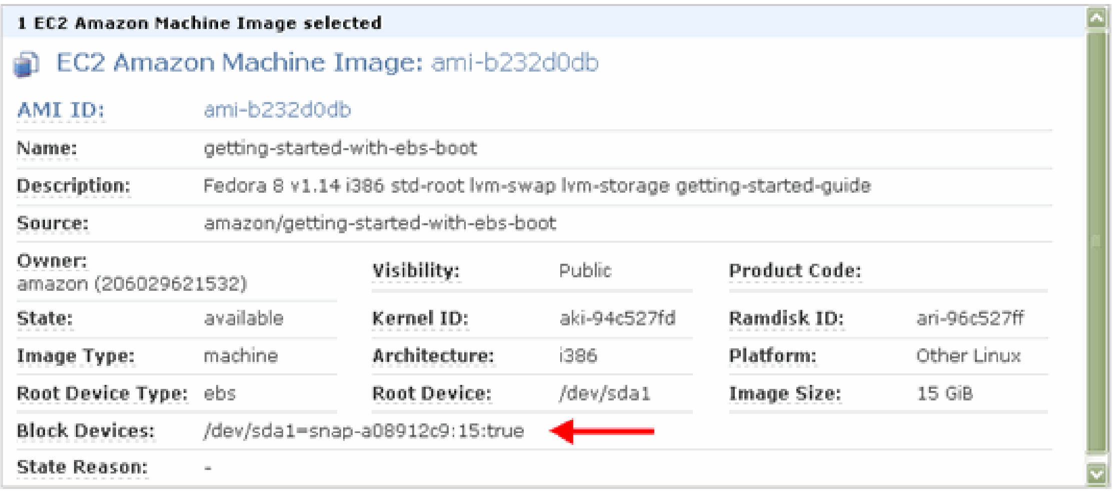
For an example of how to override the AMI's block device mapping at launch time, see Overriding the AMI's Block Device Mapping (p. 173).

Viewing Block Device Mappings

You can get a list of the block devices mapped to an AMI or an instance. The list includes only the Amazon EBS volumes that are mapped to the AMI or instance; no information about instance storage is provided. For general information about block device mapping, see Block Device Mapping (p. 167).

AWS Management Console

The devices mapped to an AMI are listed as part of the AMI details displayed in the console. Only the Amazon EBS snapshots mapped to the AMI are listed; no information is displayed for instance storage devices that are mapped. The following screenshot shows the details of a sample AMI.



1 EC2 Amazon Machine Image selected

EC2 Amazon Machine Image: ami-b232d0db

AMI ID: ami-b232d0db

Name: getting-started-with-ebs-boot

Description: Fedora 8 v1.14 i386 std-root lvm-swap lvm-storage getting-started-guide

Source: amazon/getting-started-with-ebs-boot

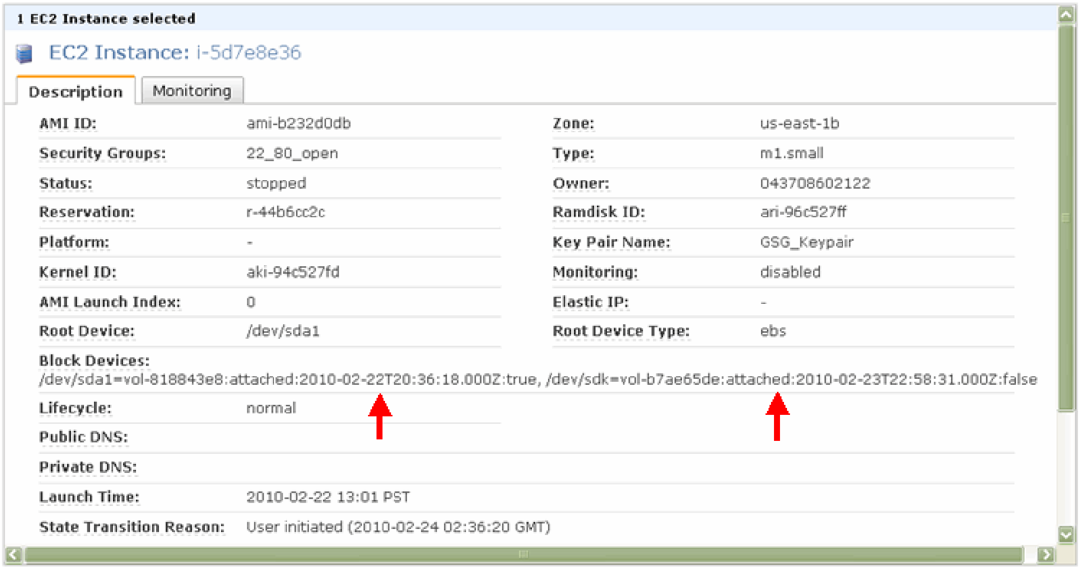
Owner: amazon (206029621532)		Visibility:	Public	Product Code:	
State:	available	Kernel ID:	aki-94c527fd	Ramdisk ID:	ari-96c527ff
Image Type:	machine	Architecture:	i386	Platform:	Other Linux
Root Device Type:	ebs	Root Device:	/dev/sda1	Image Size:	15 GiB

Block Devices: /dev/sda1=snap-a08912c9:15:true

State Reason: -

The **Block Devices** field shows information about the root device's mapping: */dev/sda1=snap-a08912c9:15:true*. This means the root device /dev/sda1 is mapped to a 15GiB volume created from the snap-a08912c9 snapshot, and the volume's `DeleteOnTermination` flag is `true`.

The devices mapped to an instance are listed as part of the instance details displayed in the console. Only the Amazon EBS volumes mapped to the instance are listed; no information is displayed for instance store devices that are mapped. The following screenshot shows the details of a sample instance.



1 EC2 Instance selected

EC2 Instance: i-5d7e8e36

AMI ID:	ami-b232d0db	Zone:	us-east-1b
Security Groups:	22_80_open	Type:	m1.small
Status:	stopped	Owner:	043708602122
Reservation:	r-44b6cc2c	Ramdisk ID:	ari-96c527ff
Platform:	-	Key Pair Name:	GSG_Keypair
Kernel ID:	aki-94c527fd	Monitoring:	disabled
AMI Launch Index:	0	Elastic IP:	-
Root Device:	/dev/sda1	Root Device Type:	ebs

Block Devices:
/dev/sda1=vol-818843e8:attached:2010-02-22T20:36:18.000Z:true, /dev/sdk=vol-b7ae65de:attached:2010-02-23T22:58:31.000Z:false

Lifecycle:	normal
Public DNS:	
Private DNS:	
Launch Time:	2010-02-22 13:01 PST
State Transition Reason:	User initiated (2010-02-24 02:36:20 GMT)

The **Block Devices** field shows information about two different devices. For each device, the following items are displayed:

- The device name (e.g., /dev/sda1)
- The volume ID (e.g., vol-12345678)
- The status of the attachment (e.g., attached)
- The time when the attachment was initiated
- The `DeleteOnTermination` value (`true` or `false`)

For the example shown in the preceding screenshot, the root device is /dev/sda1 and it's mapped to vol-818843e8. The volume was attached on February 22 at 10:36 UTC and is currently attached. The root device volume is set to be deleted when the instance terminates.

The second mapped device is /dev/sdk, and it's mapped to vol-b7ae65de. The volume was attached on February 23 at 22:58 UTC and is currently attached. This volume will not be deleted when the instance terminates.

Command Line Tools

You can get information about an AMI's or instance's block device mapping through the command line tools.

To view the block device mapping for an AMI

- Use the `ec2-describe-images` command.

  ```
  PROMPT>  ec2-describe-images ami_id
  ```

  Following is an example of the part of the response that includes the block device mapping.

  ```
  BLOCKDEVICEMAPPING      /dev/sda1               snap-a08912c9   15
  ```

This response shows device /dev/sda1 is mapped to a 15GiB volume created from the snap-a08912c9 snapshot. The response doesn't include the value of the `DeleteOnTermination` flag. However, if you run the command in verbose mode (i.e., with the `-v` switch), you can view the underlying XML response, which includes the flag's value.

To view the block device mapping for an instance

- Use the `ec2-describe-instances` command.

```
PROMPT>  ec2-describe-instances instance_id
```

  Following is an example of the part of the response that includes the block device mapping.

```
BLOCKDEVICE     /dev/sda1       vol-818843e8    2010-02-22T20:36:18.000Z
BLOCKDEVICE     /dev/sdk        vol-b7ae65de    2010-02-23T22:58:31.000Z
```

The response shows the ID of the volume mapped to the root device /dev/sda1. The response currently doesn't include the status of the volume's attachment or the value of the `DeleteOnTermination` flag. However, if you run the command in verbose mode (i.e., with the `-v` switch), you can view the underlying XML response, which includes that information.

The second mapped device is /dev/sdk, and it's mapped to vol-b7ae65de. The volume was attached on February 23 at 22:58 UTC.

API

You can get information about an AMI's or instance's block device mapping through the API.

To view the block device mapping for an AMI

1. Issue the following Query request to get a description of the AMI's block device mapping.

```
https://ec2.amazonaws.com/
?Action=DescribeImages
&...auth parameters...
```

   For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

2. In the output XML, locate the `rootDeviceName` and `blockDeviceMapping` elements for the instance (they're right next to each other in the response). Following is an example of that part of the response.

```
...
<rootDeviceName>/dev/sda1</rootDeviceName>
<blockDeviceMapping>
   <item>
     <deviceName>/dev/sda1</deviceName>
     <ebs>
       <snapshotId>snap-a08912c9</snapshotId>
       <volumeSize>15</volumeSize>
       <deleteOnTermination>true</deleteOnTermination>
     </ebs>
   </item>
```

```
</blockDeviceMapping>
...
```

This example response shows that the root device is /dev/sda1, and it's mapped to a 15GiB volume created from the snap-a08912c9 snapshot. The volume is set to delete when the instance terminates.

To view the block device mapping for an instance

1. Issue the following Query request to get a description of the instance's block device mapping.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstances
   &...auth parameters...
   ```

2. In the output XML, locate the `rootDeviceName` and `blockDeviceMapping` elements for the instance (they're right next to each other in the response). Following is an example of that part of the response.

   ```
   <rootDeviceName>/dev/sda1</rootDeviceName>
   <blockDeviceMapping>
     <item>
       <deviceName>/dev/sda1</deviceName>
       <ebs>
         <volumeId>vol-818843e8</volumeId>
         <status>attached</status>
         <attachTime>2010-02-22T20:36:18.000Z</attachTime>
         <deleteOnTermination>true</deleteOnTermination>
       </ebs>
     </item>
     <item>
       <deviceName>/dev/sdk</deviceName>
       <ebs>
         <volumeId>vol-b7ae65de</volumeId>
         <status>attached</status>
         <attachTime>2010-02-23T22:58:31.000Z</attachTime>
         <deleteOnTermination>false</deleteOnTermination>
       </ebs>
     </item>
   </blockDeviceMapping>
   ```

   The response shows the root device is /dev/sda1, and it's mapped to vol-818843e8. The volume was attached on February 22 at 20:36 UTC and is currently attached. The root device volume is set to be deleted when the instance terminates.

   The second mapped device is /dev/sdk, and it's mapped to vol-b7ae65de. The volume was attached on on February 23 at 22:58 UTC and is currently attached. This volume will not be deleted when the instance terminates.

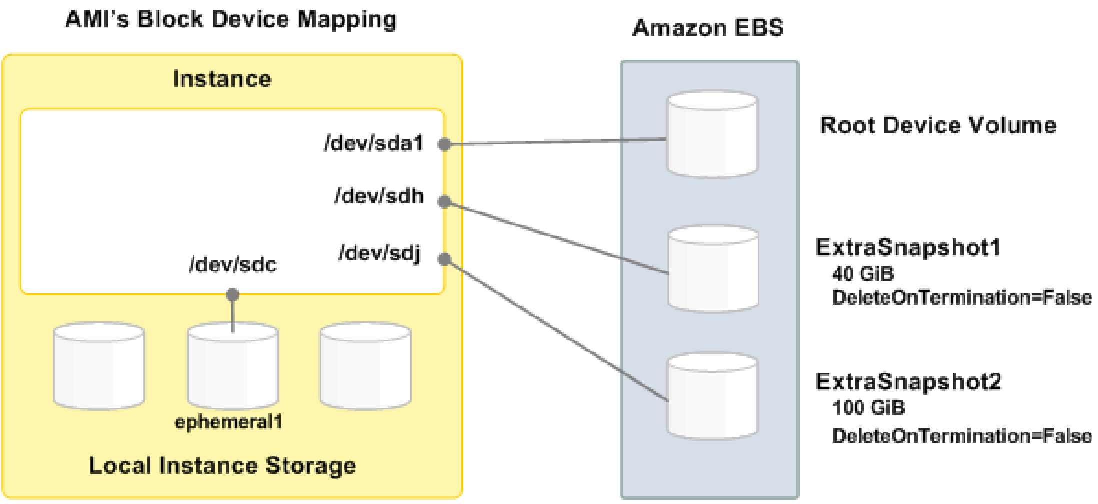
Overriding the AMI's Block Device Mapping

Any instance you launch automatically includes any storage devices in the AMI's block device mapping. You can override that mapping by specifying a block device mapping with the changes that you want at launch time. In your overrides, you can modify the mapping for any of the storage devices except the root device volume.

Note

If you want an Amazon EBS-backed instance to have an instance store (i.e., ephemeral storage), you must add it at launch time (using either the procedure described in this section, or the procedure described in Adding A Default Instance Store (p. 157)). You can't add instance storage to an instance after you've launched it.

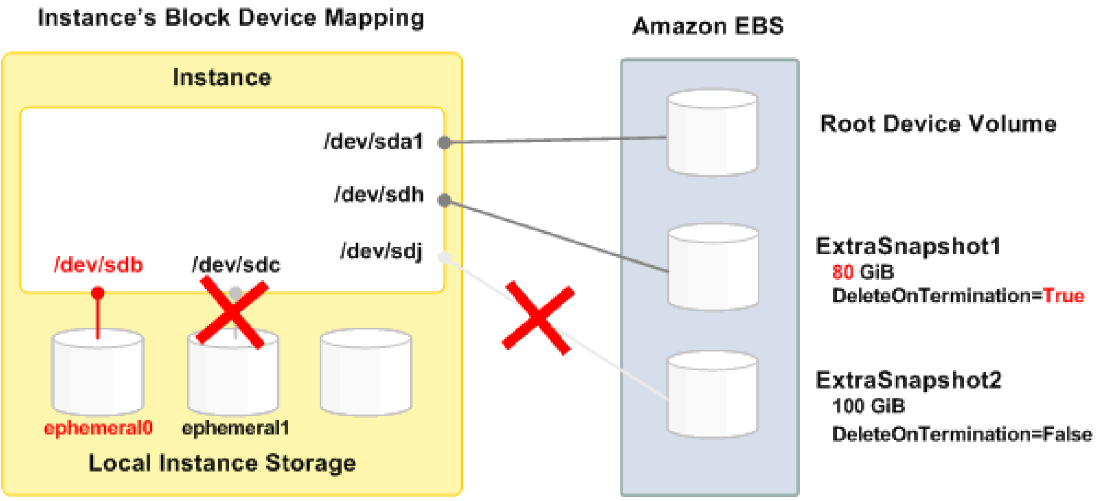
For example, let's say you have an Amazon EBS-backed AMI that has a block device mapping that maps /dev/sdc to ephemeral1, and has two additional Amazon EBS volumes (beyond the root device volume). We'll call the snapshots for these extra volumes *ExtraSnapshot1* and *ExtraSnapshot2*. Assume ExtraSnapshot1 is mapped to /dev/sdh and ExtraSnapshot2 is mapped to /dev/sdj. You've set the `DeleteOnTermination` flag to `false` for both volumes. The following figure illustrates this block device mapping for the AMI.



Let's say you want to launch a single instance that:

- Includes an 80GiB version of ExtraSnapshot1 with `DeleteOnTermination=true`
- Doesn't include ExtraSnapshot2
- Includes the instance storage device ephemeral0 at /dev/sdb
- Doesn't include the instance storage device ephemeral1 at /dev/sdc

The following figure includes the changes you want to make when you launch the instance.



The examples in the following section show how to make these changes when you launch the instance.

Note

Although you can use the AWS Management Console to launch instances, you can't specify a block device mapping as part of the launch request. That functionality is available only through the command line tools or API.

Command Line Tools

Use `ec2-run-instances` and specify a block device mapping that includes the changes you want to make.

To override the AMI's block device mapping

- Use the `ec2-run-instances` command with your changes to the block device mapping. For each storage device you want to modify, include a separate `-b` switch.

```
PROMPT>  ec2-run-instances ami-12345678 -b /dev/sdh=:80:true -b /dev/sdj=none
 -b /dev/sdb=ephemeral0 -b /dev/sdc=none
```

  The first `-b` switch in the example increases the size of the volume created from ExtraSnapshot1 to 80GiB and sets `DeleteOnTermination` to `true`. Notice that you don't need to specify the ID for ExtraSnapshot1. Specifying the mapped storage device name (/dev/sdh) is enough.

  The second `-b` switch prevents a volume of ExtraSnapshot2 from attaching to the instance at /dev/sdj at launch time. Again, you don't need to specify the ID for ExtraSnapshot2.

  The third `-b` switch maps local storage device /dev/sdb to ephemeral0.

  The fourth `-b` switch prevents local storage device /dev/sdc from mapping to ephemeral1.

  The instance launches with your changes.

You can view an AMI's or instance's block device mapping at any time. For more information, see Viewing Block Device Mappings (p. 170).

API

You use `RunInstances` and specify a block device mapping that includes the changes you want to make.

To override the AMI's block device mapping

- Issue the following Query request to run an instance with your desired block device mapping overrides.

  - The first `item` element in the block device mapping increases the size of the volume created from ExtraSnapshot1 to 80GiB and sets `deleteOnTermination` to `true`. Notice that you don't need to specify the ID for ExtraSnapshot1. Specifying the mapped storage device name (/dev/sdh) is enough.
  - The second `item` element prevents a volume of ExtraSnapshot2 from attaching to the instance at launch time. Again, you don't need to specify the ID for ExtraSnapshot2.
  - The third `item` element maps instance storage device /dev/sdb to ephemeral0.
  - The fourth `item` element prevents instance storage device /dev/sdc from mapping to ephemeral1.

  ```
  https://ec2.amazonaws.com/
  ?Action=RunInstances
  &ImageId=ami-cb4aa6a2
  ...
  &BlockDeviceMapping.1.DeviceName=/dev/sdh
  &BlockDeviceMapping.1.Ebs.VolumeSize=80
  &BlockDeviceMapping.1.Ebs.DeleteOnTermination=true
  &BlockDeviceMapping.2.DeviceName=/dev/sdj
  &BlockDeviceMapping.2.Ebs.NoDevice=true
  &BlockDeviceMapping.3.DeviceName=/dev/sdb
  &BlockDeviceMapping.3.VirtualName=ephemeral0
  &BlockDeviceMapping.4.DeviceName=/dev/sdc
  &BlockDeviceMapping.4.Ebs.NoDevice=true
  ...
  &...auth parameters...
  ```

  The instance launches with your changes.

You can view an AMI's or instance's block device mapping at any time. For more information, see Viewing Block Device Mappings (p. 170).

Using Amazon EBS-Backed AMIs and Instances

Topics

This section describes Amazon EBS-backed AMIs and how they work. It also gives procedures for common tasks you'll want to perform with AMIs backed by Amazon EBS.

Basics of Amazon EBS-Backed AMIs and Instances

Topics

Before the 2009-10-31 API, every AMI had its root device stored in Amazon S3. Instances that are launched from these AMIs have instance stores available on them with a separate root partition. When an instance is launched, an image of the root device stored in Amazon S3 is copied to the root partition. This image is used to boot the instance. We refer to these AMIs as *Amazon EC2 instance store-backed*, or *AMIs with the root device in Amazon S3*. Starting with the 2009-10-31 API, we have a new type of AMI that stores its root device as an Amazon EBS volume. We refer to these AMIs as *Amazon EBS-backed*. When an instance of this type of AMI launches, we create an Amazon EBS volume from the associated snapshot, and that volume becomes the root device.

Summary of Differences

This section describes the general differences between Amazon EBS-backed AMIs and Amazon EC2 instance store-backed AMIs.

Size Limit

Amazon EC2 instance store-backed AMIs are limited to 10 GiB storage for the root device, whereas Amazon EBS-backed AMIs are limited to 1 TiB. Many Windows AMIs come close to the 10 GiB limit, so you'll find that Windows AMIs are often backed by an Amazon EBS volume.

Note

All Windows Server 2008 and Windows Server 2008 R2 AMIs are backed by an Amazon EBS volume by default because of their larger size.

Stopped State

You can *stop* an Amazon EBS-backed instance, but not an Amazon EC2 instance store-backed instance. Stopping causes the instance to stop running (its status goes from *running* to *stopping* to *stopped*). A stopped instance persists in Amazon EBS, which allows it to be restarted. Stopping is different from terminating; you can't restart a terminated instance. Because Amazon EC2 instance store-backed AMIs can't be stopped, they're either running or terminated. For more information about what happens and what you can do while an instance is stopped, see Stop/Start (p. 179).

Default Data Storage and Persistence

Instances that use instance store for the root device automatically have instance stores available on them (for the root partition and other data you want to add). Any data on those stores is deleted if the instance fails or terminates (except for data on the root device). However, after you launch an instance, you can provide it with persistent non-root data by attaching one or more Amazon EBS volumes.

Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. The volume appears in your list of volumes like any other. The instances don't expose ephemeral storage by default. However, you can add ephemeral storage by using a block device mapping. You can also add additional Amazon EBS volumes for non-root data (for more information, see Block Device Mapping (p. 167)). For information about what happens to the ephemeral storage and volumes when you stop an instance, see Stop/Start (p. 179).

Boot Times

Amazon EBS-backed AMIs launch faster than Amazon EC2 instance store-backed AMIs. When you launch an Amazon EC2 instance store-backed AMI, all the parts have to be retrieved from Amazon S3 before the instance is available. With an Amazon EBS-backed AMI, only the parts required to boot the instance need to be retrieved from the snapshot before the instance is available. However, the performance of an instance that uses an Amazon EBS volume for its root device is slower for a short time while the remaining parts are retrieved from the snapshot and loaded into the volume. When you stop and restart the instance, it launches quickly, because the state is stored in an Amazon EBS volume.

AMI Creation

To create Linux/UNIX AMIs backed by instance store, you must create an image of your instance on the instance itself, and there aren't any APIs available to help you. To create a Windows AMI backed by instance store, there's an API call that creates an image, but you still have to call another API to register the AMI.

AMI creation is much easier for AMIs backed by Amazon EBS. There's a single API action called `CreateImage` that works for both Linux/UNIX and Windows. The single call bundles your Amazon EBS-backed AMI and registers it. There's also a button in the AWS Management Console that lets you create an image from a running instance. For more information, see Creating Amazon EBS-Backed AMIs (p. 18).

How You're Charged

With AMIs backed by instance store, you're charged for AMI storage and instance usage. With AMIs backed by Amazon EBS, you're charged for volume storage and usage in addition to the AMI and instance usage charges.

With Amazon EC2 instance store-backed AMIs, each time you customize an AMI and create a new one, all of the parts are stored in Amazon S3 for each AMI. So, the storage footprint for each customized AMI is the full size of the AMI. For Amazon EBS-backed AMIs, each time you customize an AMI and create a new one, only the changes are stored. So the storage footprint for subsequent AMIs you customize after the first is much smaller, resulting in lower AMI storage charges.

When an Amazon EBS-backed instance is stopped, you're not charged for instance usage; however, you're still charged for volume storage. We charge a full instance hour for every transition from a stopped state to a running state, even if you transition the instance multiple times within a single hour. For example, let's say the hourly instance charge for your instance is $0.10. If you were to run that instance for one hour without stopping it, you would be charged $0.10. If you stopped and restarted that instance twice during that hour, you would be charged $0.30 for that hour of usage (the initial $0.10, plus 2 x $0.10 for each restart).

The Root Device Volume

When you launch an instance of an Amazon EBS-backed AMI, the instance has an Amazon EBS volume attached to it for the root device. The volume is created from a snapshot associated at the time of AMI creation. If someone shares an Amazon EBS-backed AMI with you, you can still launch the AMI even if they haven't shared the corresponding snapshot with you. You own the resulting instance and volume. The new volume is like any other Amazon EBS volume. You can only access it when it's attached to a running instance. Any changes you make to the volume are saved (as with any other volume). You should regularly create a snapshot of the volume for backup purposes. For more information, see Root Device Storage (p. 160).

Stop/Start

When you stop an instance, the following things happen:

- The instance performs a normal shutdown and stops running (the status switches to *stopping* and then to *stopped*)
- The Amazon EBS volumes remain attached, but any ephemeral storage does not persist
- The instance retains its instance ID
- The instance will probably not retain its public and private IP addresses (exception: instances launched in Amazon VPC; they keep the same IP address when stopped and restarted)
- Any elastic IP address that was mapped to the instance is unmapped

Note

You can't launch an instance directly into the *stopped* state. You can only stop an instance after its status is *running*.

Let's address a few details for the items in the preceding list.

When the instance is stopped, you're not charged for any instance hours, only the volume storage. Each time you transition an instance from stopped to started, we charge a full instance hour, even if transitions happen multiple times within a single hour.

The root device volume and any others that you added either at the time of launch (through block device mapping) or after launch (by attaching volumes) remain attached while the instance is stopped. Any ephemeral storage does not persist. While the instance is stopped, you can't change which devices are mapped to the instance.

When you stop an instance, it will probably lose its public and private IP addresses and get new ones when you restart the instance. Exception: instances you launch with Amazon VPC retain the same private IP address when stopped and then started; it's therefore possible to have an Amazon VPC subnet with no running instances (they're all stopped), and also no remaining IP addresses available.

When you restart a Windows instance, by default, the Ec2ConfigService changes the instance hostname to match the new IP address and initiates a reboot. For more information about this service, see Appendix C: Windows Configuration Service (p. 424).

If you were using an elastic IP address with the instance, the address is automatically unmapped when the instance stops. While the instance is stopped, you're charged for the address being unmapped (unless you remap it to another instance). When you restart the instance, you need to manually remap the elastic IP address to the instance; it doesn't happen automatically. For more information about the charges for elastic IP addresses, go to the Amazon EC2 product page.

Each Amazon EBS-backed instance has an attribute called `InstanceInitiatedShutdownBehavior` that controls whether the instance stops or terminates when you initiate a shutdown from within the instance itself. The default is stop. You can modify the attribute while the instance is running or stopped.

You can modify several other instance attributes only while the instance is stopped:

- Kernel
- RAM disk
- Instance type
- User data

While the instance is stopped, you can't change anything about the instance's block device mapping except the `DeleteOnTermination` flag for an attached volume (you can also change the flag while the instance is running). For more information about modifying instance attributes while the instance is stopped, see Modifying Attributes of a Stopped Instance (p. 190).

Note

Although you can use the AWS Management Console to stop and start instances, you can't use it to modify any instance attributes. That functionality is available only through the command line tools or API.

While the instance is stopped, you can treat the root volume like any other volume, and modify it (e.g., repair file system problems or update software). You just detach the volume from the stopped instance, attach it to a running instance, make your changes, detach it from the running instance, and then reattach it to the stopped instance. Make sure you reattach it to the correct storage device (whichever device name is specified as the root device in the instance's block device mapping).

Instance Termination

When an instance terminates, any instance store associated with that instance is deleted.

By default, any volumes that were created when the instance launched (the root device and any others specified in the block device mapping) are automatically deleted when the instance terminates (in other words, their `DeleteOnTermination` flags are set to `true` by default). However, any volumes that you attached *after* the instance was running are not (their `DeleteOnTermination` flags are set to `false` by default). If you detach and then reattach a volume that was created at instance launch, it's treated like a new volume that you attached after the launch, and its `DeleteOnTermination` flag is set to `false`.

You can see the value for the `DeleteOnTermination` flag for each of the attached volumes by looking at the instance's block device mapping (for more information, see Viewing Block Device Mappings (p. 170)). You currently can't change the flag's value once the instance is launched; however, you can launch the instance with your desired value for the flag. In the launch request, you just specify a block device mapping with the value. For an example of how to change the flag at launch time, see Changing the Root Volume to Persist (p. 165).

What happens to the volume when you terminate the instance it's attached to? Assuming the volume's `DeleteOnTermination` flag is `false`, the volume persists in its current state. You could take a snapshot of it, and you could attach it to any other instance you have.

Note

When you launch an instance of a particular AMI and then terminate the instance, any remaining volumes are not automatically related to any future instances of that AMI. In other words, a new instance that you launch of that same AMI doesn't attempt to attach to the remaining volume. However, you can manually attach the remaining volume to the new instance if you want.

For more information on persisting the data after instance termination, see Data Persistence after Instance Termination (p. 129).

Starting with the 2009-10-31 API version, you can prevent an instance from being terminated. This feature is available for both Amazon EC2 instance store-backed and Amazon EBS-backed instances. Each instance has a `DisableApiTermination` attribute that defaults to `false` (i.e., the instance can be terminated). You can modify this instance attribute while the instance is running or stopped (in the case of Amazon EBS-backed instances).

Shared AMIs Backed by Amazon EBS

You can share Amazon EBS-backed AMIs like you share Amazon EC2 instance store-backed AMIs. You don't need to share the snapshot that the AMI references. Even though people you share the AMI with get a root volume created from the snapshot when they launch an instance, they can't separately create a volume from that snapshot.

Limits on Number of Instances and Volumes

Your AWS account has a limit on the number of running instances you can have. There's also a limit on the *overall* number of instances you can have with any status except `Terminated`. This overall instance limit is 2x the running instance limit. You can request to increase your running instance limit by completing the Amazon EC2 Instance Limit Request Form.

Your account is also limited to 5000 Amazon EBS volumes, or 20 TiB in total volume storage, whichever you reach first. The maximum volume size is 1 TiB. You can request to increase your volume limit by completing the Amazon EBS Volume Limit Request Form.

Launching, Stopping, and Starting

This section walks you through launching an instance from an Amazon EBS-backed AMI, stopping the instance, and then restarting it.

We assume you've already walked through the Amazon Elastic Compute Cloud Getting Started Guide and know how to use the AWS Management Console to launch, connect to, and terminate an instance.

Important

The instance you're about to launch will be live (and not running in a sandbox). You will incur the standard Amazon EC2 usage fees for the instance until you terminate it. The total charges will be minimal (typically less than a dollar). For more information about Amazon EC2 usage rates, go to the Amazon EC2 product page.

Each time you transition an instance from stopped to started, we charge a full instance hour, even if transitions happen multiple times within a single hour.

1. Locate the AMI to launch:

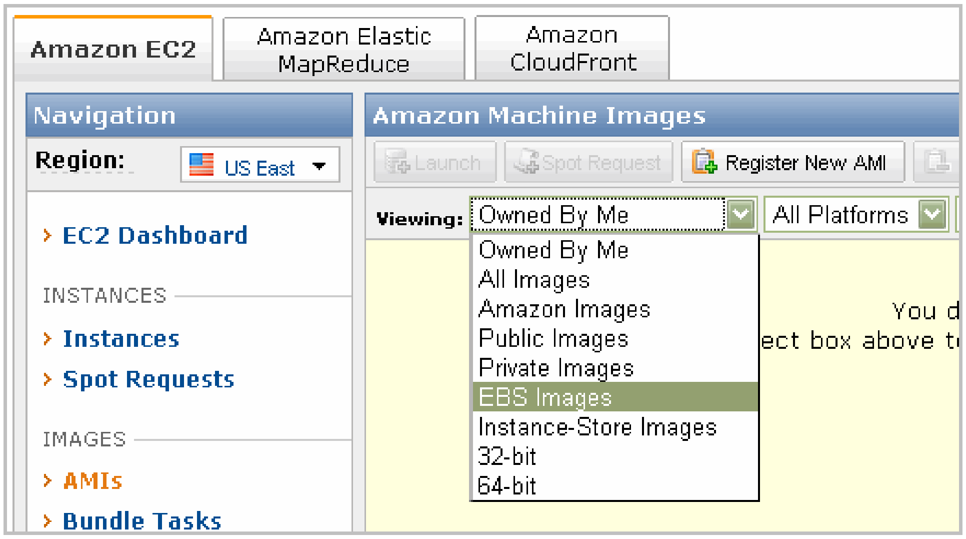
   a. Log in to the AWS Management Console and go to the EC2 Dashboard.

Note

If you're using Amazon VPC, make sure to log in with the email address and password of the AWS account that owns your VPC and subnets.

b. Click **AMIs** in the navigation on the left.
A list of AMIs is displayed.

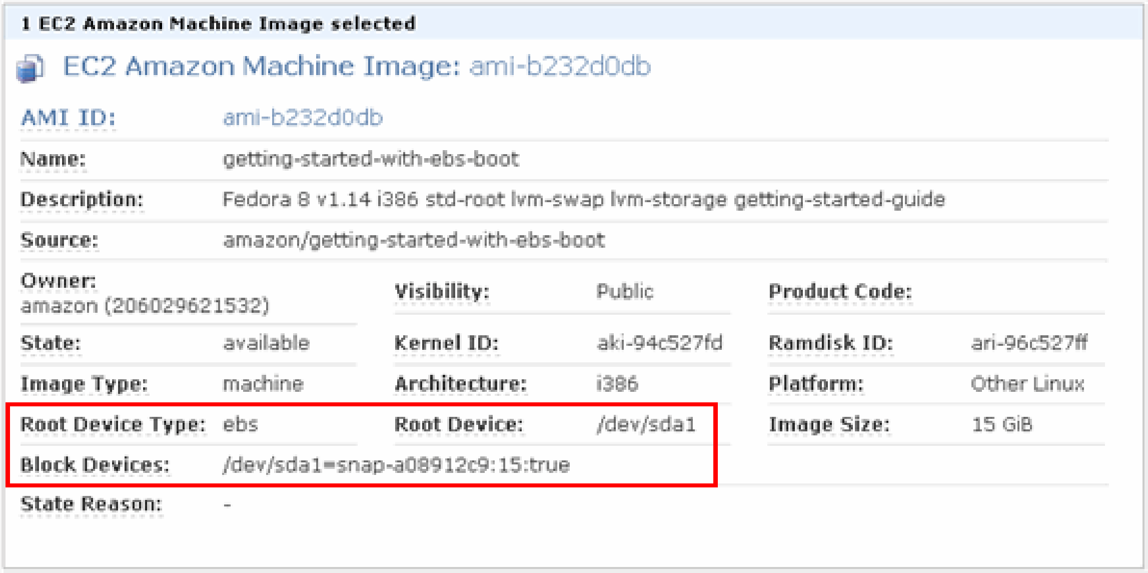
c. In the **Viewing** menu, select **EBS Images** to filter the list.



A list of Amazon EBS-backed AMIs is displayed.

d. Click the label on the **Owner** column to sort the AMIs by owner.
The AMIs created by Amazon (Owner=amazon) appear at the top of the list.

e. Locate and click the *amazon/getting-started-with-ebs-boot* AMI and look at its details.



1 EC2 Amazon Machine Image selected

EC2 Amazon Machine Image: ami-b232d0db

AMI ID: ami-b232d0db
Name: getting-started-with-ebs-boot
Description: Fedora 8 v1.14 i386 std-root lvm-swap lvm-storage getting-started-guide
Source: amazon/getting-started-with-ebs-boot

Owner: amazon (206029621532)		Visibility:	Public	Product Code:	
State:	available	Kernel ID:	aki-94c527fd	Ramdisk ID:	ari-96c527ff
Image Type:	machine	Architecture:	i386	Platform:	Other Linux
Root Device Type:	ebs	Root Device:	/dev/sda1	Image Size:	15 GiB

Block Devices: /dev/sda1=snap-a08912c9:15:true
State Reason: -

Notice that the **Root Device Type** is *ebs* and the **Root Device** is */dev/sda1*. Notice also that the **Block Devices** field shows information about the root device's mapping:

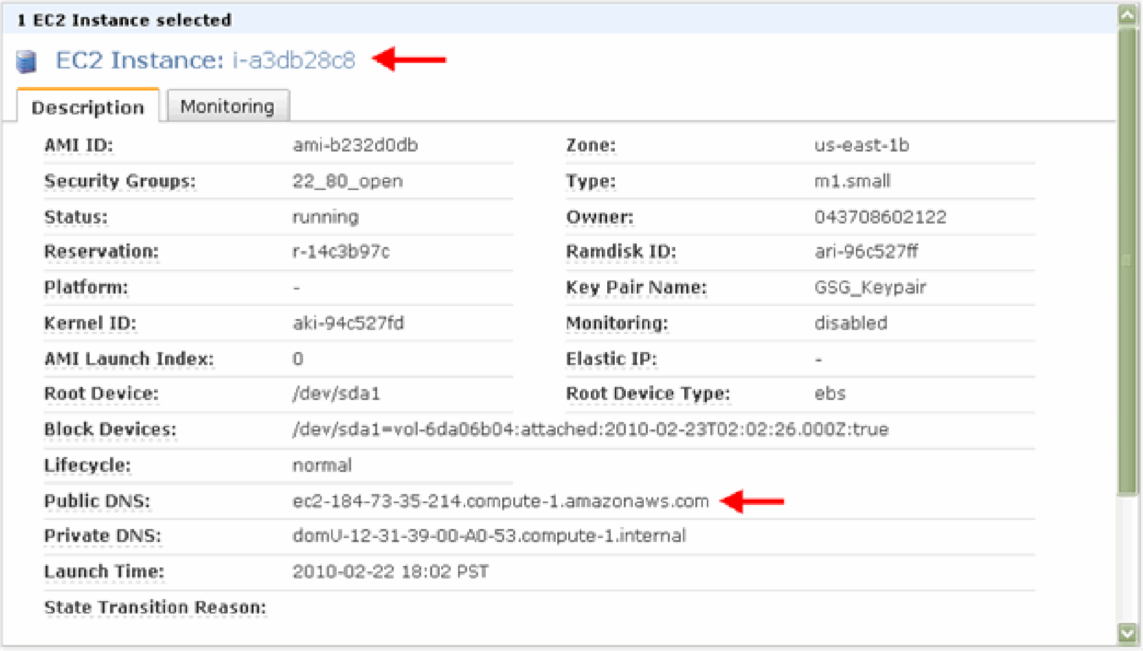
/dev/sda1=snap-a08912c9:15:true. This means the root device /dev/sda1 is mapped to a 15 GiB volume created from the snap-a08912c9 snapshot, and the volume's `DeleteOnTermination` flag is `true`.

2. Launch an instance:

   a. Right-click the AMI in the list and click **Launch Instance**.

      The launch wizard starts. Walk through the wizard and launch an instance. If you're using Amazon VPC, make sure to specify the subnet you want to launch the instance into. If you'd like, connect to the instance once it's running.

   b. Go to the **Instances** page and look at the instance's information. Specifically take note of the instance ID and public DNS.



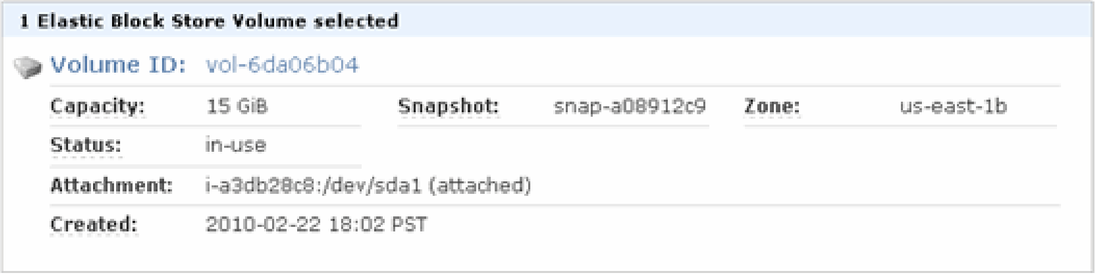
1 EC2 Instance selected

EC2 Instance: i-a3db28c8

AMI ID:	ami-b232d0db	Zone:	us-east-1b
Security Groups:	22_80_open	Type:	m1.small
Status:	running	Owner:	043708602122
Reservation:	r-14c3b97c	Ramdisk ID:	ari-96c527ff
Platform:	-	Key Pair Name:	GSG_Keypair
Kernel ID:	aki-94c527fd	Monitoring:	disabled
AMI Launch Index:	0	Elastic IP:	-
Root Device:	/dev/sda1	Root Device Type:	ebs
Block Devices:	/dev/sda1=vol-6da06b04:attached:2010-02-23T02:02:26.000Z:true		
Lifecycle:	normal		
Public DNS:	ec2-184-73-35-214.compute-1.amazonaws.com		
Private DNS:	domU-12-31-39-00-A0-53.compute-1.internal		
Launch Time:	2010-02-22 18:02 PST		
State Transition Reason:			

      For VPC users: Take note of the instance ID and private IP address. You might need to click **Show/Hide** in the top right corner of the dashboard and turn on the field called **Private IP**. This adds a new **Private IP** column to the list of instances.

   c. Go to the **Volumes** page and look at the volume that was created when you launched the instance. The snapshot ID is the one listed in the AMI's **Block Devices** field (which we saw in a previous step).



1 Elastic Block Store Volume selected

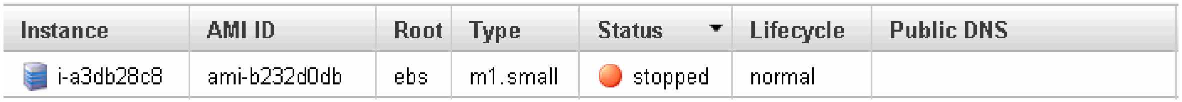
Volume ID: vol-6da06b04

Capacity:	15 GiB	Snapshot:	snap-a08912c9	Zone:	us-east-1b
Status:	in-use				
Attachment:	i-a3db28c8:/dev/sda1 (attached)				
Created:	2010-02-22 18:02 PST				

3. Stop the instance:

   a. Go back to the **Instances** page, right-click the instance, and select **Stop**.

      The instance's status changes to *stopping*, and then to *stopped*. Notice that the **Public DNS** field on the far right is now blank.



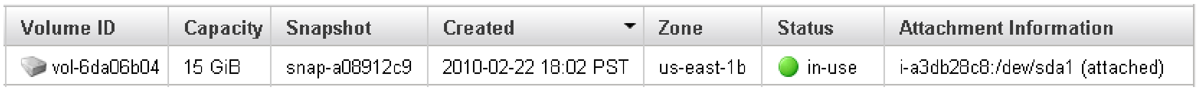
Instance	AMI ID	Root	Type	Status	Lifecycle	Public DNS
i-a3db28c8	ami-b232d0db	ebs	m1.small	stopped	normal	

Note

If you associated an Elastic IP address with this instance, stopping this instance also disassociates the Elastic IP address from it.

For VPC users: Notice that the **Private IP** field is now blank.

   b. Go back to the **Volumes** page and notice that the root device volume is still there and in use even though the instance is stopped.



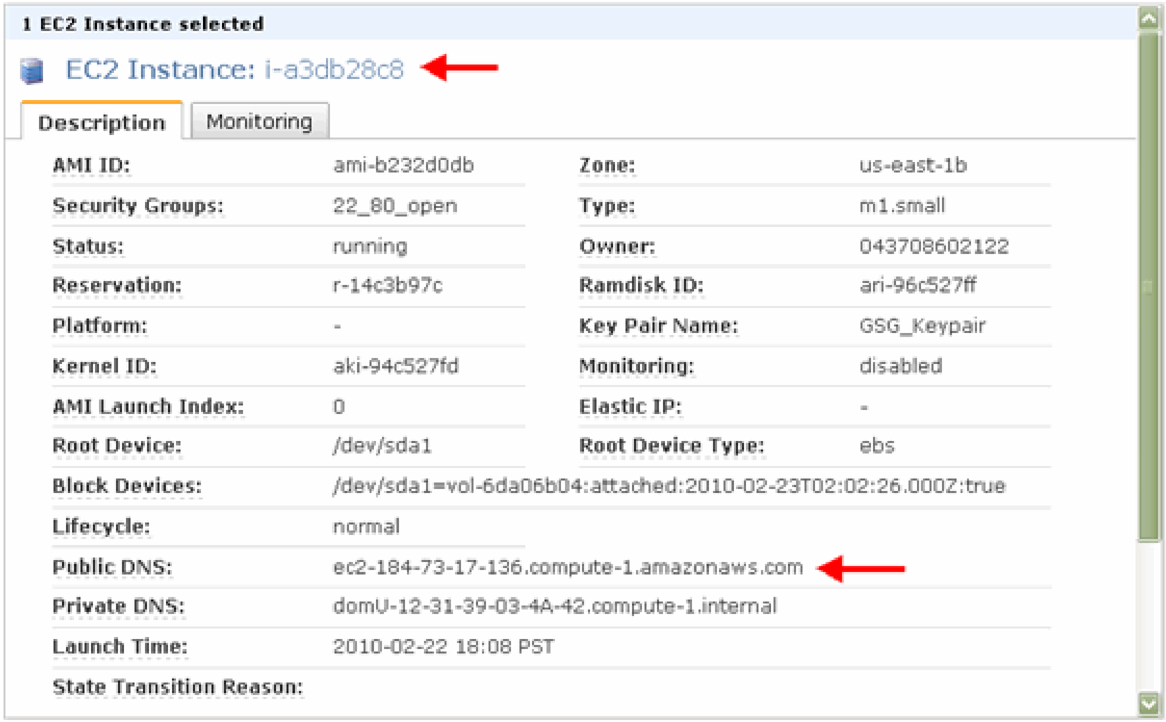
Volume ID	Capacity	Snapshot	Created	Zone	Status	Attachment Information
vol-6da06b04	15 GiB	snap-a08912c9	2010-02-22 18:02 PST	us-east-1b	in-use	i-a3db28c8:/dev/sda1 (attached)

Tip

At this point you can treat the volume like any other Amazon EBS volume, and modify it (e.g., repair file system problems or update software). You just detach the volume from the stopped instance, attach it to a running instance, make your changes, detach it from the running instance, and then reattach it to the stopped instance. Make sure you reattach it to the correct storage device (whichever device name is specified as the root device in the instance's block device mapping).

4. Go back to the **Instances** page, right-click the instance, and select **Start** to restart the instance.

   The instance's status changes to *pending*, and then to *running*. Notice that the instance ID is still the same, but the public DNS has changed.



1 EC2 Instance selected

EC2 Instance: i-a3db28c8

AMI ID:	ami-b232d0db	Zone:	us-east-1b
Security Groups:	22_80_open	Type:	m1.small
Status:	running	Owner:	043708602122
Reservation:	r-14c3b97c	Ramdisk ID:	ari-96c527ff
Platform:	-	Key Pair Name:	GSG_Keypair
Kernel ID:	aki-94c527fd	Monitoring:	disabled
AMI Launch Index:	0	Elastic IP:	-
Root Device:	/dev/sda1	Root Device Type:	ebs
Block Devices:	/dev/sda1=vol-6da06b04:attached:2010-02-23T02:02:26.000Z:true		
Lifecycle:	normal		
Public DNS:	ec2-184-73-17-136.compute-1.amazonaws.com		
Private DNS:	domU-12-31-39-03-4A-42.compute-1.internal		
Launch Time:	2010-02-22 18:08 PST		
State Transition Reason:			

For VPC users: Notice that the private IP address is still the same as before you stopped the instance. Instances in a VPC maintain their private IP address when you stop and start them. It's therefore possible to have a subnet with no running instances (they're all stopped), and also no remaining IP addresses available.

Tip

At this point, you can leave your instance running, and go to the next section on creating an image from your instance (see Creating an Image from a Running Instance (p. 185)). If you'd rather stop here, we recommend you terminate your instance to ensure you stop incurring charges for it. The root device volume is set to automatically delete upon instance termination, so you won't be charged for storage of that volume once you terminate the instance.

Tip

When you launch an instance, you can set the root device volume to persist when the instance terminates. For more information, see Changing the Root Volume to Persist (p. 165).

Creating an Image from a Running Instance

This section walks you through creating an Amazon EBS-backed AMI from a running Amazon EBS-backed instance. If you don't have a running instance that uses an Amazon EBS volume for the root device, you must start one up (for instructions, see Launching, Stopping, and Starting (p. 181)).

If you want to customize the instance, go ahead so you can later validate that the AMI you create from the instance is actually different from the original AMI you used to launch the instance. For example, you can add another volume to the instance first, or modify the root volume.

Tip

As part of the process of creating your new AMI, we power down the instance and then reboot it. If you prefer the instance not be rebooted, you can use the Amazon EC2 command line tools to create the image instead of the AWS Management Console. The `ec2-create-image` command has a `--no-reboot` option. When you use the option, the file system integrity on the created image can't be guaranteed. For more information, go to ec2-create-image in the *Amazon Elastic Compute Cloud Command Line Reference*.

To create an AMI from a running Amazon EBS-backed instance

1. On the **Instances** page, right-click your running instance and select **Create Image (EBS AMI)**.

   Tip

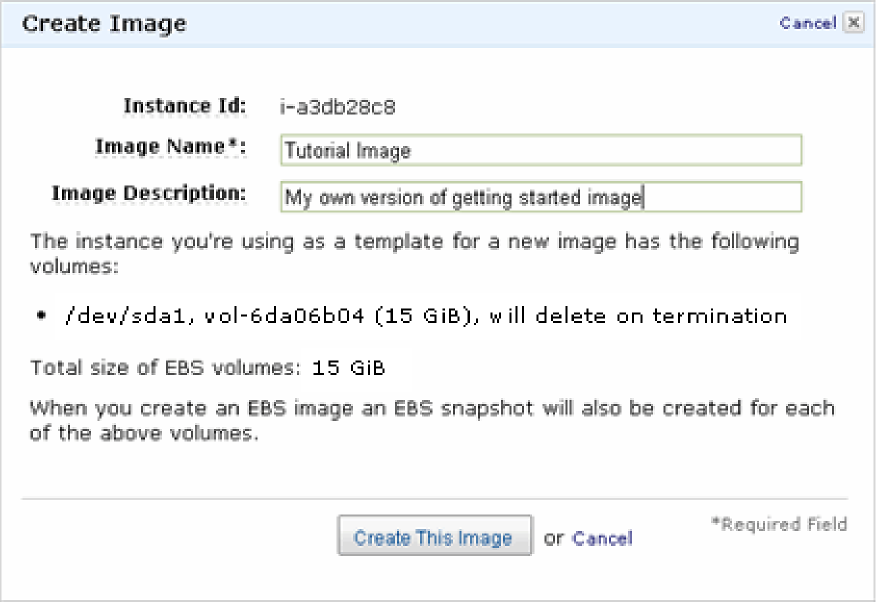
   If you don't see this option in the menu, your instance isn't an Amazon EBS-backed instance. For instructions on starting one, see Launching, Stopping, and Starting (p. 181).

   The **Create Image** dialog box opens.

2. Fill in a unique image name and an optional description of the image (up to 255 characters), and click **Create This Image**.

   Tip

   If you're familiar with Amazon EC2 instance store-backed AMIs, the image name replaces the manifest name (e.g., s3_bucket/something_of_your_choice.manifest.xml), which uniquely identifies each Amazon EC2 instance store-backed AMI.



Create Image

Instance Id: i-a3db28c8

Image Name*: Tutorial Image

Image Description: My own version of getting started image

The instance you're using as a template for a new image has the following volumes:

- /dev/sda1, vol-6da06b04 (15 GiB), will delete on termination

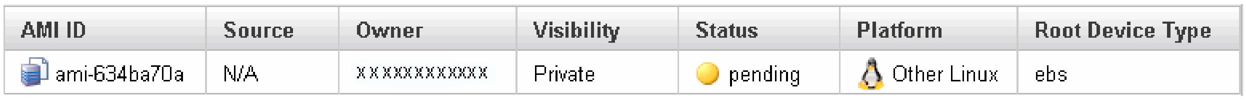
Total size of EBS volumes: 15 GiB

When you create an EBS image an EBS snapshot will also be created for each of the above volumes.

*Required Field

   Amazon EC2 powers down the instance, takes images of any volumes that were attached, creates and registers the AMI, and then reboots the instance.

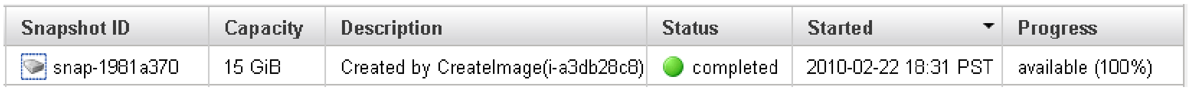
3. Go to the **AMIs** page and view the AMI's status. While the new AMI is being created, its status is *pending*.



AMI ID	Source	Owner	Visibility	Status	Platform	Root Device Type
ami-634ba70a	N/A	xxxxxxxxxxxx	Private	pending	Other Linux	ebs

It takes a few minutes for the whole process to finish.

4. Once your new AMI's status is *available*, go to the **Snapshots** page and view the new snapshot that was created for the new AMI. Any instance you launch from the new AMI uses this snapshot for its root device volume.



Snapshot ID	Capacity	Description	Status	Started	Progress
snap-1981a370	15 GiB	Created by CreateImage(i-a3db28c8)	completed	2010-02-22 18:31 PST	available (100%)

5. Go back to the **AMIs** page, right-click the image, and select **Launch Instance**.
   The launch wizard opens.
6. Walk through the wizard to launch an instance of your new AMI.
7. Once your instance's status is *running*, connect to the instance and verify that any changes you made to the original AMI have persisted.

You now have a new AMI and snapshot that you just created. Both will continue to incur charges to your account until you stop or delete them.

To delete an AMI and a snapshot

1. Go to the **AMIs** page, right-click the AMI, and select **De-register Image**.
   The image is de-registered, which means it is deleted and can no longer be launched.
2. Go to the **Snapshots** page, right-click the snapshot, and select **Delete Snapshot**.
   The snapshot is deleted.

Automatically Attaching Volumes

By default, every Amazon EBS-backed AMI has a snapshot associated with it for the root device volume. Whenever you launch an instance of the AMI, that snapshot automatically instantiates and a volume for the root device attaches to the instance.

What if you want other volumes to automatically attach to your instance? You can easily do that by specifying a block device mapping when you register the image. You can register an image using the command line tools or API; you can't use the AWS Management Console. For more information about block device mappings, see Block Device Mapping (p. 167).

Command Line Tools

You use `ec2-register` and specify a block device mapping that includes the additional volumes. In the example shown here, you register an AMI and attach two extra volumes (in addition to the root device volume).

To add additional volumes to the AMI

- Use the `ec2-register` command with the block device mapping information. The following example registers an AMI with three volumes (one for the root device, and two extra):
  - The first is an 80 GiB root device volume at /dev/sda1 created from the snap-12345678 snapshot. The root volume's `DeleteOnTermination` flag is set to `false`. Remember that if you specify a

size, it must be equal to or larger than the snapshot's size. You can omit a size value and the volume uses the snapshot's size.

- The second volume is mapped to /dev/sdh and created from snapshot snap-88888888. The mapping uses the default size of the snapshot and doesn't specify a value for `DeleteOnTermination`. The default value is `true`.
- The third volume is an empty 40 GiB volume mapped to /dev/sdj. The volume's `DeleteOnTermination` flag is set to `false`.

```
PROMPT>  ec2-register -n My_Image_Name -d My_image_description --root-device-
name /dev/sda1 -b /dev/sda1=snap-12345678:80:false -b /dev/sdh=snap-88888888
 -b /dev/sdj=:40:false
```

In response, you get the ID for your new AMI.

```
IMAGE     ami-61a54008
```

Any instance of the AMI that you launch includes the three volumes by default.

You can view the AMI's or instance's block device mapping at any time. For more information, see Viewing Block Device Mappings (p. 170).

API

You use `RegisterImage` and specify a block device mapping that includes the additional volumes. The following example registers an AMI and attaches two extra volumes (in addition to the root device volume).

To add additional volumes to the AMI

- Issue the following Query request to register the image.

  The example registers an AMI with three volumes:

  - The first is an 80 GiB root device volume at /dev/sda1 created from the snap-12345678 snapshot. The root volume's `deleteOnTermination` flag is set to `false`. Remember that if you specify a size, it must be equal to or larger than the snapshot's size. If you omit a size value, the volume uses the snapshot's size.
  - The second volume is mapped to /dev/sdh and created from snapshot snap-88888888. The mapping uses the default size of the snapshot and doesn't specify a value for `deleteOnTermination`. The default value is `true`.
  - The third volume is an empty 40 GiB volume mapped to /dev/sdj. The volume's `deleteOnTermination` flag is set to `false`.

```
https://ec2.amazonaws.com/
?Action=RegisterImage
&Name=MyImage
&KernelId=aki-f70657b2
&RamdiskId=ari-ff0d5cba
&RootDeviceName=/dev/sda1
&BlockDeviceMapping.1.DeviceName=/dev/sda1
&BlockDeviceMapping.1.Ebs.SnapshotId=snap-12345678
&BlockDeviceMapping.1.Ebs.VolumeSize=80
```

```
&BlockDeviceMapping.1.Ebs.DeleteOnTermination=false
&BlockDeviceMapping.2.DeviceName=/dev/sdh
&BlockDeviceMapping.2.Ebs.SnapshotId=snap-88888888
&BlockDeviceMapping.3.DeviceName=/dev/sdj
&BlockDeviceMapping.3.Ebs.VolumeSize=40
&BlockDeviceMapping.3.Ebs.DeleteOnTermination=false
&...auth parameters...
```

For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

Following is an example response.

```
<RegisterImageResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <imageId>ami-61a54008</imageId>
</RegisterImageResponse>
```

Any instance of the AMI that you launch includes the three volumes by default.

You can view the AMI's or instance's block device mapping at any time. For more information, see Viewing Block Device Mappings (p. 170).

Launching an Instance from a Snapshot

If you have a snapshot of the root device volume of an instance, you can terminate that instance and then later launch a new instance from the snapshot. This is useful if you don't have the original AMI available to launch new instances from.

To launch a new instance from the snapshot, you register the snapshot and then launch the resulting AMI.

Important

Registering a snapshot works only for Linux/UNIX AMIs; although you can register a snapshot to create a Windows AMI, the AMI isn't launchable.

Command Line tools

1. Use the `ec2-register` command and specify a block device mapping that maps the root device name of your choice to the snapshot.

   The following example specifies the root device as /dev/sda1, and maps it to the snap-12345678 snapshot. The resulting root device volume is the same size as the snapshot, and it will automatically be deleted on instance termination. If you were to specify the block device mapping as `/dev/sda1=snap-1234578::false`, then the volume would persist on instance termination.

   ```
   PROMPT>  ec2-register -n My_Image_Name -d My_image_description --root-device-name /dev/sda1 -b /dev/sda1=snap-12345678
   ```

   The response displays the ID for your new AMI.

```
IMAGE    ami-61a54008
```

The AMI now appears in the list of AMIs that you own. You can view that list in the AWS Management Console, or by using the following command: `ec2-describe-images -o self`.

2. Launch an instance of the AMI.

   The resulting instance has a root device volume created from the snapshot.

API

1. Issue the following Query request to register an image.

   The example specifies the root device as /dev/sda1, and maps it to the snap-12345678 snapshot. The resulting root device volume is the same size as the snapshot, and it will automatically be deleted on instance termination. You can set `DeleteOnTermination` to `false` if you'd rather the volume persist.

```
https://ec2.amazonaws.com/
?Action=RegisterImage
&Name=MyImage
&KernelId=aki-f70657b2
&RamdiskId=ari-ff0d5cba
&RootDeviceName=/dev/sda1
&BlockDeviceMapping.1.DeviceName=/dev/sda1
&BlockDeviceMapping.1.Ebs.SnapshotId=snap-12345678
&...auth parameters...
```

   For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

   Following is an example response.

```
<RegisterImageResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <imageId>ami-61a54008</imageId>
</RegisterImageResponse>
```

   The AMI now appears in the list of AMIs that you own. You can view that list by using `DescribeImages` with *`Owner`*`=self`.

2. Launch an instance of the AMI.

   The resulting instance has a root device volume created from the snapshot.

Modifying Attributes of a Stopped Instance

Instances have a set of attributes, and you can modify some of the attributes only when the instance is stopped. Following are those attributes:

- Kernel
- RAM disk
- Instance type
- User data

If you try to modify one of these attributes while the instance is running, Amazon EC2 returns the `IncorrectInstanceState` error.

For a list of all the available instance attributes you can change (not just those that require the instance to be stopped), go to ec2-modify-instance-attribute in the *Amazon Elastic Compute Cloud Command Line Reference*.

Note

Although you can use the AWS Management Console to stop and start instances, you can't use it to modify any instance attributes. That functionality is available only through the command line tools or API.

For information about kernels and RAM disks, see Kernels and RAM Disk FAQ (p. 418).

Command Line Tools

You can modify only a single attribute with each command.

To change an attribute of a stopped instance

1. Choose one of the following commands to get the current value of the attribute of interest.

```
PROMPT>  ec2-describe-instance-attribute instance_id --kernel
```

```
PROMPT>  ec2-describe-instance-attribute instance_id --ramdisk
```

```
PROMPT>  ec2-describe-instance-attribute instance_id --instance-type
```

```
PROMPT>  ec2-describe-instance-attribute instance_id --user-data
```

Following is a sample response from the first command in the preceding list.

```
kernel  i-87ad5eec      aki-94c527fd
```

2. Choose one of the following commands to modify the value of the attribute of interest.

```
PROMPT>  ec2-modify-instance-attribute instance_id --kernel kernel_id
```

```
PROMPT>  ec2-modify-instance-attribute instance_id --ramdisk ramdisk_id
```

```
PROMPT>  ec2-modify-instance-attribute instance_id --instance-type in
stance_type
```

```
PROMPT>  ec2-modify-instance-attribute instance_id --user-data user_data
```

When you restart the instance, the new attribute value takes effect.

API

You can modify only a single attribute in each call.

To change an attribute of a stopped instance

1. Issue the following Query request to first get the current value of the attribute of interest. The example gets the current value for the instance type.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstanceAttribute
   &InstanceId=i-87ad5eec
   &Attribute=instanceType
   &...auth parameters...
   ```

   For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

   Following is an example response.

   ```
   <DescribeInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
   12-01/">
       <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
      <instanceId>i-5d7e8e36</instanceId>
      <instanceType>
         <value>m1.small</value>
      </instanceType>
   </DescribeInstanceAttributeResponse>
   ```

2. Issue the following Query request to modify the attribute of interest. The example changes the instance type to c1.medium.

   ```
   https://ec2.amazonaws.com/
   ?Action=ModifyInstanceAttribute
   &InstanceId=i-87ad5eec
   &Attribute=instanceType
   &Value=c1.medium
   &...auth parameters...
   ```

   Following is an example response.

   ```
   <ModifyInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
   12-01/">
      <return>true</return>
   </ModifyInstanceAttributeResponse>
   ```

Related Topics

- Kernels and RAM Disk FAQ (p. 418)
- Instance Types and Architectures FAQ (p. 408)
- User Data Retrieval (p. 99)

Changing the Instance Initiated Shutdown Behavior

By default, when you initiate a shutdown from within an EBS-backed instance, the instance stops. You can change this so the instance terminates instead. You just modify the `InstanceInitiatedShutdownBehavior` attribute for the instance. You can do this while the instance is either running or stopped.

Command Line tools

To change the InstanceInitiatedShutdownBehavior attribute

1. View the current value for the flag with the following command.

   ```
   PROMPT>  ec2-describe-instance-attribute instance_id --instance-initiated-
   shutdown-behavior
   ```

   Sample response:

   ```
   instanceInitiatedShutdownBehavior      i-87ad5eec      stop
   ```

2. Change the flag's value to `terminate` with the following command.

   ```
   PROMPT>  ec2-modify-instance-attribute instance_id --instance-initiated-
   shutdown-behavior terminate
   ```

   Sample response:

   ```
   instanceInitiatedShutdownBehavior i-87ad5eec terminate
   ```

You can toggle the attribute between `stop` and `terminate` as often as you want.

API

To change the InstanceInitiatedShutdownBehavior attribute

1. Issue the following Query request to get the current value of the attribute.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstanceAttribute
   &InstanceId=i-87ad5eec
   &Attribute=instanceInitiatedShutdownBehavior
   &...auth parameters...
   ```

   For information about the auth parameters, go to Common Query Parameters in the *Amazon Elastic Compute Cloud API Reference*.

   Following is an example response.

```
<DescribeInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <instanceId>i-5d7e8e36</instanceId>
   <instanceInitiatedShutdownBehavior>
      <value>stop</value>
   </instanceInitiatedShutdownBehavior>
</DescribeInstanceAttributeResponse>
```

2. Issue the following Query request to change the attribute's value.

```
https://ec2.amazonaws.com/
?Action=ModifyInstanceAttribute
&InstanceId=i-87ad5eec
&Attribute=instanceInitiatedShutdownBehavior
&Value=terminate
&...auth parameters...
```

Following is an example response.

```
<ModifyInstanceAttributeResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <return>true</return>
</ModifyInstanceAttributeResponse>
```

You can toggle the attribute between `stop` and `terminate` as often as you want.

Using Instances of Your Virtual Machine in Amazon EC2

Topics

There are two ways you can launch an instance in the Amazon Elastic Compute Cloud (Amazon EC2). You can launch an instance from an AMI that you created or selected from a catalog. Or, you can launch an instance from a virtual machine (VM) that you imported from a Citrix Xen, Microsoft Hyper-V, or VMware vSphere virtualization platform. This section covers using VMs from Citrix, Microsoft, or VMware to launch instances.

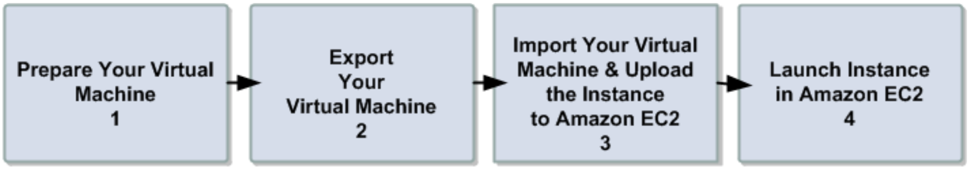
To use your virtual machine as an instance in Amazon EC2, you must first export it from the virtualization platform using the platform's tools. Then you import it to Amazon EC2 using the Amazon EC2 command line or API tools.

If you are importing a VMware vSphere VM, you can also use the Amazon EC2 VM Import Connector for VMware (Connector), a plug-in that integrates with VMware vSphere Client, to perform the task.

Whether you use the command line tools or the API, or the Connector, you will follow the same general process for importing VMs or volumes to Amazon EC2. You need to complete these tasks, which are all discussed in this section:

1. Prepare the virtual machine for import to Amazon EC2. For more information, go to Before You Get Started (p. 196)
2. Export the virtual machine from the external environment.
   - Exporting from Citrix (p. 212)
   - Exporting from Microsoft Hyper-V (p. 215)
   - Exporting from VMware (p. 217)
3. Import the virtual machine to Amazon EC2.
   For information about using the command line tools to import your VM, see Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2 (p. 211). For information about using the Connector, see Using the Amazon EC2 VM Import Connector to Import Your Virtual Machine to Amazon EC2 (p. 197).
4. Upload the instance to Amazon EC2.
5. Launch the instance in Amazon EC2.

The following figure shows the process of importing a VM or volume.


Prepare Your Virtual Machine
1
Export Your Virtual Machine
2
Import Your Virtual Machine & Upload the Instance to Amazon EC2
3
Launch Instance in Amazon EC2
4

Components in Your VM Environment

The table in this section describes the typical components in your VM environment.

Component	Description	Product Name
Virtualization product	Virtualization service for managing virtual computing infrastructure	Citrix Xen Microsoft Hyper-V VMware vSphere (vSphere)
Client	The software you need on your computer to access and manage your virtualization platform environment	Citrix Xen Center Microsoft Hyper-V Manager VMware vSphere Client
Server	The management platform for the virtualization environment	Citrix XenServer Microsoft Hyper-V VMware vCenter Server
Amazon EC2 VM Import Connector for VMware (Connector)	The virtual appliance, a plug-in to the management platform of the virtualization Server, that enables the import of virtual machines into Amazon EC2 using the Client interface	Citrix Xen (not applicable) Microsoft Hyper-V (not applicable) VMware vCenter—Amazon EC2 VM Import Connector for VMware vCenter (Connector)

Before You Get Started

This section discusses the things you need to know and what you must have before you begin the process of importing your virtual machine.

Operating Systems—The following operating systems can be imported to Amazon EC2:

- Microsoft Windows Server 2003 R2 (Standard, Datacenter, Enterprise).
- Microsoft Windows Server 2008 R1 and R2 (Standard, Datacenter, Enterprise).

Image Formats—We support import of the following image formats for importing both volumes and instances to Amazon Web Services:

- RAW format for importing volumes and instances.
- Virtual Hard Disk (VHD) image formats, which are compatible with Microsoft Hyper-V and Citrix Xen virtualization products.
- ESX Virtual Machine Disk (VMDK) image formats, which are compatible with VMware ESX and VMware vSphere virtualization products.

Note

VMDK images from VMware Workstation are not compatible.

Known Limitations—The importing of instances and volumes is subject to the following limitations:

- You can have up to five conversions and tasks in progress at the same time per Region.
- Typically, you import a compressed version of a disk image; the expanded image cannot exceed 1TB.
- Tasks must complete within 7 days of the start date.
- Importing virtual machines with more than one virtual disk is not supported. We suggest that you import the VM with only the boot volume, and import any additional disks using `ImportVolume` (`ec2-import-volume`) in the command line. After the `ImportInstance` task is complete, use `AttachVolume` (`ec2-attach-volume`) to associate the additional volumes with your instance.
- Multiple network interfaces are not currently supported. When converted and imported, your instance will have a single virtual NIC using DHCP for address assignment.
- For vCenter 4.0 and vSphere 4.0 users, remove any attached CD-ROM images or ISOs from the virtual machine.

Preparing Your Virtual Machine

Use the following guidelines to configure your virtual machine before exporting it from the virtualization environment.

Important

If you are importing a virtual machine from Citrix Xen, you must uninstall the Citrix Tools for Virtual Machines from the VM. If you don't uninstall the tools, your import will fail. For more information, see Exporting from Citrix (p. 212).

- Enable Remote Desktop (RDP) for remote access.

- Make sure your host firewall (Windows firewall), if configured, allows access to RDP. Otherwise, you will not be able to access your instance after the conversion is complete.
- Make sure all user accounts use secure passwords. This includes the administrator account.

Note

All accounts must have passwords. Otherwise, the import might fail.

- Disable any antivirus or intrusion detection software on your virtual machine. These services can be re-enabled after the import process is complete.
- Disconnect any CD-ROM drives (virtual or physical).

Using the Amazon EC2 VM Import Connector to Import Your Virtual Machine to Amazon EC2

Topics

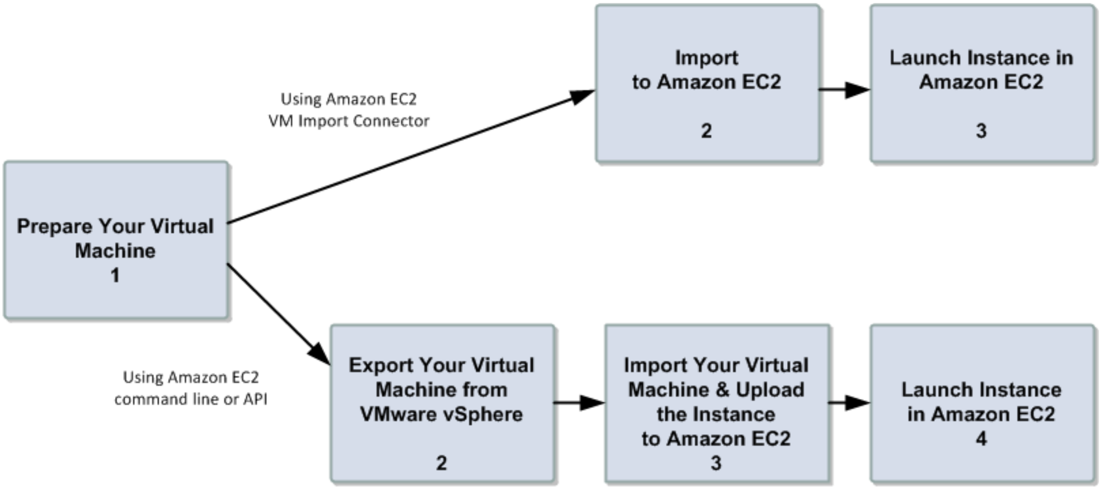
You can use the Amazon EC2 VM Import Connector virtual appliance (vApp), a plug-in for VMware vCenter, to import virtual machines from your VMware vSphere infrastructure to Amazon EC2. The Connector is a virtual appliance that works with VMware vCenter Server only. It provides an easy-to-use interface, enhancing your existing management tools to work with the Amazon EC2 VM Import Connector.

Note

You cannot use the Connector to import Citrix Xen or Microsoft Hyper-V virtual machines to Amazon EC2. Instead, use the command line tools to import your Citrix and Hyper-V virtual machines to Amazon EC2. You can also choose to use the command line tools to import your VMware VMs. For more information, see Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2 (p. 211).

Using the Connector simplifies the process of importing your VMware VMs: You prepare the virtual machine for import to Amazon EC2, and then you import it to EC2.

The following figure shows the process of importing VMware VMs using the Connector and without using the Connector.


Prepare Your Virtual Machine
1
Using Amazon EC2 VM Import Connector
Import to Amazon EC2
2
Launch Instance in Amazon EC2
3
Using Amazon EC2 command line or API
Export Your Virtual Machine from VMware vSphere
2
Import Your Virtual Machine & Upload the Instance to Amazon EC2
3
Launch Instance in Amazon EC2
4

Before You Install the Connector

To use the VM Import Connector, you first need to install the Connector virtual appliance. Before you install, read through the general prerequisites listed in the Before You Get Started (p. 196) section and make sure that your environment meets the following requirements:

VMware infrastructure

- vSphere 4.0 or later
- vCenter 4.0 or later

Amazon EC2 VM Import Connector vApp

- 256MB RAM
- Minimum 250GB of disk space

Note

Although the Connector virtual appliance is small, it temporarily stores the VM images that you import to Amazon EC2. The data store must be large enough to accommodate these images, plus the Connector. We recommend a data store size of 250GB or larger.

To estimate the disk space you need, multiply the maximum number of parallel import tasks you want to run with the Connector by the average size you expect your VMs to be, then add about 10GB for the virtual appliance. For example, if you project that you will run a maximum of 5 imported VMs averaging 75GB in size, then you'll need about 385GB of disk space.

Internet access

- Outbound Internet access, either direct or via a proxy, from the Connector appliance on TCP port 443 (SSL) to Amazon EC2 and Amazon S3.

Note

If you are adding a firewall rule to allow this access and want to further restrict this access to Amazon EC2 and Amazon S3, you can add the hosts at the published endpoints as the destinations on TCP port 443. For more information about the current endpoints go to Regions and Endpoints.

- DHCP server
- A static IP address or an IP reservation via DHCP for the virtual appliance

Note

You must set up a static DHCP lease or configure a static IP before you begin the configuration process.

Connector local network access

- Inbound TCP 443 (SSL) from vCenter Server and vSphere Client
- Inbound TCP 80 (HTTP) from the LAN for Connector Web Console

Administrative rights

- To VMware vSphere and VMware vCenter for installation. For information on how to allow a user without administrative rights to use the Connector to import VMs to Amazon EC2, see To grant permission to non-administrative users to import VMs to Amazon EC2 (p. 206).

AWS access credentials

- The Connector stores AWS credentials for each VMware vCenter user. In this way, multiple users with separate AWS credentials can use the same Connector. Alternatively, you can use the same AWS account and credentials with more than one user. Each VMware vSphere user will need to fill out the information in the **Enter AWS Credentials** dialog box the first time he or she uses the Connector. For information on how to get your AWS credentials, see How to Get Your Access Key ID and Secret Access Key (p. 267).
- Your AWS account must be subscribed to Amazon EC2 before you start the import process.

Note

The Connector virtual appliance stores your AWS credentials. To protect these credentials from unauthorized use, grant access to the virtual appliance's console and configuration only to administrators.

Installing the Connector for VMware vCenter

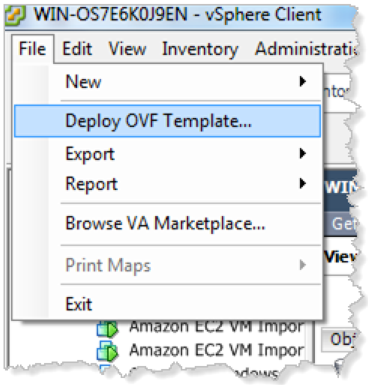
After you have confirmed that you have all the prerequisites and your environment meets the minimum requirements, you are now ready to install the Connector virtual appliance (vApp) for VMware vCenter.

The Connector virtual appliance is an Open Virtualization Format (OVF) package that is distributed in an Open Virtualization Application (OVA) file. Installing the Connector involves downloading the OVA file and deploying the OVF template.

To Install the Connector for the VMware vCenter

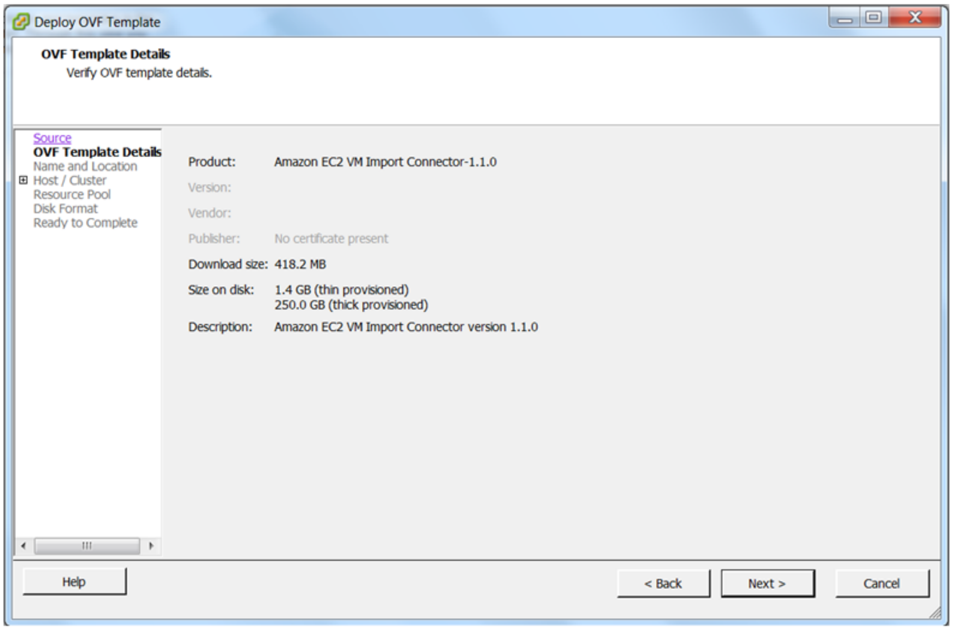
1. Download the OVA package for the Connector virtual appliance from Amazon Web Services Developer Tools and save it to your Downloads folder.
2. Start the vSphere Client and connect to your vCenter Server.

3. Using the vSphere Client, deploy the OVF template contained in the OVA file that you downloaded. On the **File** menu, select **Deploy OVF Template** and point to the location where you downloaded the OVA package.



A series of **Deploy OVF Template** screens walks you through the deployment process.

4. Confirm that the **Deploy OVF Template** screen is displaying correct information about the VM Import Connector, and click **Next**.



OVF Template Details

Verify OVF template details.

Product: Amazon EC2 VM Import Connector-1.1.0

Version:

Vendor:

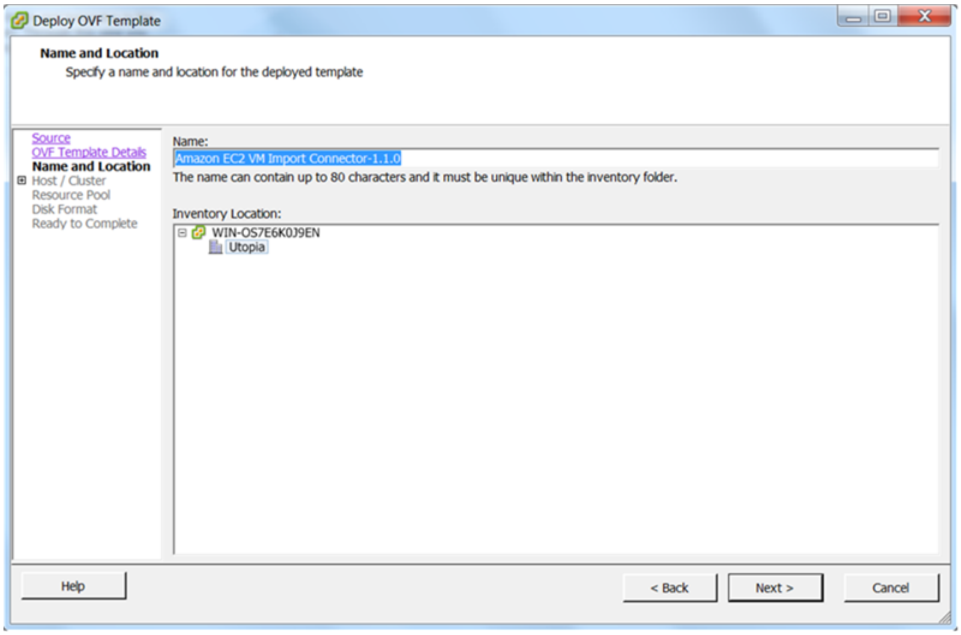
Publisher: No certificate present

Download size: 418.2 MB

Size on disk: 1.4 GB (thin provisioned)
250.0 GB (thick provisioned)

Description: Amazon EC2 VM Import Connector version 1.1.0

5. Specify the name and location of the Connector or accept the default, then click **Next**.



Name and Location

Specify a name and location for the deployed template

Name:

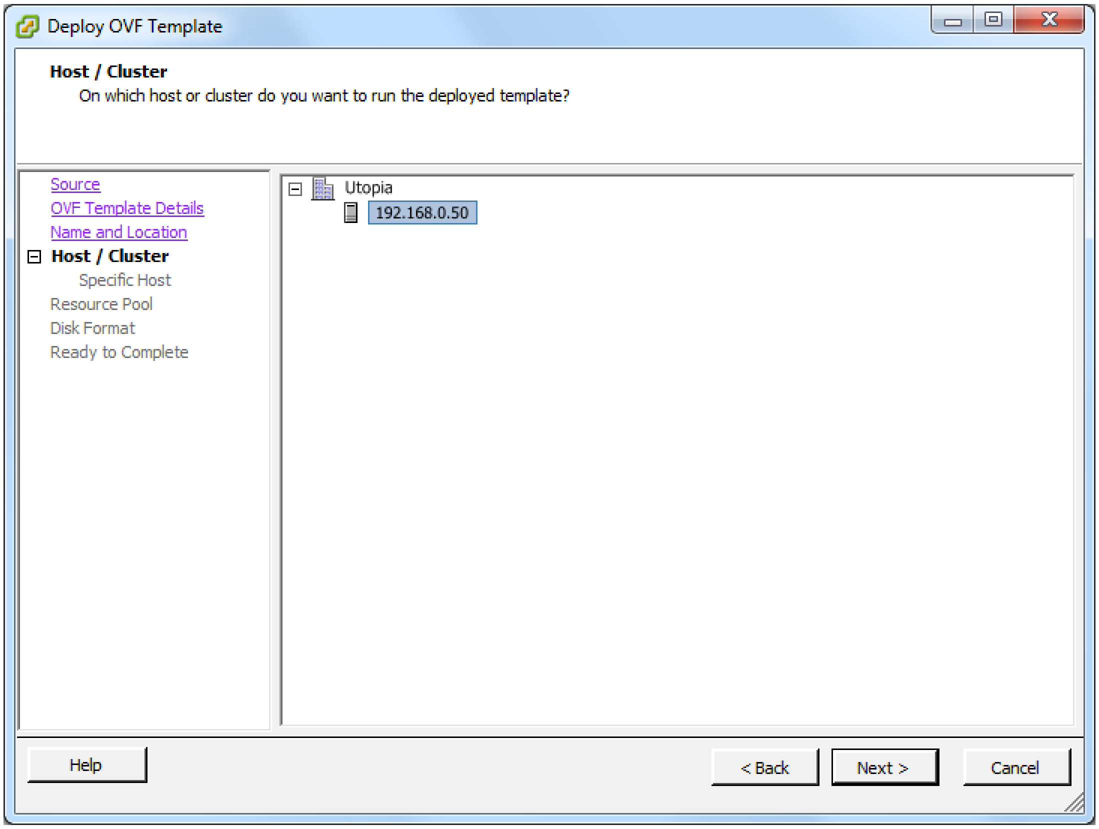
Amazon EC2 VM Import Connector-1.1.0

The name can contain up to 80 characters and it must be unique within the inventory folder.

Inventory Location:

WIN-OS7E6K0J9EN
Utopia

6. Select the host or cluster in which you want the Connector to run, then click **Next**.



Host / Cluster

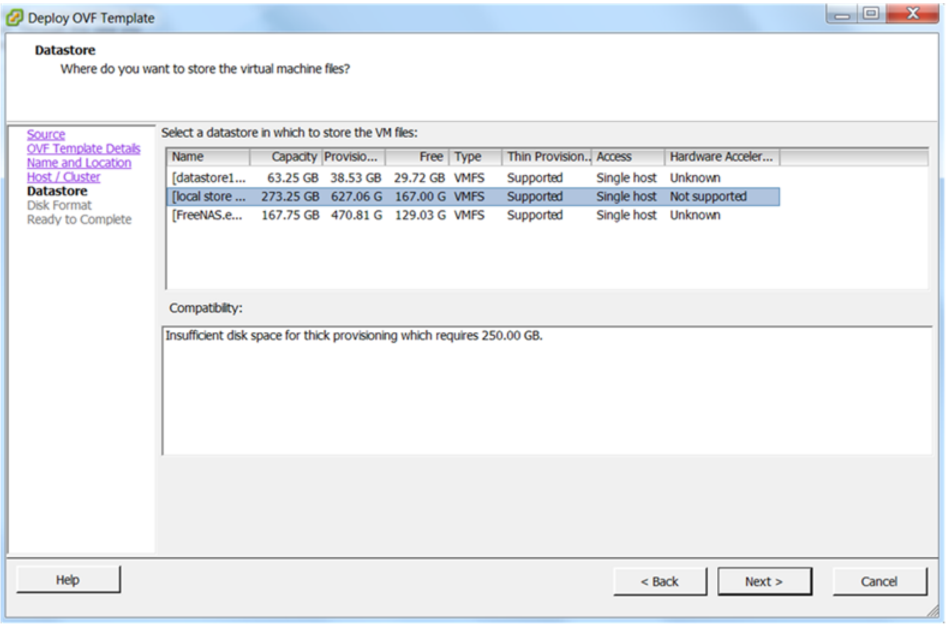
On which host or cluster do you want to run the deployed template?

Utopia
192.168.0.50

7. Select the data store where the Connector will be stored, then click **Next**.

Note

Although the Connector virtual appliance is small, it temporarily stores the images of the virtual machines that import to Amazon EC2. Therefore, the data store must be large enough to accommodate these images, as well as the Connector. We recommend a data store size of 250GB or larger.



Datastore

Where do you want to store the virtual machine files?

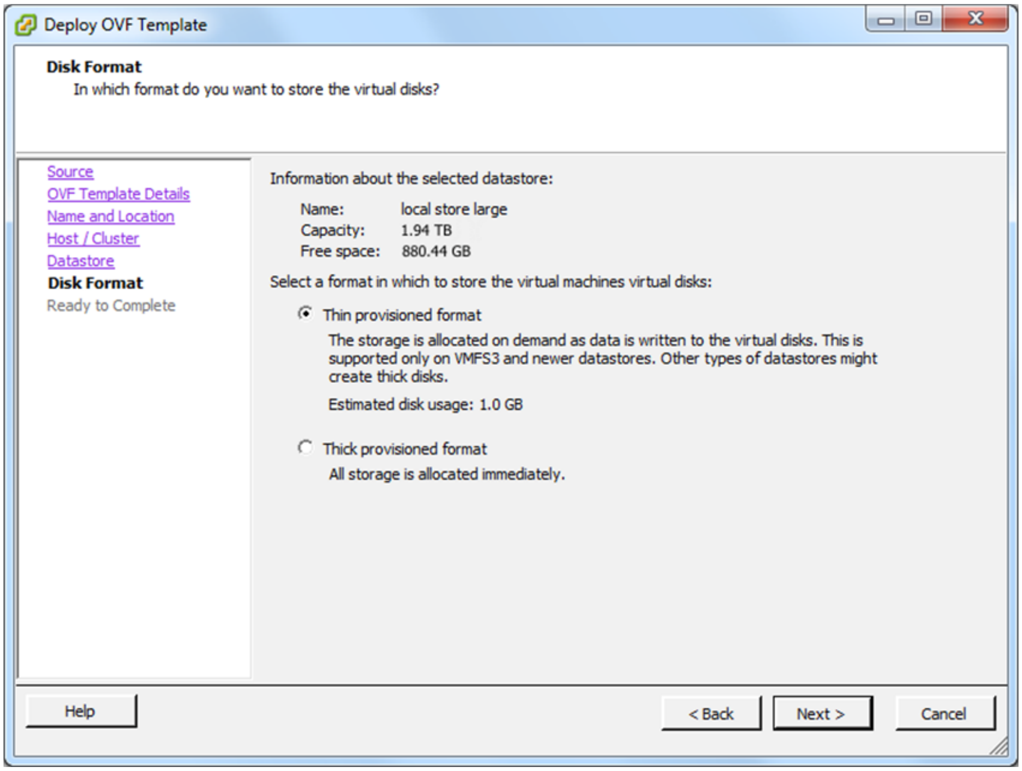
Select a datastore in which to store the VM files:

Name	Capacity	Provisio...	Free	Type	Thin Provision..	Access	Hardware Acceler...
[datastore1...	63.25 GB	38.53 GB	29.72 GB	VMFS	Supported	Single host	Unknown
[local store ...	273.25 GB	627.06 G	167.00 G	VMFS	Supported	Single host	Not supported
[FreeNAS.e...	167.75 GB	470.81 G	129.03 G	VMFS	Supported	Single host	Unknown

Compatibility:

Insufficient disk space for thick provisioning which requires 250.00 GB.

8. Select **Thin provisioned format**, then click **Next**.



Disk Format

In which format do you want to store the virtual disks?

Information about the selected datastore:

Name:	local store large
Capacity:	1.94 TB
Free space:	880.44 GB

Select a format in which to store the virtual machines virtual disks:

● Thin provisioned format

The storage is allocated on demand as data is written to the virtual disks. This is supported only on VMFS3 and newer datastores. Other types of datastores might create thick disks.

Estimated disk usage: 1.0 GB

○ Thick provisioned format

All storage is allocated immediately.

9. Confirm the details you selected and click **Finish**.

   The Connector virtual appliance will now install.

Configuring the Connector for VMware vCenter

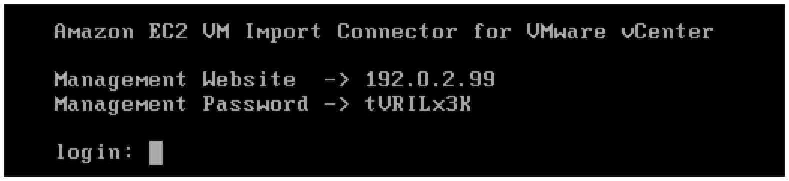
After you install the EC2 VM Import Connector, you must obtain its IP address and password and register it with the vCenter Server. To obtain the Connector's IP address and password, you first start the Connector appliance in vCenter, then go to the vCenter **Console** tab. The tab displays the IP address and password when the Connector is running. In a web browser, use this information to log in to the Connector and register it with the vCenter Server.

If the VMware vSphere Client was running when you installed the Connector virtual application, close and restart the vSphere Client, then follow these procedures.

To start the Connector for VMware vCenter

1. On the vSphere Client, right-click the Connector you just installed, select **Power** and then **Power On**.
2. To open the console, right-click the Connector you just installed and click **Open Console**.

When the Connector is running, you will find its IP address and password displayed in the vCenter **Console**. Your Connector information will be similar to the following example:



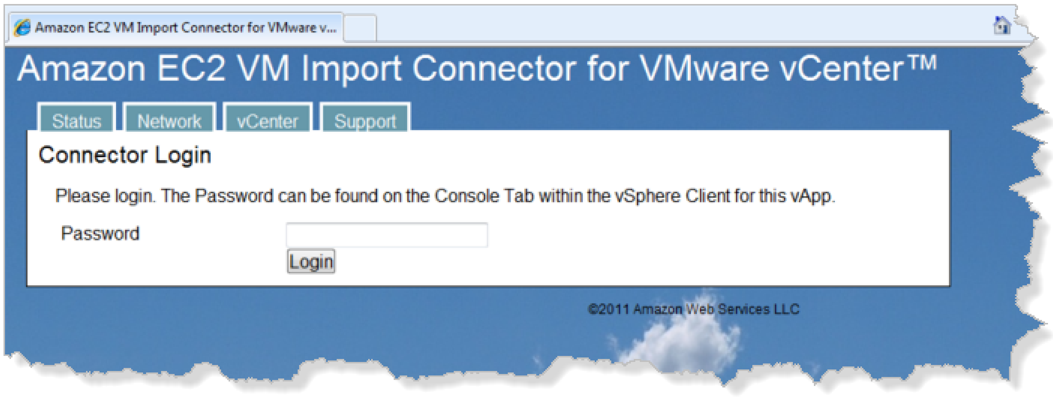
```
Amazon EC2 VM Import Connector for VMware vCenter

Management Website  -> 192.0.2.99
Management Password -> tVRILx3K

login:
```

To register the Connector with the VMware vCenter

1. Open a web browser and, in the address bar, type the Connector IP address that you obtained from the **Console** in the vCenter, and log in with the password.



Amazon EC2 VM Import Connector for VMware vCenter™

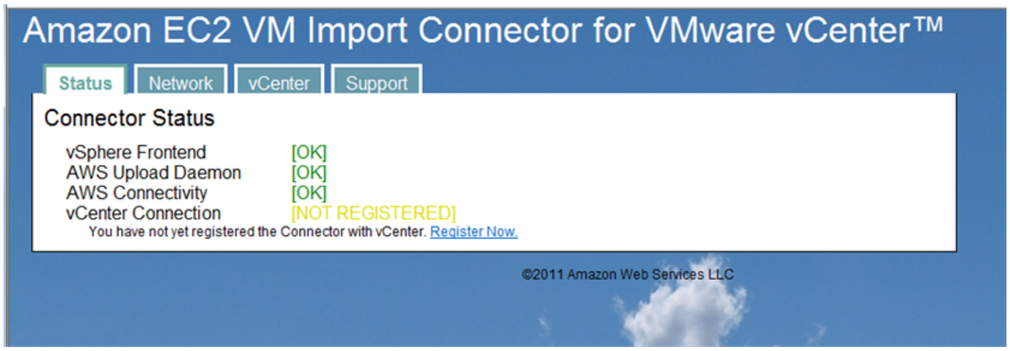
Connector Login

Please login. The Password can be found on the Console Tab within the vSphere Client for this vApp.

Password

©2011 Amazon Web Services LLC

2. When you are logged in, the browser displays Connector status information. Note that the Connector is not yet registered with the vCenter Server. Confirm that everything else in the **Connector Status** list has a status of **OK**.



Amazon EC2 VM Import Connector for VMware vCenter™

Connector Status

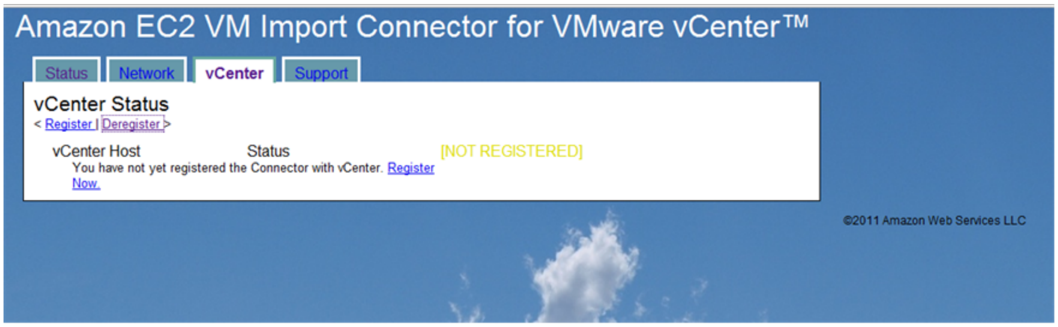
vSphere Frontend	[OK]
AWS Upload Daemon	[OK]
AWS Connectivity	[OK]
vCenter Connection	[NOT REGISTERED]

You have not yet registered the Connector with vCenter. Register Now.

©2011 Amazon Web Services LLC

3. If the version of the Connector that you are registering is not an upgrade, click **Register Now**. If you are upgrading the Connector, you must take the next two steps before registering.

   - Confirm that all import tasks are complete.
   - Deregister the old Connector from vCenter. To do this, go to the **vCenter** tab and click **Deregister**.



vCenter Status
< Register | Deregister >

vCenter Host	Status	[NOT REGISTERED]
You have not yet registered the Connector with vCenter. Register Now.

©2011 Amazon Web Services LLC

The **vCenter Connector Registration** page appears.

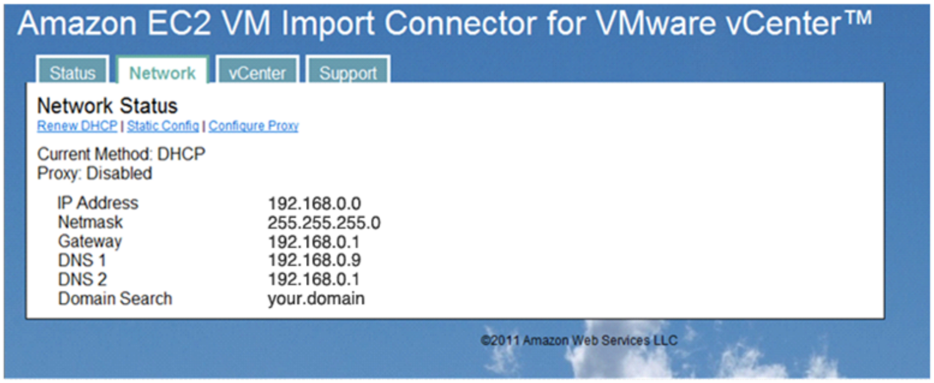
4. Provide the IP address of the vCenter with which you want to register the Connector. The user name and password you use must have administrative rights. Click **Register Connector with vCenter**.

> **Note**
>
> If an error occurs, check that the VMware vCenter IP address or name that you provided is correct. Also, confirm that the Connector virtual appliance has network access to TCP port 443 on your vCenter Server. If the user name or password you provided is incorrect, you will see a description of this error.

To configure a proxy

If you need to configure a proxy to allow the Connector to reach the Internet, you can do it using the Connector's web interface.

1. Go to the **Network** tab, click **Configure Proxy**.



Network Status
Renew DHCP | Static Config | Configure Proxy

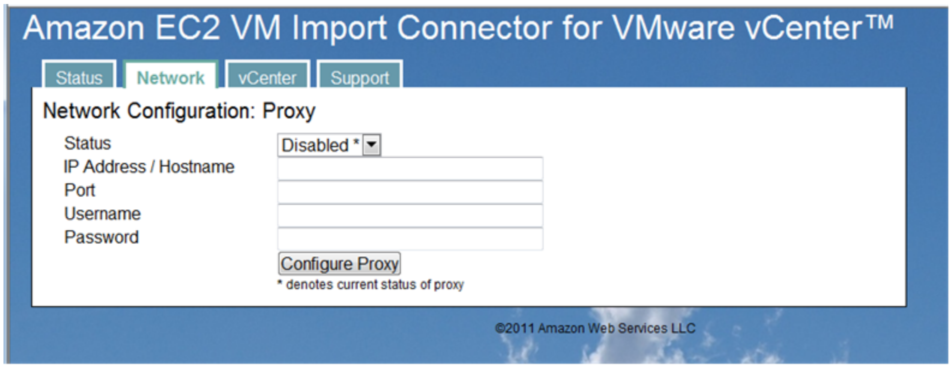
Current Method: DHCP
Proxy: Disabled

IP Address	192.168.0.0
Netmask	255.255.255.0
Gateway	192.168.0.1
DNS 1	192.168.0.9
DNS 2	192.168.0.1
Domain Search	your.domain

©2011 Amazon Web Services LLC

The **Network Configuration:Proxy** page appears.

2. Provide the information required and click **Configure Proxy**.


Amazon EC2 VM Import Connector for VMware vCenter™
Status
Network
vCenter
Support
Configure Proxy
©2011 Amazon Web Services LLC

Network Configuration: Proxy

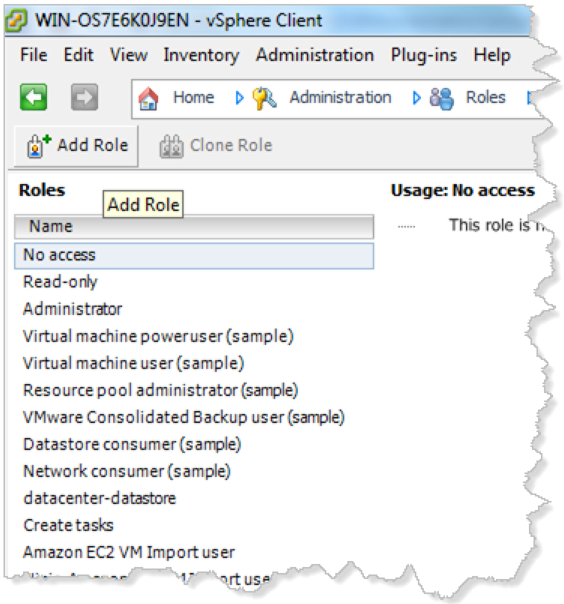
Status	Disabled *
IP Address / Hostname	
Port	
Username	
Password	

* denotes current status of proxy

After using the web browser to register the Connector with the vCenter Server, as an administrator you can import virtual machines to Amazon EC2. If you're going to import VMs using only an administrator account, skip the following section and go to Using the Connector for VMware vCenter (p. 208). If you want non-administrative users to import VMs to Amazon EC2, you must grant them permission using the vCenter.

To grant permission to non-administrative users to import VMs to Amazon EC2

1. Log in to vCenter as Administrator and from **Home**, navigate to **Roles**, and click **Add Role**.



Roles

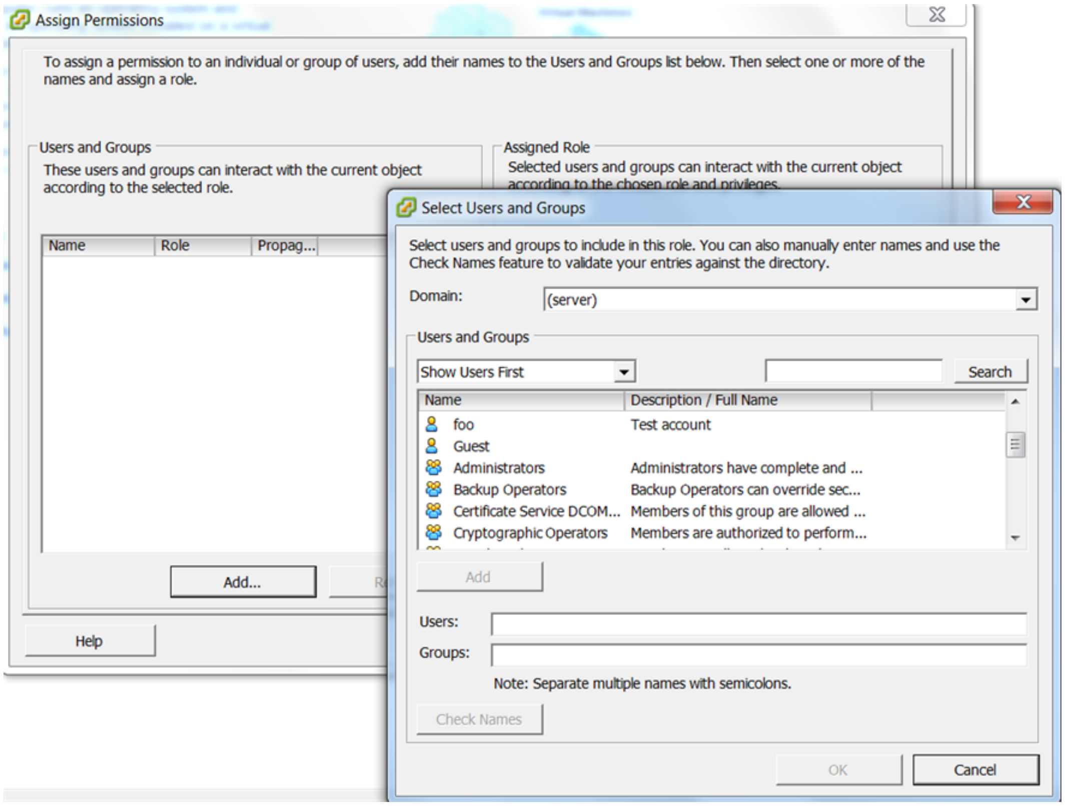
Name
No access
Read-only
Administrator
Virtual machine power user (sample)
Virtual machine user (sample)
Resource pool administrator (sample)
VMware Consolidated Backup user (sample)
Datastore consumer (sample)
Network consumer (sample)
datacenter-datastore
Create tasks
Amazon EC2 VM Import user

2. In the **Add New Role** dialog box, type the name for the new role and specify the following permissions.

   - Under **Global**, select **Cancel task**.
   - Under **Tasks**, select **Create task** and **Update task**.
   - Under **vApp**, select **Export** and **View OVF Environment**.

   Click **OK**.

3. To grant the new role permission to vCenter users or groups who will be importing virtual machines to Amazon EC2, right-click your specific vCenter in the tree-view pane and select **Add permission**. The **Assign Permissions** dialog box opens.
4. In the **Users and Groups** box on the left, select the users or groups you want to add to the new role you created.



To assign a permission to an individual or group of users, add their names to the Users and Groups list below. Then select one or more of the names and assign a role.

These users and groups can interact with the current object according to the selected role.

Selected users and groups can interact with the current object according to the chosen role and privileges.

Select users and groups to include in this role. You can also manually enter names and use the Check Names feature to validate your entries against the directory.

Domain: (server)

Show Users First

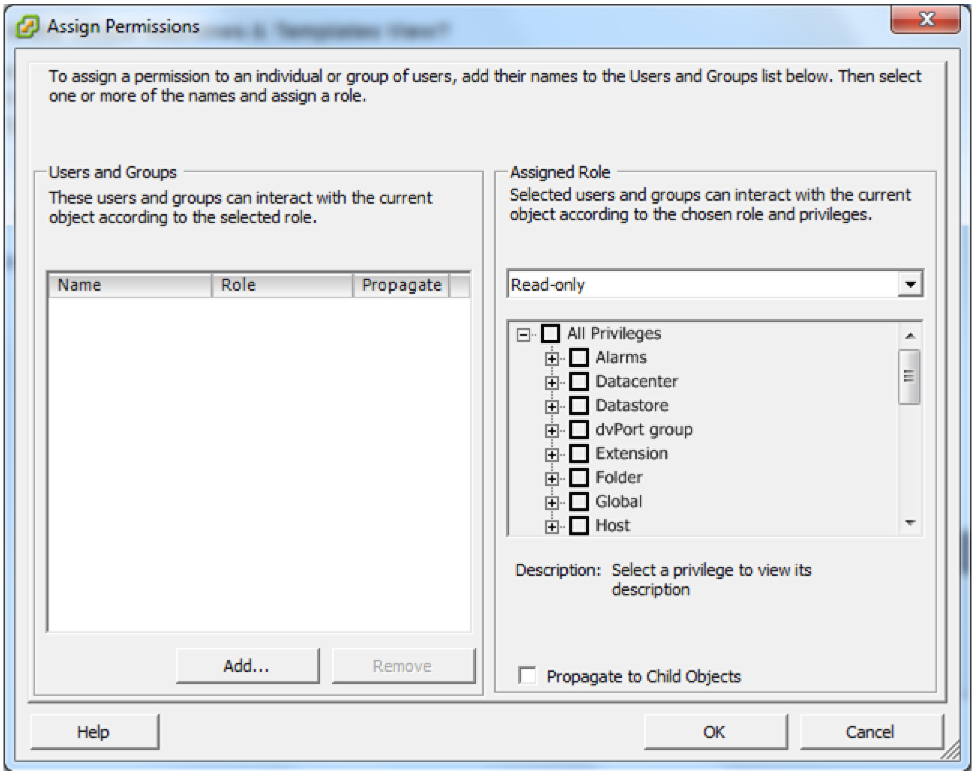
Name	Description / Full Name
foo	Test account
Guest	
Administrators	Administrators have complete and ...
Backup Operators	Backup Operators can override sec...
Certificate Service DCOM...	Members of this group are allowed ...
Cryptographic Operators	Members are authorized to perform...

Note: Separate multiple names with semicolons.

If you don't have users defined yet, click **Add**. The **Select Users and Groups** dialog box opens.

5. Select and add the users you want to add to the new role. When you have identified all the users for the role, click **OK**.
6. In the **Assigned Role** box on the right of the **Assign Permissions** dialog box, select the role that you previously created.
7. Clear the check box for **Propagate to Child Objects** and click **OK**.

   Repeat the same process to assign the role of virtual machine power user to all users and groups that you want to allow to import VMs to Amazon EC2. You can do this at the VM object level or at a higher level in the hierarchy. If you assign the role at a higher level, you must select the check box for **Propagate to Child Objects**.



To assign a permission to an individual or group of users, add their names to the Users and Groups list below. Then select one or more of the names and assign a role.

Users and Groups: These users and groups can interact with the current object according to the selected role.

Assigned Role: Selected users and groups can interact with the current object according to the chosen role and privileges.

Assigned Role: Read-only

Description: Select a privilege to view its description

□ Propagate to Child Objects

Using the Connector for VMware vCenter

This section shows you how to use the Connector to import a virtual machine to Amazon EC2 for the first time using an account with administrative rights.

Confirm that you have prepared the virtual machine according to the guidelines in Preparing Your Virtual Machine (p. 196).

Important

If you don't enable RDP and disable the Windows-based firewall, your VM will import successfully to Amazon EC2, but you will not be able to log in.

These requirements must be satisfied:

- The virtual machine must be turned off.
- The virtual machine must only use a single virtual hard drive (multiple partitions are OK).
- The virtual hard drive cannot be larger than one terabyte (1TB).
- The Connector virtual appliance must have sufficient free hard drive space to temporarily store the compressed VMDK image while it is being imported to Amazon EC2.

To use the Connector to import a VM for the first time using an account with administrative rights

1. Log in to VMware vCenter using the VMware vSphere Client. If you had a session open while you were installing the Connector, notice that the **Import to EC2** tab becomes visible.

Note

The first time you log in, you will see an SSL certificate warning. This warning indicates that the SSL certificate being used by the Connector cannot be verified by an external source. This is expected behavior. Your session will continue to use SSL for encryption. Check the **Install this certificate and do not display a security warning** option at the bottom of the screen and click **Ignore**.

You are now logged in to the vCenter Server.

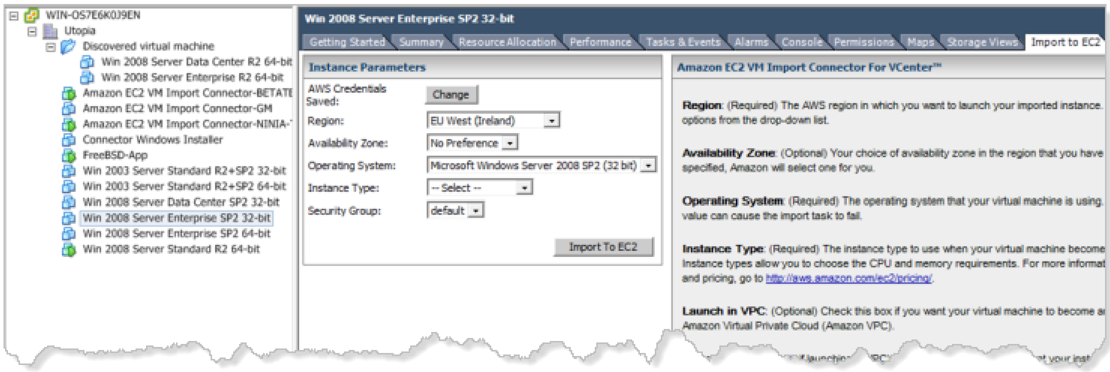
2. On the left pane, navigate the tree view to the virtual machine you want to import. Select it.
3. On the right pane, select the **Import to EC2** tab.

   Note

   You might see another security warning about SSL certificates. Click **Yes** to continue.

   The Connector initializes.
4. In the **Enter AWS Credentials** dialog box, provide your **Access Key ID** and **Secret Access Key**. For information on how to get your AWS credentials, see How to Get Your Access Key ID and Secret Access Key (p. 267).

   The main Connector page appears when your AWS credentials are verified.
5. Back in vCenter, select the virtual machine you want to import and go to the **Import to EC2** tab.


WIN-OS7E6K0J9EN
Utopia
Discovered virtual machine
Win 2008 Server Data Center R2 64-bit
Win 2008 Server Enterprise R2 64-bit
Amazon EC2 VM Import Connector-BETATE
Amazon EC2 VM Import Connector-GM
Amazon EC2 VM Import Connector-NINIA-
Connector Windows Installer
FreeBSD-App
Win 2003 Server Standard R2+SP2 32-bit
Win 2003 Server Standard R2+SP2 64-bit
Win 2008 Server Data Center SP2 32-bit
Win 2008 Server Enterprise SP2 32-bit
Win 2008 Server Enterprise SP2 64-bit
Win 2008 Server Standard R2 64-bit
Win 2008 Server Enterprise SP2 32-bit
Getting Started
Summary
Resource Allocation
Performance
Tasks & Events
Alarms
Console
Permissions
Maps
Storage Views
Import to EC2
Instance Parameters
Change
Import To EC2
Amazon EC2 VM Import Connector For VCenter™

AWS Credentials Saved:
Region: EU West (Ireland)
Availability Zone: No Preference
Operating System: Microsoft Windows Server 2008 SP2 (32 bit)
Instance Type: -- Select --
Security Group: default

Region: (Required) The AWS region in which you want to launch your imported instance. options from the drop-down list.

Availability Zone: (Optional) Your choice of availability zone in the region that you have specified, Amazon will select one for you.

Operating System: (Required) The operating system that your virtual machine is using. value can cause the import task to fail.

Instance Type: (Required) The instance type to use when your virtual machine become Instance types allow you to choose the CPU and memory requirements. For more informat and pricing, go to http://aws.amazon.com/ec2/pricing/.

Launch in VPC: (Optional) Check this box if you want your virtual machine to become a Amazon Virtual Private Cloud (Amazon VPC).

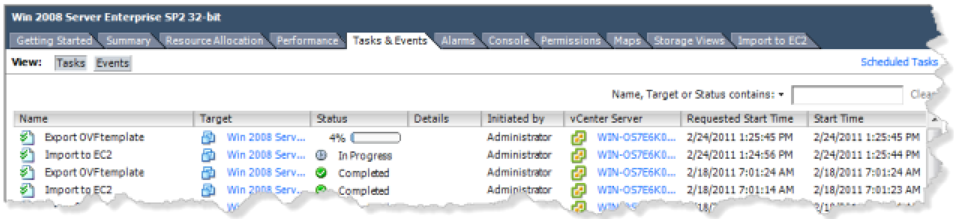
6. In the **Instance Parameters** dialog box, specify the values for the following options, then click the **Import to EC2** button.
   - **Region—**(Required)The AWS Region in which you want to launch your imported instance. Select one of the options from the drop-down list.
   - **Availability Zone—**(Optional) Your choice of Availability Zone in the Region that you have selected. If not specified, Amazon will select one for you.
   - **Operating System—**(Required) The operating system that your virtual machine is using. Selecting an incorrect value can cause the import task to fail.

     For virtual machines using Windows Server 2008 R2, always select **Microsoft Windows Server 2008 (64-bit)**. If the virtual machine you're importing runs on Windows Server 2008 SP2 (also called Windows Server 2008 R1), determine first whether the operating system is a 32-bit or 64-bit **System Type** then select the **Operating System** accordingly. For more information about 32-bit and 64-bit Windows, go to 32-bit and 64-bit Windows: frequently asked questions. For more information about how to determine System Type for Windows Server 2003, go to Microsoft Support.

- **Instance Type—**(Required) The instance type to use when your virtual machine becomes an instance in EC2. Instance types allow you to choose the CPU and memory requirements. For more information on instance types and pricing, go to Amazon EC2 Pricing.
- **Launch in VPC—**(Optional) Check this box if you want your virtual machine to become an instance within Amazon Virtual Private Cloud (Amazon VPC). For information, go to Amazon Virtual Private Cloud (Amazon VPC).
- **Subnet—**(Required only if launching in VPC) Select the subnet that you want your instance placed within in a VPC.
- **Private IP address—**(Optional, applies only if launching in VPC) Specify the private IP address of your instance within VPC.
- **Security Group—**(Required) Select the security group to use with your instance. Defaults to the default security group.

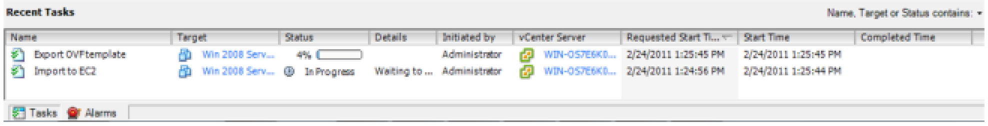
The values you specified are listed in the **Confirm Import Options** box. Check the information and click **Import**.

7. Monitor the progress of the import task in the **Tasks & Events** or **Recent Tasks** tab of the VMware vSphere Client.



Win 2008 Server Enterprise SP2 32-bit

Name	Target	Status	Details	Initiated by	vCenter Server	Requested Start Time	Start Time
Export OVFtemplate	Win 2008 Serv...	4%		Administrator	WIN-OS7E6K0...	2/24/2011 1:25:45 PM	2/24/2011 1:25:45 PM
Import to EC2	Win 2008 Serv...	In Progress		Administrator	WIN-OS7E6K0...	2/24/2011 1:24:56 PM	2/24/2011 1:25:44 PM
Export OVFtemplate	Win 2008 Serv...	Completed		Administrator	WIN-OS7E6K0...	2/18/2011 7:01:24 AM	2/18/2011 7:01:24 AM
Import to EC2	Win 2008 Serv...	Completed		Administrator	WIN-OS7E6K0...	2/18/2011 7:01:14 AM	2/18/2011 7:01:23 AM



Recent Tasks

Name	Target	Status	Details	Initiated by	vCenter Server	Requested Start Ti...	Start Time	Completed Time
Export OVFtemplate	Win 2008 Serv...	4%		Administrator	WIN-OS7E6K0...	2/24/2011 1:25:45 PM	2/24/2011 1:25:45 PM	
Import to EC2	Win 2008 Serv...	In Progress	Waiting to ...	Administrator	WIN-OS7E6K0...	2/24/2011 1:24:56 PM	2/24/2011 1:25:44 PM	

- **Export OVF template—**Creates a stream-optimized VMDK image. This process consolidates your virtual machine to a single image. In addition, stream-optimized VMDKs are compressed and are well-suited for transfer over a WAN connection. The stream-optimized VMDK will be temporarily stored on your Connector virtual appliance.
- **Import to EC2—**Transfers the stream-optimized VMDK that was created in the first task to Amazon EC2, and converts your virtual machine to an Amazon EC2 instance.

The **Import to EC2** task can take up to a few hours to complete. In addition, you might notice that the task progress will pause for up to 10 minutes at times. This is expected behavior.

Important

Keep your session in VMware vCenter open until all tasks complete. If you quit the vSphere Client or log off of your vCenter, the import task will not complete successfully, and the progress indicator will not be updated. If this occurs, you can use the command line tools to check the status of your import task. For more information, see Checking on the Status of Your Import in Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2 (p. 211). When you have verified that the task is complete, you can safely cancel the import task by right-clicking the task in the vSphere Client and selecting **Cancel**.

Although you can see your instance in the AWS Management Console when the import process begins, do not launch your instance until the import process completes. For information about launching instances, see Running an Instance (p. 85).

Uninstalling the Connector for VMware vCenter

If you no longer want to use the Connector for VMware vCenter and you want to uninstall the virtual appliance, you will follow a two-part process:

- Deregister the Connector from the vCenter Server.
- Shut down the virtual appliance.

To uninstall the Connector from the VMware vCenter

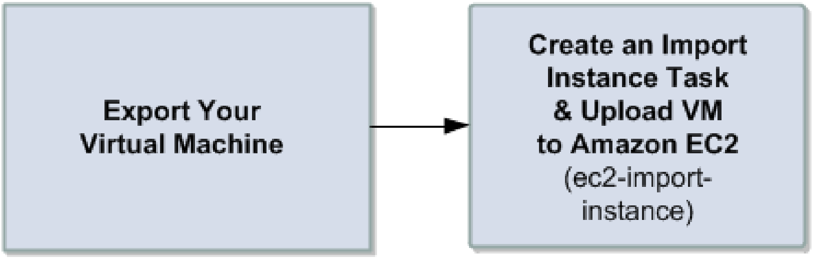
1. Open a web connection to the Connector's IP address and log in.
2. Click the **vCenter** tab, then click **Deregister**.
3. In the **vCenter Connector Deregistration** page, enter the vCenter IP information and user name and password, then click **Deregister Connector with vCenter**.
4. When the Connector is no longer registered with the vCenter Server, shut down the virtual appliance and remove it from vCenter.

Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2

Topics

In this section, you'll learn how to use the Amazon EC2 command line tools to import your Citrix, Microsoft Hyper-V, or VMware virtual machine to Amazon EC2.

Importing VMs into Amazon EC2 is a two-step process. First, you export your virtual machine from the virtualization environment. Next, you create an import task and upload your virtual machine into Amazon EC2. The following diagram illustrates this process.


Export Your Virtual Machine
Create an Import Instance Task & Upload VM to Amazon EC2 (ec2-import-instance)

Use the following commands when you perform import tasks using the Amazon EC2 command line tools:

Command	Description
`ec2-import-instance`	Creates a new import instance task using metadata from the specified disk image and imports the instance to Amazon EC2.

Command	Description
`ec2-import-volume`	Creates a new import volume task using metadata from the specified disk image and imports the volume to Amazon EC2.
`ec2-resume-import`	Resumes the upload of a disk image associated with an import instance or import volume task ID.
`ec2-describe-conversion-tasks`	Lists and describes your conversion tasks.
`ec2-cancel-conversion-task`	Cancels the active conversion task. The task can be the import of an instance or volume.
`ec2-delete-disk-image`	Deletes a partially or fully uploaded disk image for conversion from Amazon S3.

For information about these commands and other EC2 commands, go to the Amazon Elastic Compute Cloud Command Line Reference.

Note

You can use a graphical user interface provided through the Amazon EC2 VM Import Connector (Connector) to import your VMware virtual machines to Amazon EC2. For more information, see Using the Amazon EC2 VM Import Connector to Import Your Virtual Machine to Amazon EC2 (p. 197). You cannot use the Connector to import Citrix or Microsoft Hyper-V virtual machines.

Exporting Your Virtual Machines from Their Virtual Environment

The process of exporting virtual machines depends on the source virtualization environment. In this section, we will show you the basic steps to export virtual machines from Citrix, Microsoft Hyper-V, and VMware virtualization products. For in-depth information, consult the documentation for these products.

- Exporting from Citrix (p. 212)
- Exporting from Microsoft Hyper-V (p. 215)
- Exporting from VMware (p. 217)

Before starting the export process, prepare the Windows Server environment of the virtual machine you are exporting so that:

- Remote desktop is enabled.
- Windows firewall allows public RDP traffic.
- Autologon is disabled.
- There are no pending Microsoft updates and the computer is not set to install software when it reboots.

Exporting from Citrix

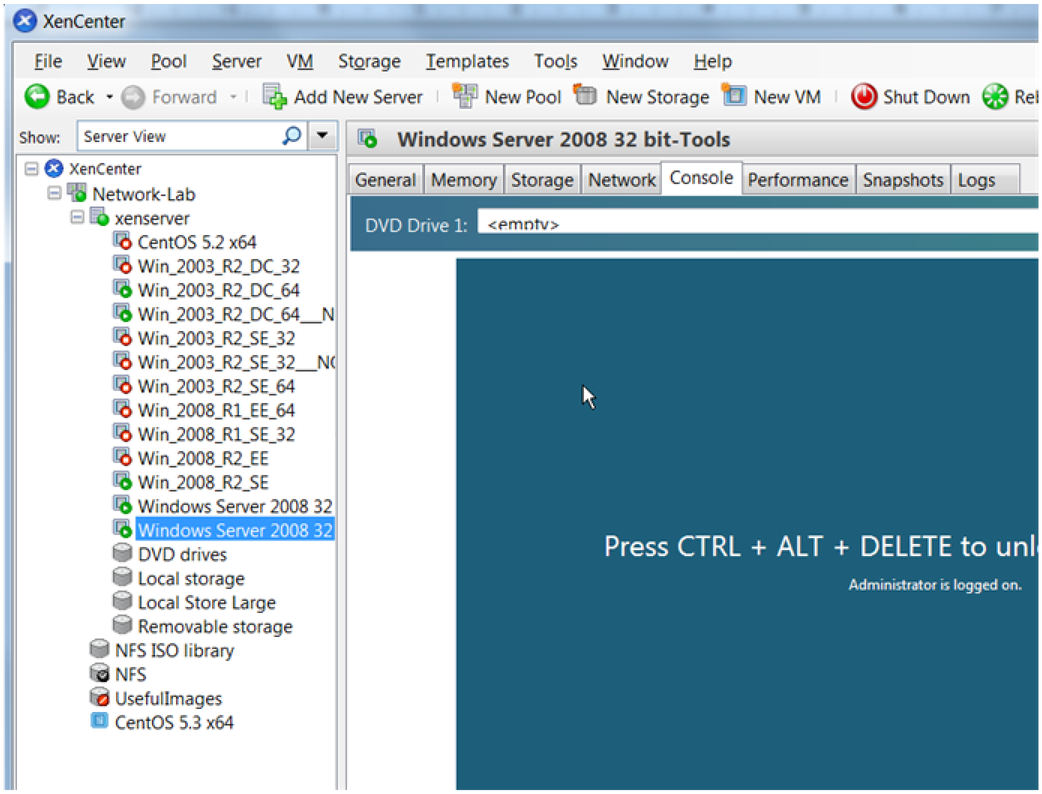
Before you export your virtual machines from Citrix XenCenter, you must perform the following tasks:

- Remove the **Citrix Tools for Virtual Machines** from the VM.
- Use the Citrix XenCenter console to remove the tools.

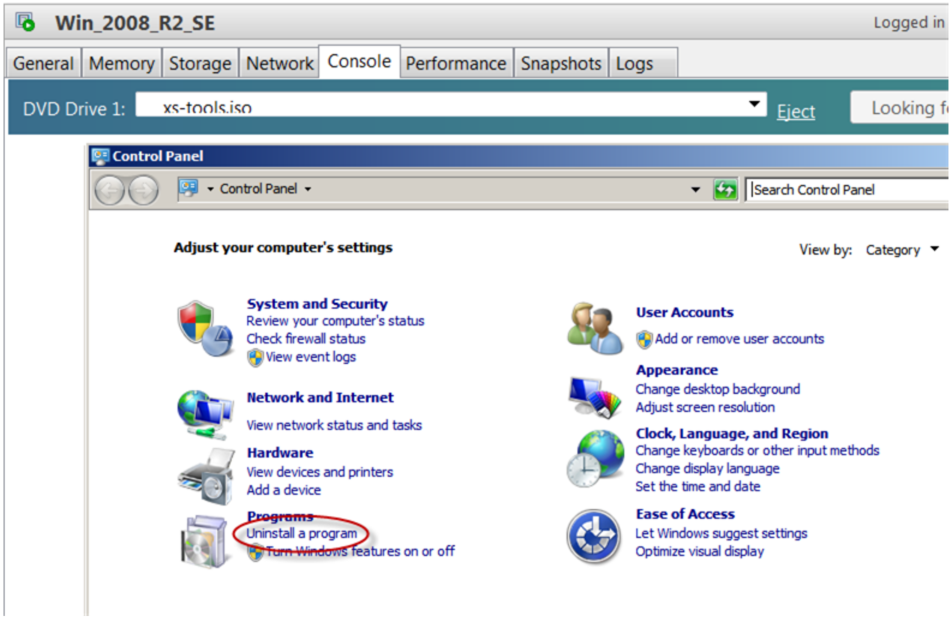
If you have multiple virtual disks that you want to export from Citrix Xen, we recommend that you export the disks one at a time. The export of multiple virtual disks at the same time results in a list of randomly named VHD files.

To remove Citrix tools for virtual machines

1. In **XenCenter**, select the virtual machine you want to export, and click the **Console** tab, which shows the Windows desktop of the virtual machine.



2. Using the **Console**, access the Control Panel of the virtual machine's Windows operating system and uninstall the **Citrix Tools for Virtual Machines**.



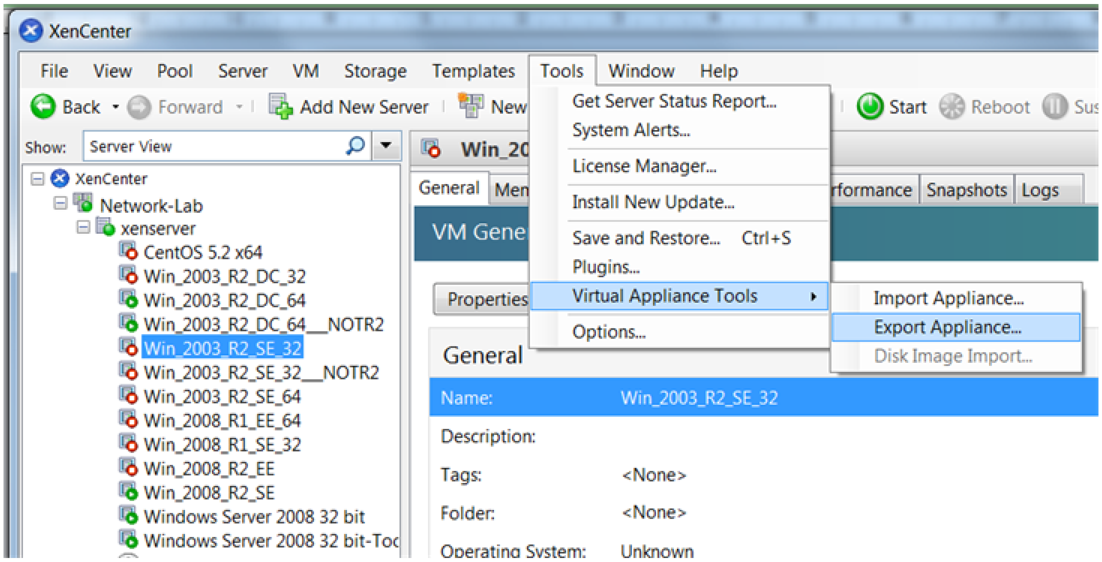
3. After the tools are removed, reboot the virtual machine when prompted, log in again and then shut down using Windows.
   You can now proceed to export the VM.

To export an image from Citrix

1. In the Citrix XenCenter, select the virtual machine you want to export, and then go to the **Tools** menu, click **Virtual Appliance Tools**, and then **Export Appliance**.

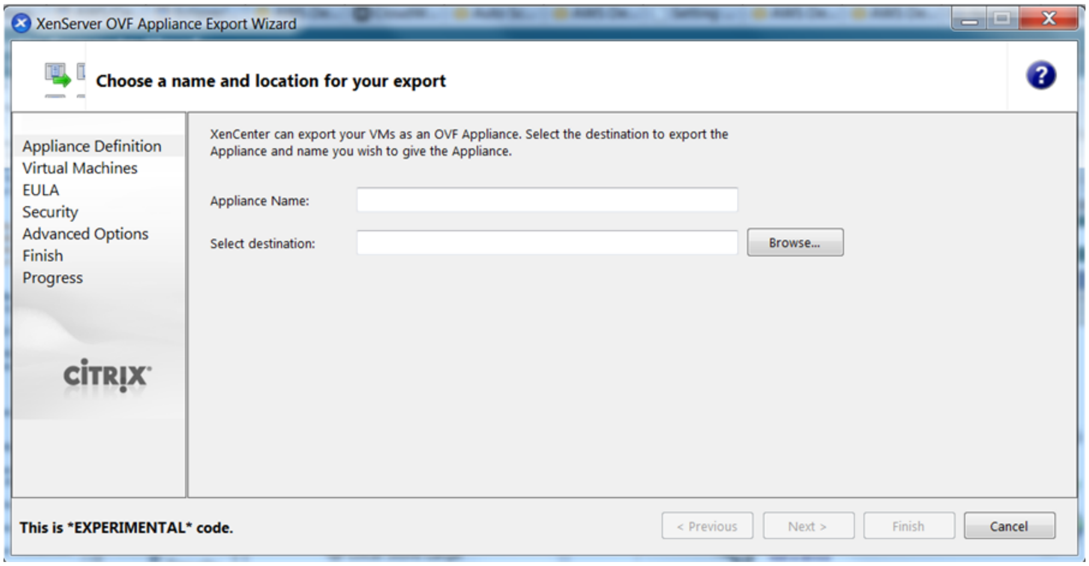
   Note

   Do not export the image by right-clicking a stopped instance and selecting "Export to File."



Name: Win_2003_R2_SE_32
Description:
Tags: <None>
Folder: <None>
Operating System: Unknown

2. When the **XenServer OVF Appliance Export Wizard** starts, specify the destination of the VM files, click **Next**, accept the defaults, and proceed through the screens until you click **Finish**.



XenCenter can export your VMs as an OVF Appliance. Select the destination to export the Appliance and name you wish to give the Appliance.

Appliance Name:

Select destination:

This is *EXPERIMENTAL* code.

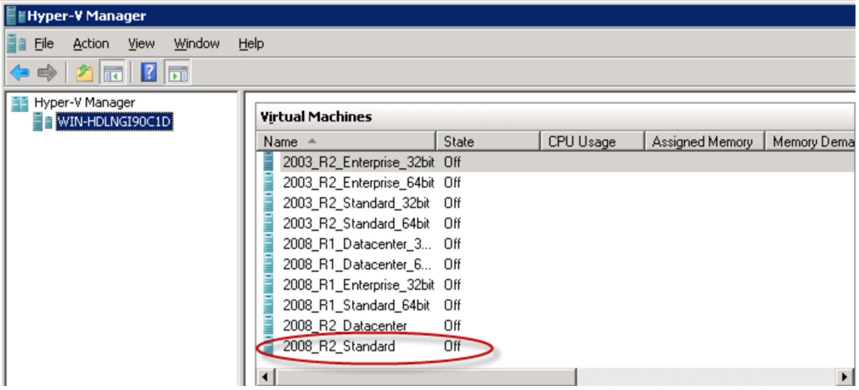
3. When the export completes, you can proceed and import the VM files to Amazon EC2 by following the steps in Importing Your Virtual Machine into Amazon EC2 (p. 218) and specifying `VHD` as the file format.

Exporting from Microsoft Hyper-V

To export Hyper-V virtual disks from Microsoft, you use the Hyper-V Manager.

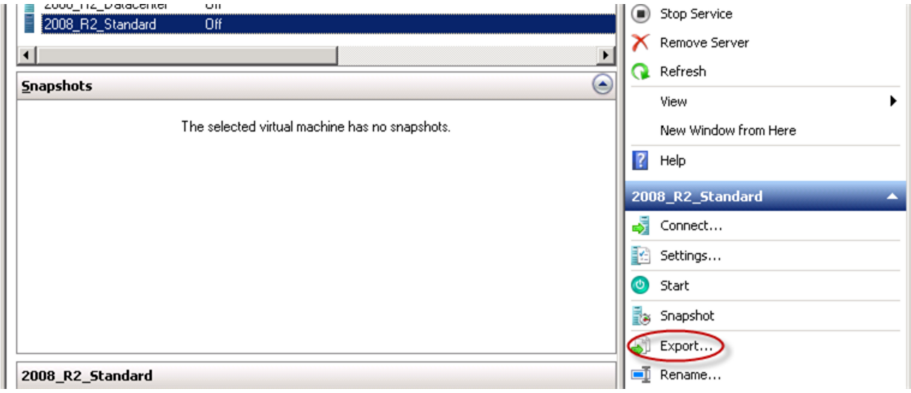
To export a Hyper-V image from Microsoft

1. In the **Hyper-V Manager**, shut down the virtual machine you want to export.



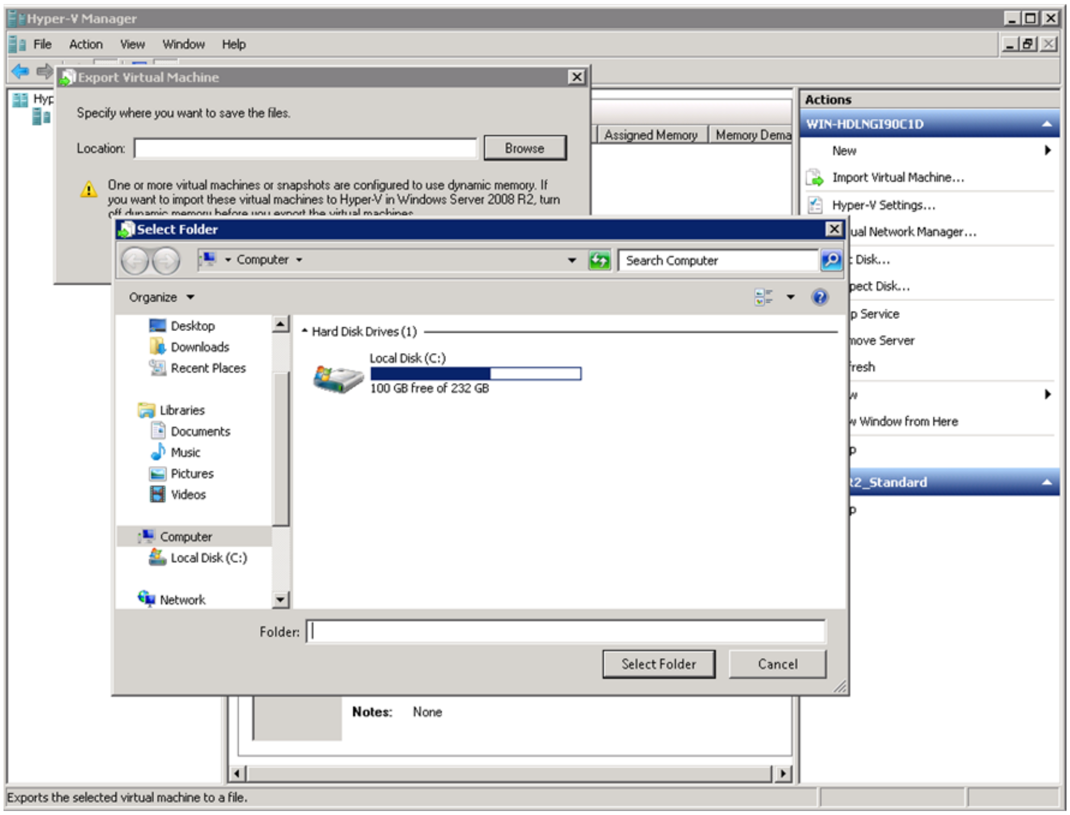
Name	State	CPU Usage	Assigned Memory	Memory Dema
2003_R2_Enterprise_32bit	Off			
2003_R2_Enterprise_64bit	Off			
2003_R2_Standard_32bit	Off			
2003_R2_Standard_64bit	Off			
2008_R1_Datacenter_3...	Off			
2008_R1_Datacenter_6...	Off			
2008_R1_Enterprise_32bit	Off			
2008_R1_Standard_64bit	Off			
2008_R2_Datacenter	Off			
2008_R2_Standard	Off			

2. In the **Actions** pane for the virtual machine, select **Export**.



The selected virtual machine has no snapshots.

3. In the **Export Virtual Machine** dialog box, for **Location**, click **Browse**, navigate to a destination location that has plenty of space, and click **Export**.



Specify where you want to save the files.

Location:

One or more virtual machines or snapshots are configured to use dynamic memory. If you want to import these virtual machines to Hyper-V in Windows Server 2008 R2, turn off dynamic memory before you export the virtual machines.

Local Disk (C:) 100 GB free of 232 GB

Folder:

Notes: None

Exports the selected virtual machine to a file.

4. Track the export progress through the **Status** of your VM in the Hyper-V Manager. Wait for the export to complete.

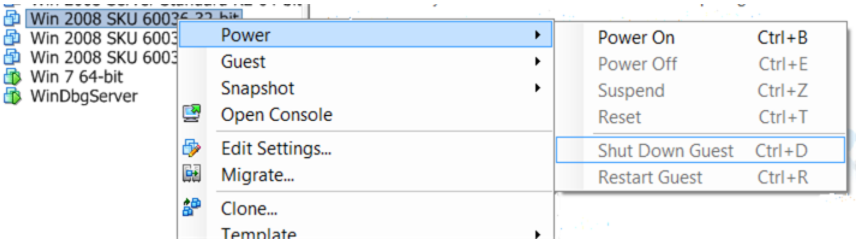
   When the export completes, you can proceed and import the VM files to Amazon EC2 by following the steps in Importing Your Virtual Machine into Amazon EC2 (p. 218) and specifying `VHD` as the file format. The VHD file will be located in the folder you specified in the **Export Virtual Machine** dialog box.

Exporting from VMware

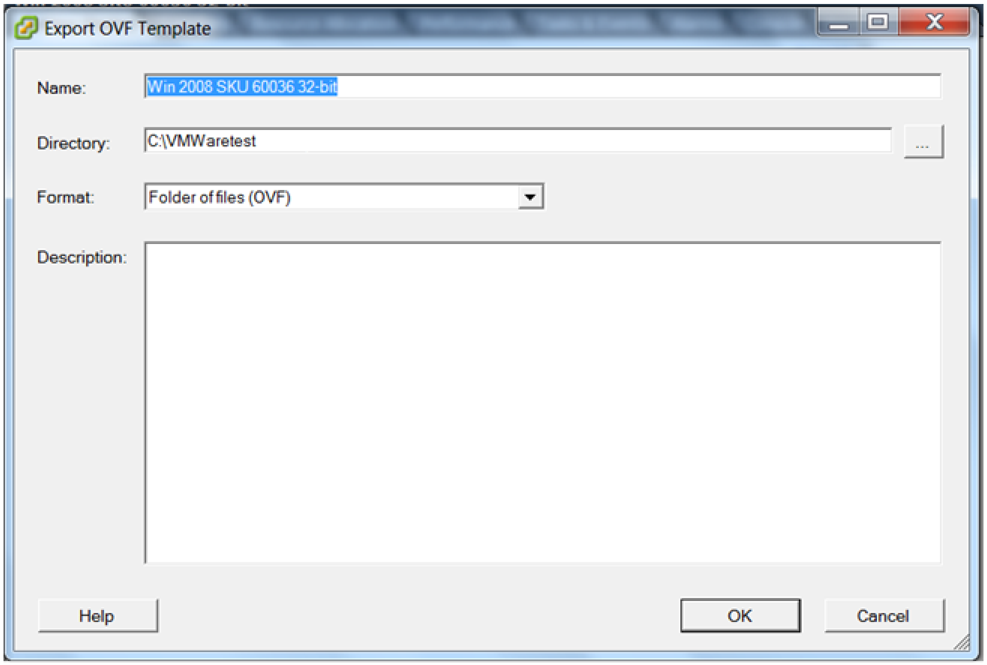
Before you can import a VMware vSphere VM or volume into Amazon EC2, you must export the VMDK disk image file. The following procedure shows you how to use the VMware vSphere Client to export a VM (and VMDK file). For more detailed information, consult your VMware documentation.

To export a disk image from VMware

1. In the VMware vSphere Client, select the virtual machine to export.
2. Shut down the machine.



3. From the **File** menu, select **Export**, and then **Export OVF Template**.


Export OVF Template

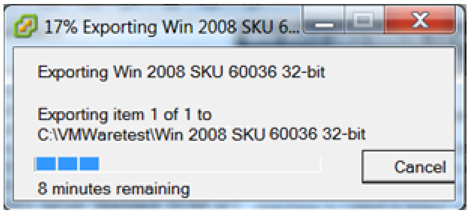
Name: Win 2008 SKU 60036 32-bit

Directory: C:\VMWaretest

Format: Folder of files (OVF)

Description:

4. In the **Export OVF Template** dialog box, enter a name for the disk image file and a directory to save it in.


17% Exporting Win 2008 SKU 6...

Exporting Win 2008 SKU 60036 32-bit

Exporting item 1 of 1 to
C:\VMWaretest\Win 2008 SKU 60036 32-bit

8 minutes remaining

You will see a box displaying the progress of the export. Because of the size and number of files associated with the VM, and the network connection, the export might take some time. If the connection times out, the export process will fail. If this happens, restart the export. When the export completes, vSphere saves the disk image file in the directory you specified. Use the name of the VMDK file as an argument in `ec2-import-instance` to import a virtual machine into Amazon EC2, or in `ec2-import-volume` to import a volume into Amazon Elastic Block Store (Amazon EBS).

Importing Your Virtual Machine into Amazon EC2

After exporting your virtual machine from the third-party virtual environment, you can import it into Amazon EC2. The import process is the same regardless of the origin of the virtual machine.

Here are some important things to know about your VM instance, as well as some security and storage recommendations:

- Amazon EC2 automatically assigns a DHCP IP address to your instance. The DNS name and IP address are available via the `ec2-describe-instances` command when the instance starts running.
- Your instance has only one Ethernet network interface.
- If you didn't specify an Amazon Virtual Private Cloud (Amazon VPC) subnet to use when you created the conversion task, your instance has a public IP address. We recommend you use a restrictive security group to control access to your instance.
- We recommend that your instance contain strong passwords for all user accounts.

To import a virtual machine

- Use `ec2-import-instance` to create a new import instance task.
  The syntax of the command is:

```
ec2-import-instance DISK_IMAGE_FILENAME -t INSTANCETYPE -f FORMAT -a ARCHI
TECTURE-SYSTEM -b S3_BUCKET_NAME -o OWNER -w SECRETKEY
```

The following command creates an import instance task that imports a Windows Server 2008 SP2 (32-bit) VM into the us-east-1 Region.

Note

The example uses the VMDK format. You can also use VHD or RAW.

```
ec2-import-instance ./WinSvr8-2-32-disk1.vmdk  –f VMDK -t m1.small -a i386
 -b MyBucket -o AKIADQKE4SARGYLE -w eW91dHViZS5jb20vd2F0Y2g/dj1SU3NKMTlzeTNK
SQ==
```

This request uses the VMDK file, `WinSvr8-2-32-disk1.vmdk`, to create the import task. The output is similar to the following example.

```
Requesting volume size: 25 GB
Disk image format: Stream-optimized VMDK
Converted volume size: 26843545600 bytes (25.00 GiB)
Requested EBS volume size: 26843545600 bytes (25.00 GiB)
TaskType       IMPORTINSTANCE  TaskId  import-i-fhbx6hua      ExpirationTime
  2011-09-09T15:03:38+00:00       Status  active  StatusMessage   Pending
InstanceID      i-6ced060c
```

```
DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   5070303744
VolumeSize      25      AvailabilityZone        us-east-1c      Approximate
BytesConverted       0       Status  active StatusMessage    Pending
Creating new manifest at testImport/9cba4345-b73e-4469-8106-
2756a9f5a077/Win_2008_R1_EE_64.vmdkmanifest.xml
Uploading the manifest file
Uploading 5070303744 bytes across 484 parts
0% |--------------------------------------------------| 100%
   |==================================================|
Done
```

Checking on the Status of Your Import

The `ec2-describe-conversion-tasks` command returns the status of an import. Status values include:

- **active—**Your instance or volume is still importing.
- **cancelling—**Your instance or volume is still being canceled.
- **cancelled—**Your instance or volume is canceled.
- **completed—**Your instance or volume is ready to use.
  The imported instance is in the stopped state. You use `ec2-start-instance` to start it. For more information, go to ec2-start-instances in the *Amazon Elastic Compute Cloud Command Line Reference*.

To check the status of your import

- Use `ec2-describe-conversion-task` to return the status of the task. The syntax of the command is:

```
ec2-describe-conversion-tasks TASKID
```

The following example enables you to see the status of your import instance task.

```
ec2-describe-conversion-tasks import-i-ffvko9js
```

The following response shows that the `IMPORTINSTANCE` status is `active`, and 73747456 bytes out of 893968896 have been converted.

```
TaskType       IMPORTINSTANCE  TaskId  import-i-ffvko9js      ExpirationTime
  2011-06-07T13:30:50+00:00       Status  active  StatusMessage   Pending
InstanceID      i-17912579
DISKIMAGE      DiskImageFormat VMDK    DiskImageSize   893968896 VolumeSize
      12      AvailabilityZone        us-east-1a      ApproximateBytesCon
verted       73747456      Status  active StatusMessage    Pending
```

The following response shows that the `IMPORTINSTANCE` status is `active`, and at 7% progress and that the DISKIMAGE is completed.

```
TaskType       IMPORTINSTANCE  TaskId  import-i-ffvko9js      ExpirationTime
  2011-06-07T13:30:50+00:00       Status  active  StatusMessage   Progress:
 7%    InstanceID      i-17912579
```

```
DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   893968896 VolumeId
        vol-9b59daf0    VolumeSize      12      AvailabilityZone        us-
east-1a      ApproximateBytesConverted      893968896 Status  completed
```

The following response shows that the `IMPORTINSTANCE` status is `completed`.

```
TaskType       IMPORTINSTANCE  TaskId  import-i-ffvko9js      ExpirationTime
  2011-06-07T13:30:50+00:00       Status  completed    InstanceID      i-
17912579
DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   893968896 VolumeId
        vol-9b59daf0    VolumeSize      12      AvailabilityZone        us-
east-1a      ApproximateBytesConverted      893968896 Status  completed
```

Note

The `IMPORTINSTANCE` status is what you use to determine the final status. The `DISKIMAGE` status will be `completed` for a period of time before the `IMPORTINSTANCE` status is `completed`.

You can now use commands such as `ec2-stop-instance`, `ec2-start-instance`, `ec2-reboot-instance`, and `ec2-terminate-instance` to manage your instance.

Note

By default, when you terminate an instance, Amazon EC2 does not delete the associated Amazon EBS volume. You can optionally use the `ec2-modify-instance-attribute` command to change this behavior.

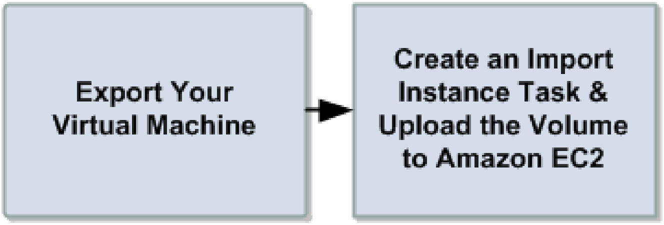
Importing Your Volumes into Amazon EBS

This section describes how to import your data storage into Amazon EBS, and then attach it to one of your existing Amazon EC2 instances. Amazon EC2 supports importing RAW and VMDK disk formats.

Important

We recommend utilizing Amazon EC2 security groups to limit network access to your imported instance. Configure a security group to allow only trusted Amazon EC2 instances and remote hosts to connect to RDP and other service ports. For more information about security groups, see Using Security Groups (p. 304).

After you have exported your virtual machine from the virtualization environment, importing the volume to Amazon EBS is a single-step process. You create an import task and upload the volume in one step, as illustrated in the following diagram.


Export Your Virtual Machine
Create an Import Instance Task & Upload the Volume to Amazon EC2

To import a volume into Amazon EBS

1. Use `ec2-import-volume` to create a task that allows you to upload your volume into Amazon EBS. The syntax of the command is:

   ```
   $ ec2-import-volume DISK_IMAGE_FILENAME -f FORMAT -s SIZE-IN-GB -z AVAILAB
   ILITY_ZONE -b S3_BUCKET_NAME -o OWNER -w SECRETKEY
   ```

   The following example creates an import volume task for importing a volume to the us-east-1a Region.

   Note

   The example uses the VMDK format. You can also use VHD or RAW.

   ```
   Requesting volume size: 25 GB
   Disk image format: Stream-optimized VMDK
   Converted volume size: 26843545600 bytes (25.00 GiB)
   Requested EBS volume size: 26843545600 bytes (25.00 GiB)
   TaskType        IMPORTVOLUME    TaskId  import-vol-ffut5xv4     ExpirationTime
     2011-09-09T15:22:30+00:00       Status  active  StatusMessage   Pending
   DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   5070303744
   VolumeSize      25      AvailabilityZone        us-east-1d      Approximate
   BytesConverted       0
   Creating new manifest at tesdfgting/0fd8fcf5-04d8-44ae-981f-
   3c9f56d04520/Win_2008_R1_EE_64.vmdkmanifest.xml
   Uploading the manifest file
   Uploading 5070303744 bytes across 484 parts
   0% |--------------------------------------------------| 100%
      |==================================================|
   Done
   ```

   Amazon EC2 returns a task ID that you use in the next step. In this example, the ID is `import-vol-ffut5xv4`.

2. Use `ec2-describe-conversion-tasks` to confirm that your volume imported successfully.

   ```
   $ ec2-describe-conversion-tasks import-vol-ffut5xv4
   TaskType        IMPORTVOLUME    TaskId  import-vol-ffut5xv4     ExpirationTime
     2011-09-09T15:22:30+00:00       Status  completed
   DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   5070303744
   VolumeId        vol-365a385c    VolumeSize      25      AvailabilityZone
         us-east-1d      ApproximateBytesConverted       5070303744
   ```

   The status in this example is `completed`, which means the import succeeded.

3. Use `ec2-attach-volume` to attach the Amazon EBS volume to one of your existing Amazon EC2 instances. The following example attaches the volume, `vol-2540994c`, to the `i-a149ec4a` instance on the device, `/dev/sde`.

   ```
   $ ec2-attach-volume vol-2540994c -i i-a149ec4a -d /dev/sde
   ATTACHMENT vol-2540994c i-a149ec4a /dev/sde attaching 2010-03-
   23T15:43:46+00:00
   ```

Restarting an Upload

Connectivity problems can interrupt an upload. When you restart an upload, Amazon EC2 automatically starts the upload from where it stopped. The following procedure steps you through determining how much of an upload succeeded and how to restart it.

To restart an upload

- Use the task ID with `ec2-resume-import` to continue the upload. The command uses the HTTP `HEAD` action to determine where to restart.

  ```
  ec2-resume-import DISK_IMAGE_FILENAME -t TASK_ID -o OWNER -w SECRETKEY
  ```

  The following example resumes an import instance task.

  ```
  Disk image size: 5070303744 bytes (4.72 GiB)
  Disk image format: Stream-optimized VMDK
  Converted volume size: 26843545600 bytes (25.00 GiB)
  Requested EBS volume size: 26843545600 bytes (25.00 GiB)
  Uploading 5070303744 bytes across 484 parts
  0% |--------------------------------------------------| 100%
     |==================================================|
  Done
  Average speed was 10.316 MBps
  The disk image for import-i-ffni8aei has been uploaded to Amazon S3
  where it is being converted into an EC2 instance.  You may monitor the
  progress of this task by running ec2-describe-conversion-tasks.  When
  the task is completed, you may use ec2-delete-disk-image to remove the
  image from S3.
  ```

Canceling an Upload

Use `ec2-cancel-conversion-task` to cancel an active conversion task. The task can be the upload of an instance or a volume. The command removes all artifacts of the conversion, including uploaded volumes or instances.

If the conversion is complete or still transferring the final disk image, the command fails and returns an exception similar to the following:

```
Client.CancelConversionTask Error: Failed to cancel conversion task import-i-
fh95npoc
```

To cancel a conversion task

- Use the task ID of the upload you want to delete with `ec2-cancel-conversion-task`. The following example cancels the upload associated with the task ID `import-i-fh95npoc`.

  ```
  PROMPT> ec2-cancel-conversion-task import-i-fh95npoc
  ```

  The output for a successful cancellation is similar to the following:

  ```
  CONVERSION-TASK import-i-fh95npoc
  ```

You can use the `ec2-describe-conversion-tasks` command to check the status of the cancellation. For example:

```
$ ./ec2-describe-conversion-tasks import-i-fh95npoc
TaskType        IMPORTINSTANCE  TaskId  import-i-fh95npoc       ExpirationTime
  2010-12-20T18:36:39+00:00      Status  cancelled   InstanceID     i-825063ef
DISKIMAGE       DiskImageFormat VMDK    DiskImageSize   2671981568
VolumeSize     40      AvailabilityZone       us-east-1c  ApproximateBytesCon
verted       0       Status  cancelled
```

In the above example, the status is *`cancelled`*. If it were still in process, the status would be *`cancelling`*.

Cleaning up After an Upload

You can use `ec2-delete-disk-image` to remove the image file after it is uploaded. If you do not delete it, you will be charged for its storage in Amazon S3.

To delete a disk image

- Use the task ID of the disk image you want to delete with `ec2-delete-disk-image`.

  The following example deletes the disk image associated with the task ID, `import-i-fh95npoc`.

  ```
  PROMPT> ec2-delete-disk-image import-i-fh95npoc
  ```

  The output for a successful cancellation is similar to the following:

  ```
  DELETE-TASK import-i-fh95npoc
  ```

Reserved Instances

Amazon EC2 Reserved Instances enable you to take advantage of the benefits of elastic computing while lowering costs and reserving capacity. With Reserved Instances, you pay a low, one-time fee and get a significant discount on the hourly fee for that instance. Reserved Instances can provide substantial savings over owning your own hardware or running only On-Demand instances. At the same time, you know that the capacity you need is available to you when you require it.

Reserved Instances are available in three varieties—Heavy Utilization, Medium Utilization, and Light Utilization—that enable you to optimize your Amazon EC2 costs based on your expected utilization. You can start lowering your costs today by matching your application's usage pattern with the pricing model that works best for you.

To purchase an Amazon EC2 Reserved Instance, you must select an instance type (such as m1.small), platform (Linux/UNIX, Windows, Windows with SQL Server, or SUSE), location (Region and Availability Zone), and term (either one year or three years). If you want to use Amazon Virtual Private Cloud (Amazon VPC), you must specify that you want your capacity reservation allocated to Amazon VPC.

If you're using Amazon VPC, you must select a platform that includes *Amazon VPC* in its name. You can also choose to purchase Reserved Instances that are physically isolated at the host hardware level by specifying *Dedicated* as the instance tenancy. For more information about Dedicated Instances, go to Using EC2 Dedicated Instances Within Your VPC in the *Amazon Virtual Private Cloud User Guide*.

These sections describe Reserved Instance basics:

- Reserved Instances Offerings (p. 224)
- Checklist for Finding, Purchasing, and Verifying Reserved Instances (p. 225)

To understand consolidated billing and how the pricing benefit of Reserved Instances is applied, see Understanding the Pricing Benefit and Consolidated Billing (p. 226).

If you want to get started with Reserved Instances, see Working with Reserved Instances (p. 231).

Reserved Instances Offerings

You can choose a Reserved Instance type based on how you plan to utilize your instance. You can pick the fee structure that fits your needs.

Starting with API version 2011-11-01, you can choose from a variety of Reserved Instance offerings that address your projected utilization of the instance: *Heavy Utilization*, *Medium Utilization*, and *Light Utilization*.

Heavy Utilization Reserved Instances enable workloads that have a consistent baseline of capacity or run steady-state workloads. Heavy Utilization Reserved Instances require the highest up-front commitment, but if you plan to run more than 79 percent of the Reserved Instance term you can earn the largest savings (up to 58 percent off of the On-Demand price). Unlike the other Reserved Instances, with Heavy Utilization Reserved Instances, you pay a one-time fee, followed by a lower hourly fee for the duration of the term *regardless* of whether or not your instance is running.

Medium Utilization Reserved Instances are the best option if you plan to leverage your Reserved Instances a substantial amount of the time, but want either a lower one-time fee or the flexibility to stop paying for your instance when you shut it off. This offering type is equivalent to the Reserved Instance offering available before API version 2011-11-01. Medium Utilization Reserved Instances are a more cost-effective option when you plan to run more than 40 percent of the Reserved Instance term. This option can save you up to 49 percent off of the On-Demand price. With Medium Utilization Reserved Instances, you pay a slightly higher one-time fee than with Light Utilization Reserved Instances, and you receive lower hourly usage rates when you run an instance.

Light Utilization Reserved Instances are ideal for periodic workloads that run only a couple of hours a day or a few days per week. Some of these use cases, such as disaster recovery, also require reserved capacity to meet potential demand without notice. Using Light Utilization Reserved Instances, you pay a one-time fee followed by a discounted hourly usage fee when your instance is running. You can start saving when your instance is running more than 17 percent of the Reserved Instance term, and you can save up to 33 percent off of the On-Demand rates over the entire term of your Reserved Instance.

Remember that discounted usage fees for Reserved Instance purchases are tied to instance type and Availability Zone. If you shut down a running EC2 instance on which you have been getting a discounted rate as a result of a Reserved Instance purchase, and the term of the Reserved Instance has not yet expired, you will continue to get the discounted rate if you launch another instance with the same specifications during the term.

The following table summarizes the differences between the Reserved Instances offering types.

Reserved Instance Offerings

Offering	Upfront Cost	Usage Fee	Advantage
Heavy Utilization	Highest	Lowest hourly fee. Applied to the whole term whether or not you're using the Reserved Instance.	Lowest overall cost if you plan to utilize your Reserved Instances more than 79 percent of a 3-year term.
Medium Utilization	Average	Hourly usage fee charged for each hour you use the instance. We encourage you to turn off your instances when you aren't using them so you won't be charged for them.	Suitable for elastic workloads or when you expect moderate usage, more than 40 percent of a 3-year term.
Light Utilization	Lowest	Hourly usage fee charged. Highest fees of all the offering types, but they apply only when you're using the Reserved Instance. We strongly encourage you to turn off your instances when you aren't using them so you won't be charged for them.	Highest overall cost if you plan to run all of the time, however lowest overall cost if you anticipate you will use your Reserved Instances infrequently, more than about 15 percent of a 3-year term.

Checklist for Finding, Purchasing, and Verifying Reserved Instances

The following list describes the process for using Reserved Instances.

1. Specify the details about the Reserved Instance you want to launch.
   - The platform on which to run your instance.
   - The instance type of the instance.

- The term (time period) over which you want to hold the instance.
- The tenancy specification, if you want to reserve capacity for your instance to run in single-tenant hardware.
- The region and Availability Zone where you want to run the instance.

2. Choose the offering type that addresses your cost and payment needs.
   - Heavy Utilization
   - Medium Utilization
   - Light Utilization

   For more information about these offering types, see Reserved Instances Offerings (p. 224).
3. Search for offerings that meet the criteria you specified.
   If you want to use a Reserved Instance with Amazon Virtual Private Cloud (Amazon VPC), make sure to search for and select an offering that includes *Amazon VPC* in the description.
4. Purchase offerings that fulfill your requirements.
5. Run instances of the purchased instance type in the correct region and Availability Zone.
6. Confirm that your instances are running as specified.

You can use the AWS Management Console (p. 232), the Command Line Tools (p. 233), or the API (p. 235) to search for offerings.

Understanding the Pricing Benefit and Consolidated Billing

Topics

When you purchase Reserved Instances, you get two benefits: a capacity reservation for a number of instances you will launch at some future time and a discounted hourly usage fee. The rest of the experience of launching and working with instances remains the same as with On-Demand instances.

This section discusses how Amazon EC2 applies the pricing benefit of Reserved Instances. You'll also see how consolidated billing works.

Applying the Pricing Benefit of Reserved Instances

With Reserved Instances, you pay an upfront fee for the capacity reservation to launch Amazon EC2 instances to the specifications you defined during the purchase. After you purchase Reserved Instances, you cannot change the specifications (product platform, instance type, Availability Zone, etc.).

In addition to the capacity reservation, you also get a discounted rate on the hourly fee for running On-Demand EC2 instances associated with the same account that purchased the Reserved Instances. For the discount to apply, the On-Demand instances must match the specifications for the Reserved Instances.

For example, let's say User A is running the following On-Demand EC2 instances:

- (4) m1.small instances in Availability Zone us-east-1a
- (4) c1.medium instances in Availability Zone us-east-1b
- (2) c1.xlarge instances in Availability Zone us-east-1b

Then User A purchases the following Medium Utilization Reserved Instances:

- (2) m1.small instances in Availability Zone us-east-1a
- (3) c1.medium instances in Availability Zone us-east-1a
- (1) c1.xlarge instance in Availability Zone us-east-1b

In purchasing the Reserved Instances, User A pays an upfront fee for the capacity reservation so he can launch the six instances to his specifications when he needs them. In addition, he is also getting a discount on the hourly usage fees for the equivalent of six instances each month. Since he already has instances running when he purchases the Reserved Instances, Amazon EC2 will automatically apply the discounted hourly rates to the already running On-Demand instances that match the specfications for the Reserved Instances he purchased.

This is what happens:

- Amazon EC2 applies the discounted usage fee rate for two m1.small Reserved Instances that he purchased to two of the four running Amazon EC2 instances in Availability Zone us-east-1a.

  The other two EC2 instances in Availability Zone us-east-1a will be charged at the current On-Demand rate.
- Amazon EC2 doesn't apply discounted rates from the three c1.medium Reserved Instances because the c1.medium Reserved Instances are in a different Availability Zone from the running c1.medium Amazon EC2 instances.

  The four running c1.medium Amazon EC2 instances will be charged at the current On-Demand rate.

  If User A launches a c1.medium EC2 instance in Availability Zone us-east-1a, Amazon EC2 will apply the Reserved Instance discounted usage fee rate to that instance.
- Amazon EC2 applies the discounted usage fee rate for one c1.xlarge Reserved Instance that he purchased to one of the two running Amazon EC2 instances in Availability Zone us-east-1b.

  The other c1.xlarge EC2 instance in Availability Zone us-east-1b will be charged at the current On-Demand rate.

In this example scenario, by purchasing the six Reserved Instances, User A saves on the hourly fee charged against two m1.small and one c1.xlarge On-Demand EC2 instances he had already running. At the same time, he is assured of the capacity to run the six instances when he needs them.

Reserved Instances and Consolidated Billing

The pricing benefits of Reserved Instances are shared when the purchasing account is part of a set of accounts billed under one Consolidated Billing Payer account. Consolidated billing allows you to pay all of your charges via one Payer account (the one that will pay the bill). The amount of hourly usage for each month across all child accounts is also aggregated in the parent account. This billing is typically useful for companies in which there are different functional teams or groups.

For Reserved Instances, the amount of usage across all linked accounts is also aggregated in the payer account, by the hour for each month, and then the normal Reserved Instance logic is applied to calculate the bill. The discount is spread across the consolidated account. The allocation of the total cost is determined by the ratio of each child account's usage divided by the total usage of the parent account. However, it is important to note that the capacity reservation remains with the purchasing account. Keep in mind that capacity reservation only applies to the product platform, instance type, and Availability Zone specified in the purchase.

If the account is part of a consolidated billing account, the upfront cost of the Reserved Instance purchases is paid by the Payer account. Additionally, all of the subsequent usage charges will be paid by the payer account as long as the account remains linked with the payer account. However, all of the pricing benefit will also be shared with all of the linked accounts as long as it remains linked to the payer account.

To illustrate, we'll expand on the previous example. For the purpose of explaining how billing works, we're simplifying the scenario with the following assumptions:

- All instances are of the same instance type, running on the same product platform in the same Availability Zone.
- The usage fee cost per hour for each instance is $1.00.
- The discounted usage fee cost per hour for each instance is $0.50.
- User A, User B, and User C, fall under account CB AZ.
- In one month, the total number of instances running across all three users is 18 instances, all at the same rate.
- There are 30 days in a month, and the instances run 10 hours a day. This is a total of 300 hours a month.
- User A, the purchaser of the six Reserved Instances, is running 9 Instances, which is one-half of the total instances billed to the master account.
- User B is running 6 Instances, which is one-third of the total instances billed.
- User C is running 3 Instances, which is one-sixth of the total instances billed.

The following table shows how the consolidated bill would look without the benefit of Reserved Instances purchased by User A.

Note

Amazon EC2 doesn't determine or allocate Reserved Instances usage to a particular account. Instead, Amazon EC2 adds up the usage, calculates cost, and divides the total charges across each account.

Consolidated Billing Without Reserved Instances Purchased

User	Instance Used	Calculated Usage Fee Cost
User A	9 Instances Comprises one-half of total 18 instances	9 (instances) X 300 (hours for the month) = $2,700.00
User B	6 Instances Comprises one-third of total 18 instances	6 (instances) X 300 (hours for the month) = $1,800.00
User C	3 Instances Comprises one-sixth of total 18 instances	3 (instances) X 300 (hours for the month) = $900.00

With User A's purchase of Reserved Instances, the benefit of the discounted rate for six Reserved Instances will be divided across the three accounts under the master account CB AZ. The division will be based on the usage rate:

- User A: one-half of 6 = 3 instances
- User B: one-third of 6 = 2 instances
- User C: one-sixth of 6 = 1 instance

Master account CB AZ will save a total of $900 in usage fees after the Reserved Instances discounts are applied. The following table shows how the consolidated bill would look with the benefit of Reserved Instances purchased by User A.

Consolidated Billing Showing Benefits of Reserved Instances Purchased

User	Instance Used (with Reserved Instance Usage Fee Discount)	Calculated Usage Fee Cost	Savings
User A	9 Instances Comprises one-half of total 18 instances Gets one-half of discount for 6 Reserved Instances applied	3 (instances) X 300 (hours for the month) X $0.50 (Reserved Instance rate) = $450.00 6 (instances) X 300 (hours for the month) X $1.00= $1,800.00 Total cost: $2,250.00	$450.00
User B	6 Instances Comprises one-third of total 18 instances Gets one-third of discount for 6 Reserved Instances applied	2 (instances) X 300 (hours for the month) X $0.50 (Reserved Instance rate) = $300.00 4 (instances) X 300 (hours for the month) = $1,200.00 Total cost: $1,500.00	$300.00
User C	3 Instances Comprises one-sixth of total 18 instances Gets one-sixth of discount for 6 Reserved Instances applied	1 (instance) X 300 (hours for the month) X $0.50 (Reserved Instance rate) = $150.00 2 (instances) X 300 (hours for the month) = $600.00 Total cost: $750.00	$150.00

Note

There are different pricing options for Reserved Instances, depending on the instance type, Availability Zone, term period, and product platform. For information about pricing, see the Amazon EC2 Reserved Instances product page.

Tools and Prerequisites

To work with Reserved Instances, we assume you have read and completed the instructions described in Getting Started with EC2, which provides information on creating your Amazon EC2 account and credentials.

You can use the AWS Management Console (p. 232), the Command Line Tools (p. 233), or the API (p. 235) to work with Amazon EC2 Reserved Instances and search for offerings.

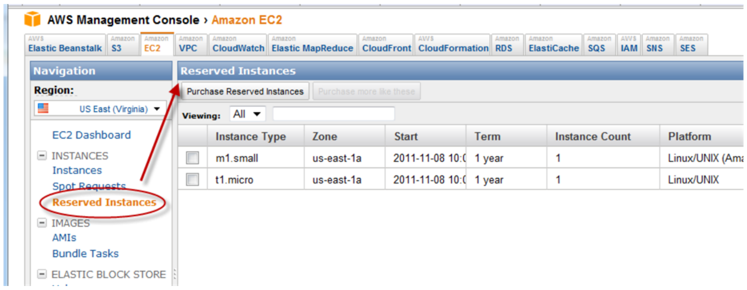
If you use the Amazon EC2 command line interface (CLI) tools, we assume that you have read and completed the instructions described in Getting Started with the Command Line Tools. This topic walks you through setting up your environment for use with the CLI tools.

In addition, whichever you choose to use—AWS Management Console, the CLI, or API—Amazon EC2 provides tools that help you to identify Reserved Instances offerings, purchase your Reserved Instances, and manage them.

AWS Management Console

The AWS Management Console has tools specifically designed for Reserved Instances tasks. The console also has general tools that you can use to manage the instances launched when you use your Reserved Instances. The following list identifies the tools you can use in the Amazon EC2 Dashboard of the AWS Management Console:

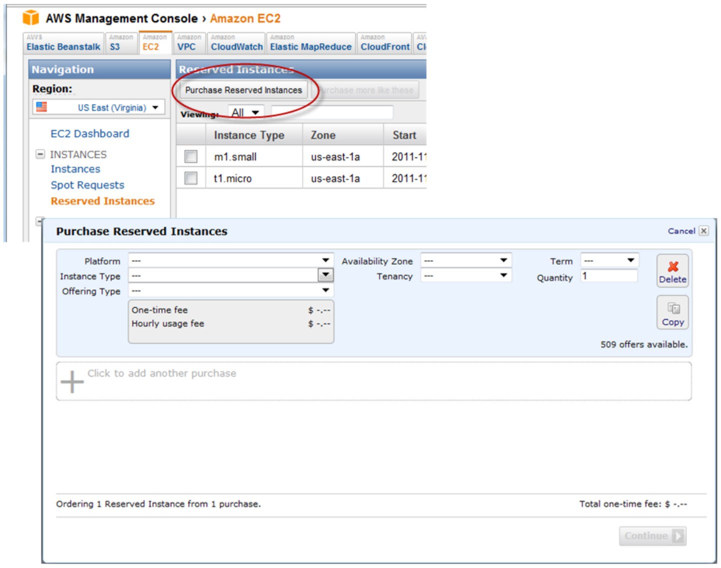
- The **Reserved Instances** page is where you interact with your Reserved Instances.



Viewing: All

	Instance Type	Zone	Start	Term	Instance Count	Platform
□	m1.small	us-east-1a	2011-11-08 10:(	1 year	1	Linux/UNIX (Ama
□	t1.micro	us-east-1a	2011-11-08 10:(	1 year	1	Linux/UNIX

- Use the **Purchase Reserved Instances** page to specify the details of Reserved Instances you want to purchase.



Viewing: All

	Instance Type	Zone	Start
□	m1.small	us-east-1a	2011-1
□	t1.micro	us-east-1a	2011-1

Purchase Reserved Instances

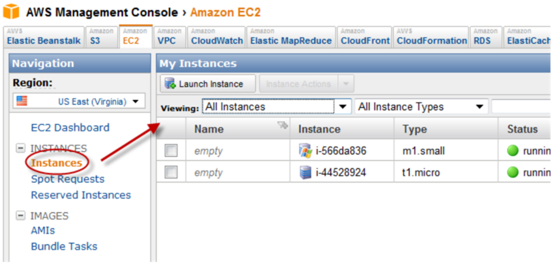
Platform ---
Instance Type ---
Offering Type ---
Availability Zone ---
Tenancy ---
Term ---
Quantity 1

One-time fee $ -.--
Hourly usage fee $ -.--

509 offers available.

Ordering 1 Reserved Instance from 1 purchase.
Total one-time fee: $ -.--

- Use the **Instances** page to manage the instances you launched using your Reserved Instances.



	Name	Instance	Type	Status
□	empty	i-566da836	m1.small	● runnin
□	empty	i-44528924	t1.micro	● runnin

Command Line Tools

To purchase Reserved Instances, you use Amazon EC2 command line tools specifically designed for these tasks. To manage the instances when your Reserved Instances are launched, use the same CLI commands that you use for Amazon EC2 instances.

The following table lists the CLI commands you use for Reserved Instances tasks.

Task	CLI
List Reserved Instances that you have purchased.	`ec2-describe-reserved-instances`
View the Reserved Instances offerings that are available for purchase.	`ec2-describe-reserved-instances-offerings`
Purchase a Reserved Instance.	`ec2-purchase-reserved-instances-offering`

For information about CLI commands, go to the Amazon Elastic Compute Cloud Command Line Reference.

API

To purchase Reserved Instances, you use API calls specifically designed for these tasks. To manage the instances when your Reserved Instances are launched, use the same API calls that you use for Amazon EC2 instances.

The following table lists the API calls you use for Reserved Instances tasks.

Task	API
List Reserved Instances that you have purchased.	`DescribeReservedInstances`
View the Reserved Instances offerings that are available for purchase.	`DescribeReservedInstancesOfferings`
Purchase a Reserved Instance.	`PurchaseReservedInstancesOffering`

For information about API actions, go to the Amazon Elastic Compute Cloud API Reference.

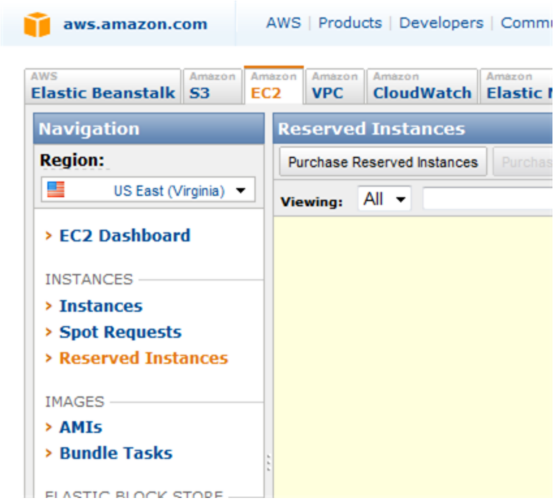
Working with Reserved Instances

In this section, we walk you through the steps of finding, purchasing, and verifying your Reserved Instances.

AWS Management Console

To find and purchase a Reserved Instance

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Reserved Instances** in the **Navigation** pane.
   The console displays a list of your account's instances.
3. Click **Purchase Reserved Instances**.

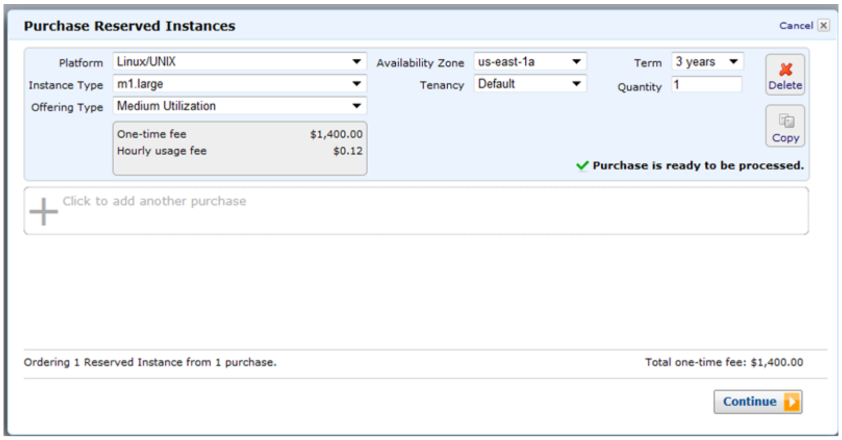


Region: US East (Virginia)

Viewing: All

4. Locate a Reserved Instance to purchase by specifying the following:

   - Platform (if you want to use the Reserved Instance with Amazon Virtual Private Cloud, make sure to select a platform that includes *Amazon VPC* in its name)
   - Instance Type
   - Offering Type (For more information, go to Reserved Instances Offerings (p. 224).)
   - Availability Zone
   - Term
   - Tenancy (For more information, go to Dedicated Instances Basics in the *Amazon Virtual Private Cloud User Guide*.)



Purchase Reserved Instances

Platform: Linux/UNIX
Instance Type: m1.large
Offering Type: Medium Utilization
Availability Zone: us-east-1a
Tenancy: Default
Term: 3 years
Quantity: 1

One-time fee $1,400.00
Hourly usage fee $0.12

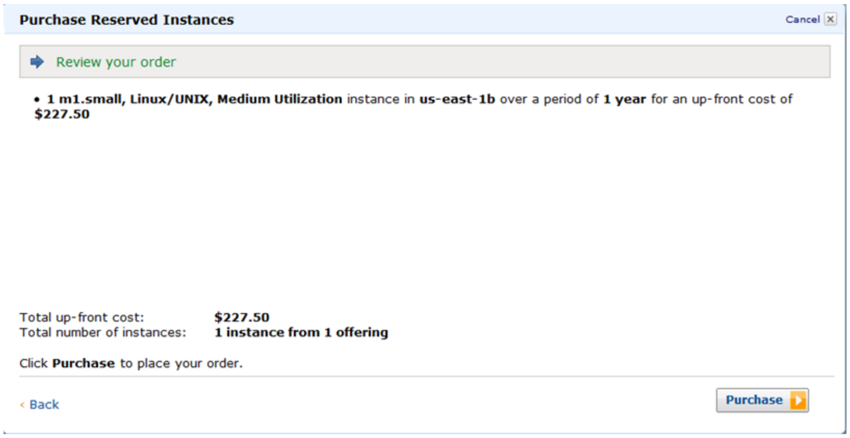
✓ Purchase is ready to be processed.

Ordering 1 Reserved Instance from 1 purchase.

Total one-time fee: $1,400.00

The AWS Management Console displays the **One-time fee** and the **Hourly usage fee**.

5. Select the number of instances to purchase and click **Continue**.
   The AWS Management Console prompts you to confirm your order.

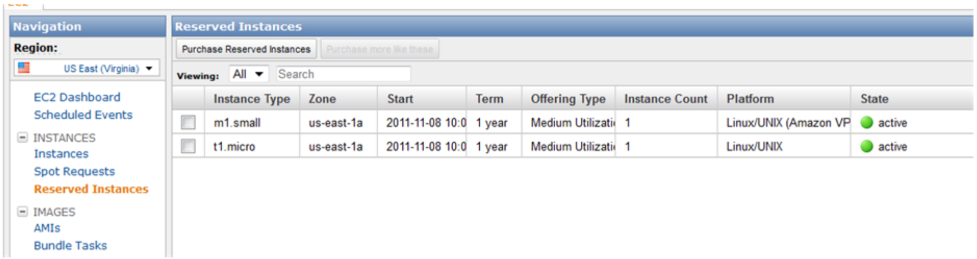


- **1 m1.small, Linux/UNIX, Medium Utilization** instance in **us-east-1b** over a period of **1 year** for an up-front cost of **$227.50**

Total up-front cost: **$227.50**
Total number of instances: **1 instance from 1 offering**

Click **Purchase** to place your order.

6. To confirm your order, click **Purchase**.
   Your purchase is complete.
7. To verify your order, select **Reserved Instances**.



	Instance Type	Zone	Start	Term	Offering Type	Instance Count	Platform	State
□	m1.small	us-east-1a	2011-11-08 10:0	1 year	Medium Utilizati	1	Linux/UNIX (Amazon VP	● active
□	t1.micro	us-east-1a	2011-11-08 10:0	1 year	Medium Utilizati	1	Linux/UNIX	● active

The AWS Management Console displays a list of Reserved Instances that belong to your account.

Command Line Tools

To find and purchase a Reserved Instance

1. Check to see which Reserved Instances are available.

```
PROMPT> ec2-describe-reserved-instances-offerings
```

Amazon EC2 returns output similar to the following:

```
PROMPT> ec2-describe-reserved-instances-offerings
ReservedInstancesOfferingId AvailabilityZone InstanceType Duration FixedPrice
 UsagePrice ProductDescription Currency InstanceTenancy OfferingType
OFFERING 248e7b75-c83a-48c1-bcf7-b7f03e9c43fe us-east-1b c1.medium 3y  700.0
 0.06 Linux/UNIX (Amazon VPC) USD default Medium Utilization
OFFERING 3a98bf7d-05c0-40d0-a173-81a3986ba568 us-east-1b c1.medium 3y  700.0
 0.125 Windows USD default Medium Utilization
OFFERING 4b2293b4-ff40-4a1a-9fef-1f12ad37a711 us-east-1b c1.medium 3y  700.0
```

```
 0.06 Linux/UNIX USD default Medium Utilization
OFFERING 60dcfab3-71e1-42cd-bd3e-f5e1e61ec5a8 us-east-1b c1.medium 3y  700.0
 0.125 Windows (Amazon VPC) USD default Medium Utilization
OFFERING 649fd0c8-cafc-4060-ace6-a074c9621f75 us-east-1b c1.medium 3y  865.0
 0.155 Windows (Amazon VPC) USD dedicated Medium Utilization
...
```

The preceding output shows a part of the overall offerings that are available. Here we show both 3-year (3y) and 1-year (1y) Reserved Instances available for m1.small instances in either the us-east-1b or us-east-1d Availability Zones.

Important

If you want to use the Reserved Instance with Amazon Virtual Private Cloud, make sure to select an offering that includes *Amazon VPC* in the description (e.g., Linux/UNIX (Amazon VPC)).

Tip

You can filter this list to return only certain types of Reserved Instances offerings of interest to you. For more information about how to filter the results, go to ec2-describe-reserved-instances-offerings in the *Amazon Elastic Compute Cloud Command Line Reference*.

2. From the list of available Reserved Instances, purchase the Reserved Instances that meet your requirements. To purchase a Reserved Instance, use `ec2-purchase-reserved-instances-offering`.

```
PROMPT> ec2-purchase-reserved-instances-offering --offering offering_id -
-instance-count count
```

Amazon EC2 returns output similar to the following:

```
PURCHASE   af9f760e-c1c1-449b-8128-1342d3a6927a  438012d3-80c7-42c6-9396-
a209c58607f9
```

The response includes the offering ID and a reservation ID.

3. Write down or save the reservation ID for future reference.
4. Verify the purchase:

```
PROMPT> ec2-describe-reserved-instances
```

Amazon EC2 returns output similar to the following:

```
RESERVEDINSTANCE  af9f760e-c1c1-449b-8128-1342d3a6927a    us-east-1b
m1.small        1y      227.5   0.03    Linux/UNIX  Active
```

API

To find and purchase a Reserved Instance

1. Check to see which Reserved Instances are available and where they are located using the following Query request.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeReservedInstancesOfferings
   &...auth parameters...
   ```

   Following is an example response (note that the duration is in seconds; 31536000 seconds equals 1 year).

   ```
   <DescribeReservedInstancesOfferings xmlns="http://ec2.amazonaws.com/doc/2011-
   12-01/">
     <reservedInstancesOfferingsSet>
       <item>
         <reservedInstancesOfferingId>438012d3-80c7-42c6-9396-
   a209c58607f9</reservedInstancesOfferingId>
         <instanceType>m1.small</instanceType>
         <availabilityZone>us-east-1b</availabilityZone>
         <duration>31536000</duration>
         <fixedPrice>227.5</fixedPrice>
         <usagePrice>0.03</usagePrice>
         <productDescription>Linux/UNIX</productDescription>
       </item>
       ...
     </reservedInstancesOfferingsSet>
     </DescribeReservedInstancesOfferings>
   ```

   Important

   If you want to use the Reserved Instance with Amazon Virtual Private Cloud, make sure to select an offering that includes *Amazon VPC* in the product description (e.g., Linux/UNIX (Amazon VPC)).

   Tip

   You can filter this list to return only certain types of Reserved Instances offerings of interest to you. For more information about how to filter the results, go to DescribeReservedInstancesOfferings in the *Amazon Elastic Compute Cloud API Reference*.

2. From the list of available Reserved Instances, purchase an offering using a Query request similar to the following.

   ```
   https://ec2.amazonaws.com/
   ?Action=PurchaseReservedInstancesOffering
   &ReservedInstancesOfferingId=438012d3-80c7-42c6-9396-a209c58607f9
   &instanceCount=2
   &...auth parameters...
   ```

   Following is an example response.

```
<PurchaseReservedInstancesOffering xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <reservedInstancesId>af9f760e-c1c1-449b-8128-1342d3a6927a</reservedIn
stancesId>
</PurchaseReservedInstancesOffering>
```

3. To verify the purchase, construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeReservedInstances
&...auth parameters...
```

Following is an example response (note that the duration is in seconds; 31536000 seconds equals 1 year):

```
<DescribeReservedInstances xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <reservedInstancesSet>
    <item>
     <reservedInstancesId>af9f760e-c1c1-449b-8128-1342d3a6927a</reservedIn
stancesId>
      <instanceType>m1.small</instanceType>
      <availabilityZone>us-east-1b</availabilityZone>
      <duration>31536000</duration>
      <usagePrice>227.5</usagePrice>
      <fixedPrice>0.03</fixedPrice>
      <instanceCount>2</instanceCount>
      <productDescription>Linux/UNIX</productDescription>
      <state>Active</state>
    </item>
  </reservedInstancesSet>
</DescribeReservedInstances>
```

Related Topics

- Reserved Instances FAQs (p. 416)

Spot Instances

If you have flexibility on when your application will run, you can bid on unused Amazon EC2 compute capacity, called Spot Instances, and lower your costs significantly. Set by Amazon EC2, the Spot Price for these instances fluctuates periodically depending on the supply of and demand for Spot Instance capacity.

To use Spot Instances, you place a Spot Instance request (your bid) specifying the maximum price you are willing to pay per hour per instance. If the maximum price of your bid is greater than the current Spot Price, your request is fulfilled and your instances run until you terminate them or the Spot Price increases above your maximum price (whichever comes first).

You will often pay less per hour than your maximum bid price. The Spot Price is adjusted periodically as requests come in and the available supply of instances changes. Everyone pays that same Spot Price for that period regardless of whether their maximum bid price was higher, and you will never pay more than your hourly maximum bid price.

Quick Look: Getting Started with Spot Instances Video

The following video shows how to launch your first Spot Instance using the AWS Management Console. This video includes instructions on placing a bid, determining when the instance is fulfilled, and canceling the instance. Getting Started with Spot Instances

Checklist for Getting Started with Spot Instances

If you want to get started working with Spot Instances, here are the resources you need to get going.

- Get Started with Spot Instances (p. 238)
  - View Spot Price History (p. 241)
  - Create a Spot Instance Request (p. 244)
  - Find Running Spot Instances (p. 249)
  - Cancel Spot Instance Requests (p. 253)
- Plan for Interruptions (p. 256)
- Bidding Strategies (p. 257)
- Advanced Tasks (p. 257)
  - Persist Your Root EBS Partition (p. 258)
  - Tag Spot Instance Requests (p. 258)
  - Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)
  - Subscribe to Your Spot Instance Data Feed (p. 261)

What's New in Spot Instances

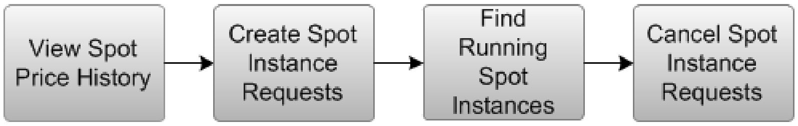
Here's a quick look at what's new in Spot Instances:

- Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)
  For a quick look at how to launch Spot Instances in Amazon VPC, see the video Quick Look: Launching Spot Instances in Amazon VPC Video (p. 259).

- AWS Java SDK (p. 240)

Get Started with Spot Instances

This section gives you a comprehensive overview of what you can do with Spot Instances. The following diagram gives you a high-level look at the tasks you can perform with Spot Instances.


View Spot Price History
Create Spot Instance Requests
Find Running Spot Instances
Cancel Spot Instance Requests

If you are new to Spot Instances, take a look at Prerequisites for Using Spot Instances (p. 238) to make sure you can take full advantage of the benefits of this Amazon EC2 product.

If you have been using Amazon EC2 and you're ready to look into making a Spot Instance request, proceed to one of the following steps:

- View Spot Price History (p. 241)
- Create a Spot Instance Request (p. 244)
- Find Running Spot Instances (p. 249)
- Cancel Spot Instance Requests (p. 253)
- Advanced Tasks (p. 257)

Prerequisites for Using Spot Instances

To work with Amazon EC2 Spot Instances, we assume you have read and completed the instructions described in the Getting Started with EC2, which provides information on creating your Amazon EC2 account and credentials.

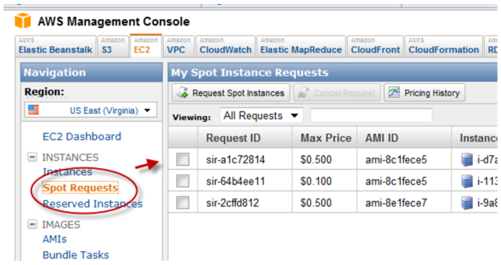
If you use the Amazon EC2 command line interface (CLI) tools, we assume that you have read and completed the instructions described in the Getting Started with the Command Line Tools of the *Amazon EC2 User Guide*. It walks you through setting up your environment for use with the CLI tools.

In addition, whichever you choose to use—AWS Management Console, the CLI, or API—Amazon EC2 provides tools that specifically help you work with Spot Instances to assess Spot Price history, submit Spot Instance requests, and manage your Spot Instance requests (bids) and instances. You can also use programming languages to perform most of your work.

AWS Management Console

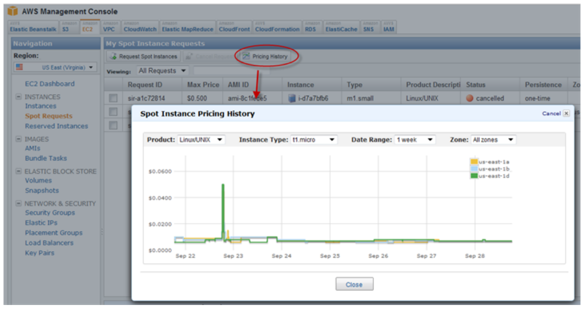
The AWS Management Console has tools specifically designed for Spot Instance request tasks. The console also has general tools that you can use to manage the instances launched when your Spot Instance requests are fulfilled. The following list identifies the tools you can use in the Amazon EC2 dashboard of the AWS Management Console:

- The **Spot Requests** page is the main way you interact with your Spot Instance requests.



	Request ID	Max Price	AMI ID	Instanc
□	sir-a1c72814	$0.500	ami-8c1fece5	i-d7a
□	sir-64b4ee11	$0.100	ami-8c1fece5	i-113
□	sir-2cffd812	$0.500	ami-8e1fece7	i-9a8

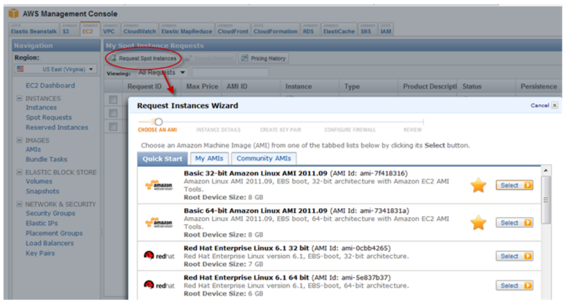
- **Spot Instance Pricing History** gives you an insight in the pricing patterns for specific Spot Instance types in Availability Zones over a defined period.



	Request ID	Max Price	AMI ID	Instance	Type	Product Descripti	Status	Persistence
□	sir-a1c72814	$0.500	ami-8c1fece5	i-d7a7bfb6	m1.small	Linux/UNIX	cancelled	one-time

Product: Linux/UNIX Instance Type: t1.micro Date Range: 1 week Zone: All zones

- Use the **Request Spot Instances** page to specify the details of instances to be launched when your Spot Instance bid succeeds.



Choose an Amazon Machine Image (AMI) from one of the tabbed lists below by clicking its **Select** button.

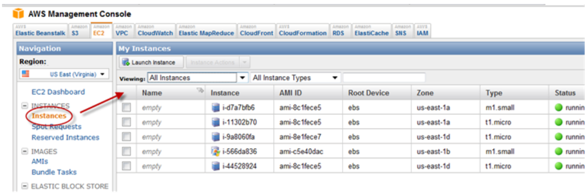
Basic 32-bit Amazon Linux AMI 2011.09 (AMI Id: ami-7f418316)
Amazon Linux AMI 2011.09, EBS boot, 32-bit architecture with Amazon EC2 AMI Tools.
Root Device Size: 8 GB

Basic 64-bit Amazon Linux AMI 2011.09 (AMI Id: ami-7341831a)
Amazon Linux AMI 2011.09, EBS boot, 64-bit architecture with Amazon EC2 AMI Tools.
Root Device Size: 8 GB

Red Hat Enterprise Linux 6.1 32 bit (AMI Id: ami-0cbb4265)
Red Hat Enterprise Linux version 6.1, EBS-boot, 32-bit architecture.
Root Device Size: 7 GB

Red Hat Enterprise Linux 6.1 64 bit (AMI Id: ami-5e837b37)
Red Hat Enterprise Linux version 6.1, EBS-boot, 64-bit architecture.
Root Device Size: 6 GB

- Use the **Instances** page to manage the instances launched when your Spot Instance bid succeeds.



	Name	Instance	AMI ID	Root Device	Zone	Type	Status
□	empty	i-d7a7bfb6	ami-8c1fece5	ebs	us-east-1a	m1.small	runnin
□	empty	i-11302b70	ami-8c1fece5	ebs	us-east-1a	t1.micro	runnin
□	empty	i-9a8060fa	ami-8e1fece7	ebs	us-east-1d	t1.micro	runnin
□	empty	i-566da836	ami-c5e40dac	ebs	us-east-1b	m1.small	runnin
□	empty	i-44528924	ami-8c1fece5	ebs	us-east-1d	t1.micro	runnin

Command Line Tools

To manage your Spot Requests, you use Amazon EC2 command line tools specifically designed for these tasks. To manage the instances launched when your Spot Request is fulfilled, use the same CLI commands that you use for Amazon EC2 instances.

The following table lists the CLI commands you use for Spot Request tasks.

Task	CLI
View Spot Price History (p. 241).	`ec2-describe-spot-price-history`
Find Running Spot Instances (p. 249).	`ec2-describe-spot-instance-requests`
Create a Spot Instance Request (p. 244).	`ec2-request-spot-instances`
Subscribe to Your Spot Instance Data Feed (p. 262).	`ec2-create-spot-datafeed-subscription`
Data Feed Filename and Format (p. 261).	`ec2-describe-spot-datafeed-subscription`
Delete a Spot Instance Data Feed (p. 263).	`ec2-delete-spot-datafeed-subscription`
Cancel Spot Instance Requests (p. 253).	`ec2-cancel-spot-instance-requests`

For information about CLI commands, go to the Amazon Elastic Compute Cloud Command Line Reference.

API

To manage your Spot Requests, you use Amazon EC2 API tools specifically designed for these tasks. To manage the instances launched when your Spot Request is fulfilled, use the same API actions that you use for Amazon EC2 instances.

The following table lists the API actions you use for Spot Request tasks.

Task	API
View Spot Price History (p. 241).	`DescribeSpotPriceHistory`
Find Running Spot Instances (p. 249).	`DescribeSpotInstanceRequests`
Create a Spot Instance Request (p. 244).	`RequestSpotInstances`
Subscribe to Your Spot Instance Data Feed (p. 262).	`CreateSpotDatafeedSubscription`
Data Feed Filename and Format (p. 261).	`DescribeSpotDatafeedSubscription`
Delete a Spot Instance Data Feed (p. 263).	`DeleteSpotDatafeedSubscription`
Cancel Spot Instance Requests (p. 253).	`CancelSpotInstanceRequests`

For information about API actions, go to the Amazon Elastic Compute Cloud API Reference.

AWS Java SDK

Java developers can go to the *AWS SDK for Java* to consult the growing library of tutorials on Spot Instances:

- Tutorial for Java Developers: Amazon EC2 Spot Instances
- Tutorial for Java Developers: Advanced Amazon EC2 Spot Instance Request Management

View Spot Price History

Before specifying a rate you want to bid for your Spot Instance, we recommend that you view the Spot Price history. You can view the Spot Price history over a period from one to 90 days based on the instance type, the operating system you want the instance to run on, the time period, and the Availability Zone in which it will be launched.

For example, let's say you want to bid for a Linux/UNIX micro instance to be launched in the us-east-1a Availability Zone. Specify these values in the API or select them in the **Spot Price History** page of the AWS Management Console. You will be able to view the price for the instance type and operating system in the Availability Zone you want during the period you specified. On the other hand, if you don't need to launch the instances in a specific Availability Zone and consequently you don't specify this option, Amazon EC2 returns the prices for all Availability Zones.

Keep in mind that if you use the `DescribeSpotPriceHistory` action or `ec2-describe-spot-price-history` command before the 2011-05-15 API version, you will get the lowest price across the Region for the given time period and the prices will be returned in chronological order.

Note

Make sure you have set up the prerequisites to working with Amazon EC2. If you haven't, go to Prerequisites for Using Spot Instances (p. 238).

AWS Management Console

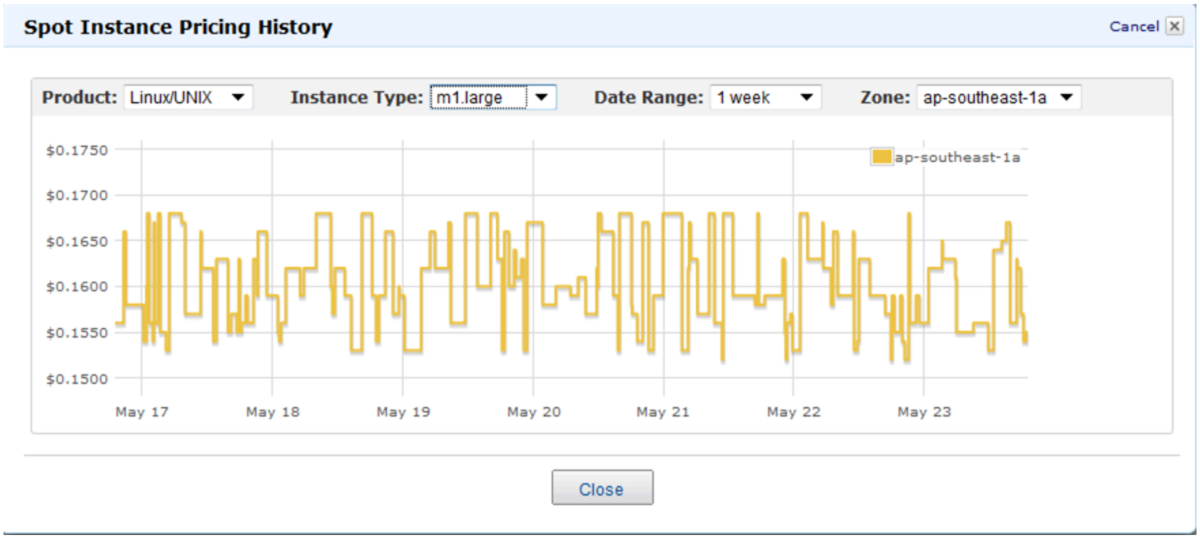
To view Spot Price history

1. From the AWS Management Console, click **Spot Requests** in the navigation pane.
   The **My Spot Instance Requests** pane opens.
2. At the top of the **My Spot Instance Requests** pane, click the **Pricing History** button.
   The console displays the Spot Instance pricing history.

   Note

   Using the historical pricing as a guide, select a price that you think would be likely to keep your instances running for the period of time you need.

3. If you want to view the Spot Price history for specific Availability Zones, click the **Zone** drop-down list and select an Availability Zone.
   The **Spot Instance Pricing History** page displays the Spot Instance pricing history for the zone you selected.



Spot Instance Pricing History

Product: Linux/UNIX Instance Type: m1.large Date Range: 1 week Zone: ap-southeast-1a

Command Line Tools

To view Spot Price history

1. Enter the following command:

```
PROMPT> ec2-describe-spot-price-history -H --instance-type m1.xlarge
```

   Amazon EC2 returns output similar to the following:

```
SPOTINSTANCEPRICE   0.384000    2011-05-25T11:37:48-0800    m1.xlarge   Windows
 us-east-1b
SPOTINSTANCEPRICE   0.384000    2011-05-25T11:37:48-0800    m1.xlarge   Windows
 us-east-1d
…
SPOTINSTANCEPRICE   0.242000    2011-04-18T14:39:14-0800    m1.xlarge   SUSE
 Linux  us-east-1d
SPOTINSTANCEPRICE   0.242000    2011-04-18T14:39:14-0800    m1.xlarge   SUSE
 Linux  us-east-1a
```

   In this example, the price for the `m1.xlarge` instance type ranges between $0.242 and $0.384.

2. Based on the historical pricing, select a price that is likely to keep your instances running for the period of time that you need.

Tip

You can filter the spot history data so it includes only instance types or dates of interest to you. For more information about how to filter the results, go to ec2-describe-spot-price-history in the *Amazon Elastic Compute Cloud Command Line Reference*.

API

To view Spot Price history

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeSpotPriceHistory
&InstanceType=instance_type
&...auth parameters...
```

Following is an example response.

```
<DescribeSpotPriceHistoryResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
    <spotPriceHistorySet>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.28</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1a</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.28</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1a</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.31</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1b</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.30</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1b</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.25</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1c</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.28</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1c</availabilityZone>
      </item>
      <item>
        <instanceType>m1.small</instanceType>
        <productDescription>Linux/UNIX</productDescription>
        <spotPrice>.35</spotPrice>
        <timestamp>2009-12-01T11:51:50.000Z</timestamp>
        <availabilityZone>us-east-1c</availabilityZone>
      </item>
    </spotPriceHistorySet>
    <nextToken/>
</DescribeSpotPriceHistoryResponse>
```

Note

Based on the historical pricing, select a price that is likely to keep your instances running for the period of time that you need.

Tip

You can filter the spot history data so it includes only instance types or dates of interest to you. For more information about how to filter the results, go to DescribeSpotPriceHistory in the *Amazon Elastic Compute Cloud API Reference*.

What do you want to do next?

- Create a Spot Instance Request (p. 244)
- Find Running Spot Instances (p. 249)
- Bidding Strategies (p. 257)
- Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)

Create a Spot Instance Request

After deciding on a maximum price, you are ready to request Spot Instances. The maximum price is not necessarily the price that you will pay. For example, if you specify $.05 as your maximum price and the Spot Price is $.03 for the period, you will pay $.03. If the Spot Price drops, you will pay less. If the Spot Price increases, you will pay the new price (up to your maximum).

If your maximum price is greater than the Spot Price and all additional constraints are met, Amazon EC2 automatically launches instances on your behalf.

If your Spot Instances request is for multiple instances, Amazon EC2 creates a separate Spot Instance request ID (e.g., sir-1a2b3c4d) for each instance. You can then track the status of each of the requests separately.

This section discusses the details of creating requests for Spot Instances to be launched in Amazon EC2. You can also launch Spot Instances in the Amazon Virtual Private Cloud (Amazon VPC) and the steps are similar, but they will not be discussed here. For more information about launching Spot Instances in Amazon VPC, see Launch Spot Instances in Amazon Virtual Private Cloud (p. 259).

Before requesting a Spot Instance, ensure you have an Amazon Machine Image (AMI) that does the following for your instance:

- Automatically performs the tasks you want at start-up because the instance will start asynchronously without notification.
- Stores important data regularly and in a place that won't be affected by instance termination, such as Amazon S3, Amazon SimpleDB, or Amazon EBS.
- Handles termination gracefully.

For information about creating AMIs, see Creating Your Own AMIs (p. 17).

Important

Although Spot Instances can use Amazon EBS-backed AMIs, be aware that the EBS-backed AMIs don't support Stop/Start. In other words, you can't stop and start Spot Instances launched from an AMI with an Amazon EBS root device.

Types of Spot Instance Requests

You can make two types of Spot Instance requests—a one-time request or a persistent request. A one-time request remains active until either all the requested instances launch, the request expires, or you cancel the request. For example, if you create a one-time request for three instances, the request is considered complete after all three instances launch.

Persistent Spot Instance requests remain active until they expire or you cancel them, even if the request was previously satisfied. For example, if you create a persistent Spot Instance request for one instance at $.30, Amazon EC2 launches or keeps your instance running if your maximum bid price is above $.30 and terminates your instance if your maximum bid price is below $.30.

With both one-time and persistent requests, instances continue to run until either they no longer exceed the Spot Price, you terminate them, or they terminate on their own. If the maximum price is exactly equal to the Spot Price, an instance might or might not continue running (depending on available capacity).

Launching Spot Instances in Launch Groups and Availability Zones

You can opt to have your Spot Instances launch at the same time or in the same Availability Zone. To tell Amazon EC2 to launch your instances only if all the instances in the request can be fulfilled, specify a Launch Group in your Spot Instance request.

To specify a launch group in the AWS Management Console, you select **Launch Group** in the Request Instances Wizard; with the CLI or API tools, you specify the `launch group` option when you make the request spot instances call.

If you want all your instances to be launched together in a single Availability Zone, specify an Availability Zone Group. Do this by selecting **Availability Zone Group** when using the Request Instances Wizard in the AWS Managment Console, or by specifying the `availability-zone-group` option in the CLI or API.

Although the Launch Group and Availability Zone Group specifications can be advantageous, use them only when needed. Avoiding these requirements increases the chances that your Spot Instance request can be fulfilled.

When you want to identify a bid price for a Spot Instance in a specific Availability Zone, just view the price history for the Availability Zone where you want to make your bid. Do this by selecting the Availability Zone from the **Zone** drop-down menu in the **Spot Instance Pricing History** page in the AWS Management Console. Alternatively, call `DescribeSpotPriceHistory` or use the `ec2-describe-spot-price-history` CLI command. You will see the price history for the specified Availability Zone with the most recent set of prices listed first. If you don't specify an Availability Zone, you will get the lowest prices across all Availability Zones for the time period, starting with the most recent set.

> **Note**
>
> When you use the `DescribeSpotPriceHistory` action or the `ec2-describe-spot-price-history` command before the 2011-05-15 API version, you will get the lowest price across the Region for the given time period and the prices will be returned in chronological order—that is, from the oldest to the most recent.

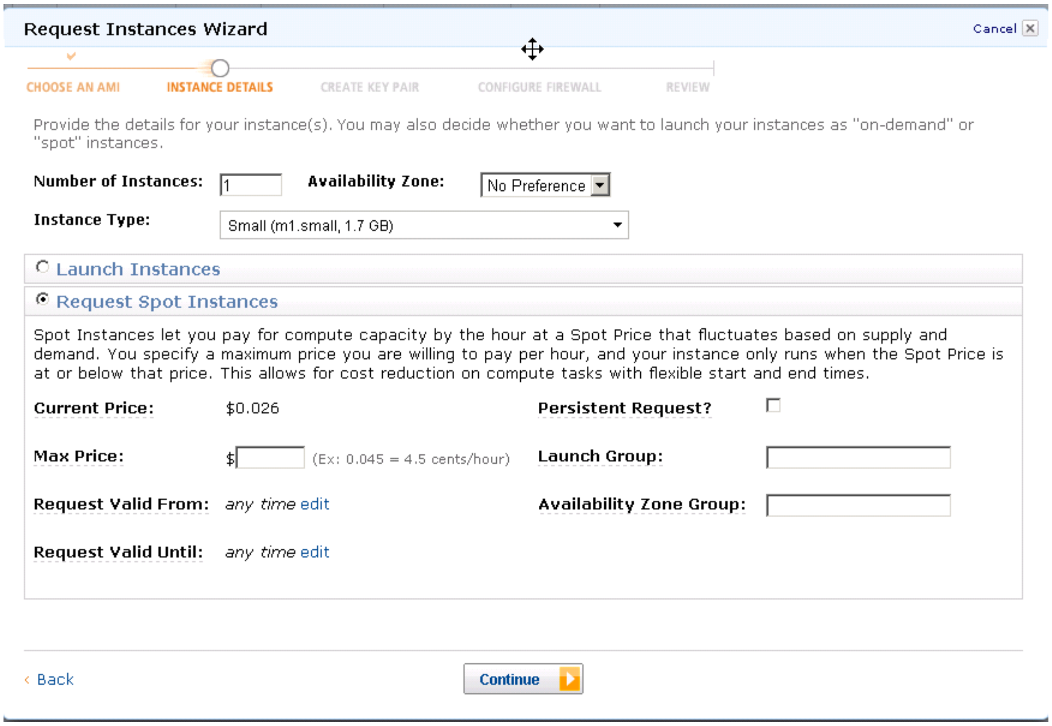
For more information about Availability Zones, see Region and Availability Zone Concepts (p. 321).

> **Note**
>
> Make sure you have set up the prerequisites for working with Amazon EC2. If you haven't, go to Prerequisites for Using Spot Instances (p. 238).

AWS Management Console

To create a Spot Price request

1. From the Amazon EC2 tab of the AWS Management Console, click **Spot Requests** in the navigation pane.
   The **My Spot Instance Request** pane opens.
2. Click the **Request Spot Instances** button in the right pane.
   The Request Instances Wizard starts.
3. Choose an AMI and click **Select**.
   The **INSTANCE DETAILS** page appears with the **Request Spot Instances** button already selected.



Provide the details for your instance(s). You may also decide whether you want to launch your instances as "on-demand" or "spot" instances.

Number of Instances: 1 Availability Zone: No Preference

Instance Type: Small (m1.small, 1.7 GB)

○ Launch Instances

● Request Spot Instances

Spot Instances let you pay for compute capacity by the hour at a Spot Price that fluctuates based on supply and demand. You specify a maximum price you are willing to pay per hour, and your instance only runs when the Spot Price is at or below that price. This allows for cost reduction on compute tasks with flexible start and end times.

Current Price: $0.026 Persistent Request? □

Max Price: $ (Ex: 0.045 = 4.5 cents/hour) Launch Group:

Request Valid From: *any time* edit Availability Zone Group:

Request Valid Until: *any time* edit

4. Specify the number and type of instances you want, and the desired Availability Zone (if any).

Note

Current Price displays the price for the Availability Zone you specified. If you do not specify an Availability Zone, **Current Price** displays the lowest price across all Availability Zones. The information displayed when the cursor hovers over the **Current Price** shows the price for each of the Availability Zones.

5. Configure the Spot Instance settings using the following table as a guide, and then click **Continue**.

Option	Description
Max Price	Specifies the maximum price you are willing to pay per instance hour.
Request Valid From and **Request Valid Until**	Defines the validity period of a Spot Request. The valid period, beginning with **Request Valid From** through **Request Valid Until**, specifies the length of time that your request will remain valid. By default, a Spot Request will be considered for fulfillment from the time it is created until it is either fulfilled or canceled by you. However, you can constrain the validity period using these options. **Note** The end time you specify doesn't apply to the Spot Instances launched by this request. This end time only applies to the Spot Instance request.
Persistent Request?	Determines whether your request is one-time or persistent. By default, it is one-time. A one-time request can only be fulfilled once. A persistent request is considered for fulfillment whenever there is no Spot Instance running for the same request.
Launch Group	Groups a set of requests together in the same launch group. All requests in a launch group have their instances started and terminated together.
Availability Zone Group	Groups a set of requests together in the same Availability Zone. All requests that share an Availability Zone group will launch Spot Instances in the same Availability Zone.

6. Continue with the launch wizard as you normally would when launching instances.

   If your request specified multiple instances, EC2 creates a separate Spot Instance request for each instance. The requests are displayed on the **Spot Requests** page.

Command Line Tools

To create a Spot Price request

- Enter the following command:

```
PROMPT> ec2-request-spot-instances
--price price
--user-data data
--instance-count count
--type one-time | persistent
[--valid-from timestamp]
[--valid-until timestamp]
[--launch-group launchgroup]
(run-instances-arguments)
```

  For example:

```
PROMPT> ec2-request-spot-instances --price 0.32 ami-1234abcd --key MyKeypair
 --group websrv --instance-type m1.small --instance-count 10 --type one-time
```

  Amazon EC2 returns output similar to the following:

```
SPOTINSTANCEREQUEST     sir-09fb0a04    0.32    one-time        Linux/UNIX
   open     2010-04-06T10:03:09+0200   ami-1234abc     m1.small
MyKeypair     websrv      monitoring-disabled
...
```

If your request specified multiple instances, EC2 creates a separate Spot Instance request for each instance; each instance has a different ID (e.g., sir-09fb0a04).

API

To create a Spot Price request

- Construct a Query request similar to the following.

```
https://ec2.amazonaws.com/
?Action=RequestSpotInstances
&SpotPrice.1=0.50
&InstanceCount.1=10
&Type.1=one-time
&AvailabilityZoneGroup.1=MyAzGroup
&LaunchSpecification.ImageId.1=ami-43a4412a
&LaunchSpecification.KeyName.1=MyKeypair
&LaunchSpecification.GroupSet.1=websrv
&LaunchSpecification.InstanceType.1=m1.small
&...auth parameters...
```

  Following is an example response.

```
<RequestSpotInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-
01/">
   <spotInstanceRequestSet>
      <item>
         <spotInstanceRequestId>sir-876aff12</spotInstanceRequestId>
         <spotPrice>0.32</spotPrice>
         <type>one-time</type>
         <state>open</state>
         <fault/>
         <validFrom/>
         <validUntil/>
         <launchGroup/>
         <availabilityZoneGroup>MyAzGroup</availabilityZoneGroup>
         <launchSpecification>
            <imageId>ami-43a4412a</imageId>
            <keyName>MyKeypair</keyName>
            <groupSet>
               <item>
                  <groupId>websrv</groupId>
               </item>
            </groupSet>
            <instanceType>m1.small</instanceType>
            ...
         </launchSpecification>
         <instanceId/>
         <createTime>2009-10-19T00:00:00+0000</createTime>
         <productDescription>Linux/UNIX</productDescription>
         <tagSet/>
      </item>
      ...
   </spotInstanceRequestSet>
</RequestSpotInstancesResponse>
```

If your request specified multiple instances, EC2 creates a separate Spot Instance request for each instance; each instance has a different ID (e.g., sir-876aff12).

What do you want to do next?

- Find Running Spot Instances (p. 249)
- Cancel Spot Instance Requests (p. 253)
- Plan for Interruptions (p. 256)
- Bidding Strategies (p. 257)
- Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)
- Tag Spot Instance Requests (p. 258)
- Subscribe to Your Spot Instance Data Feed (p. 261)

Find Running Spot Instances

Spot Instances continue running until the Spot Price is greater than your maximum bid price. In addition, when your Spot Instance request is canceled, the Spot Instances that were launched previously through the canceled request don't automatically get terminated. The only time that the Spot Instance service terminates a running instance is when the Spot Price exceeds your price. This section describes how to find running Spot Instances.

Important

Although Spot Instances can use Amazon EBS-backed AMIs, be aware that the EBS-backed AMIs don't support Stop/Start. In other words, you can't stop and start Spot Instances launched from an AMI with an Amazon EBS root device.

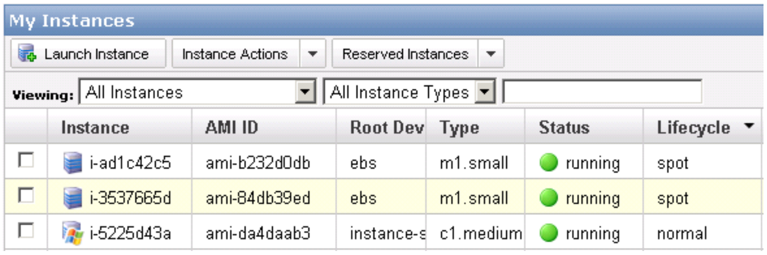
Note

Make sure you have set up the prerequisites for working with Amazon EC2. If you haven't, go to Prerequisites for Using Spot Instances (p. 238).

AWS Management Console

To find running Spot Instances

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. From the AWS Management Console for EC2, click **Instances** in the **Navigation** pane.
   The console displays a list of running instances.



Viewing: All Instances | All Instance Types

	Instance	AMI ID	Root Dev	Type	Status	Lifecycle
□	i-ad1c42c5	ami-b232d0db	ebs	m1.small	● running	spot
□	i-3537665d	ami-84db39ed	ebs	m1.small	● running	spot
□	i-5225d43a	ami-da4daab3	instance-s	c1.medium	● running	normal

3. Look for instances in the table where the **Lifecycle** column contains *spot*. You might need to turn on the display of the column (click **Show/Hide** in the top right corner of the page).

Command Line Tools

To find running Spot Instances

- Use `ec2-describe-spot-instance-requests`. If your Spot Instance request has been fulfilled (an instance has been launched), the instance ID appears in the response.

```
PROMPT> ec2-describe-spot-instance-requests
SPOTINSTANCEREQUEST     sir-e1471206    0.09    one-time        Linux/UNIX
    active  2010-09-13T16:50:44-0800
i-992cf7dd      ami-813968c4    m1.small        MyKey   default
            monitoring-disabled
```

- You can alternately use `ec2-describe-instances` with the following filter: `--filter instance-lifecycle=spot`. If you're using the command line tools on a Windows system, you might need to use quotation marks (`--filter "instance-lifecycle=spot"`). For more information about filters, see Listing and Filtering Your Resources (p. 270).

```
PROMPT>  ec2-describe-instances --filter instance-lifecycle=spot
```

EC2 returns output similar to the following:

```
RESERVATION     r-b58651f1      999988887777    default
INSTANCE        i-992cf7dd      ami-813968c4    ec2-184-72-8-111.us-west-
1.compute.amazonaws.com    ip-10-166-105-139.us-west-1.compute.internal
  running    MyKey   0              m1.small        2010-09-13T23:54:40+0000
        us-west-1a      aki-a13667e4    ari-a33667e6
  monitoring-disabled   184.72.8.111    10.166.105.139                  ebs
    spot    sir-e1471206                paravirtual
```

API

To find running Spot Instances

1. Construct a `DescribeInstances` Query request and use a filter to look for instances where `instance-lifecycle=spot`. For more information about filters, see Listing and Filtering Your Resources (p. 270).

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstances
   &Filter.1.Name=instance-lifecycle
   &Filter.1.Value.1=spot
   &...auth parameters...
   ```

   Following is an example response. It includes an `instanceLifecycle` element with `spot` as the value.

   ```
   <DescribeInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

     ...
      <instancesSet>
      <item>
        <instanceId>i-992cf7dd</instanceId>
        <imageId>ami-813968c4</imageId>
        <instanceState>
          <code>16</code>
          <name>running</name>
        </instanceState>
       <privateDnsName>ip-10-166-105-139.us-west-1.compute.internal</privateDns
   Name>
        <dnsName>ec2-184-72-8-111.us-west-1.compute.amazonaws.com</dnsName>
        <reason/>
        <keyName>MyKey</keyName>
        <amiLaunchIndex>0</amiLaunchIndex>
        <productCodes/>
        <instanceType>m1.small</instanceType>
        <launchTime>2010-09-13T23:54:40.000Z</launchTime>
        <placement>
          <availabilityZone>us-west-1a</availabilityZone>
          <groupName/>
        </placement>
        <kernelId>aki-a13667e4</kernelId>
        <ramdiskId>ari-a33667e6</ramdiskId>
        <monitoring>
   ```

```
       <state>disabled</state>
     </monitoring>
     <privateIpAddress>10.166.105.139</privateIpAddress>
     <ipAddress>184.72.8.111</ipAddress>
     <architecture>i386</architecture>
     <rootDeviceType>ebs</rootDeviceType>
     <rootDeviceName>/dev/sda1</rootDeviceName>
     <blockDeviceMapping>
       <item>
         <deviceName>/dev/sda1</deviceName>
         <ebs>
           <volumeId>vol-61088f0a</volumeId>
           <status>attached</status>
           <attachTime>2010-09-13T23:54:42.000Z</attachTime>
           <deleteOnTermination>true</deleteOnTermination>
         </ebs>
       </item>
     </blockDeviceMapping>
     <instanceLifecycle>spot</instanceLifecycle>
     <spotInstanceRequestId>sir-e1471206</spotInstanceRequestId>
     <virtualizationType>paravirtual</virtualizationType>
   </item>
 </instancesSet>
</DescribeInstancesResponse>
```

2. You can alternately use `DescribeSpotInstanceRequests`. If your Spot Instance request has been fulfilled (an instance has been launched), the instance ID appears in the response. Following is an excerpt from a response.

```
...
<spotInstanceRequestSet>
   <item>
      <spotInstanceRequestId>sir-e1471206</spotInstanceRequestId>
      <spotPrice>0.09</spotPrice>
      <type>one-time</type>
      <state>active</state>
      <launchSpecification>
         <imageId>ami-813968c4</imageId>
         <keyName>MyKey</keyName>
         <groupSet>
            <item>
               <groupId>default</groupId>
            </item>
         </groupSet>
         <instanceType>m1.small</instanceType>
         <blockDeviceMapping/>
         <monitoring>
            <enabled>false</enabled>
         </monitoring>
      </launchSpecification>
      <instanceId>i-992cf7dd</instanceId>
      <createTime>2010-09-13T23:50:44.000Z</createTime>
      <productDescription>Linux/UNIX</productDescription>
      <launchedAvailabilityZone>us-east-1c</launchedAvailabilityZone>
   </item>
```

```
<spotInstanceRequestSet/>
...
```

What do you want to do next?

- Cancel Spot Instance Requests (p. 253)
- Plan for Interruptions (p. 256)
- Bidding Strategies (p. 257)
- Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)
- Persist Your Root EBS Partition (p. 258)
- Tag Spot Instance Requests (p. 258)
- Subscribe to Your Spot Instance Data Feed (p. 261)

Cancel Spot Instance Requests

Each Spot Instance request can be in one of the following states:

- **Open—**The request is not fulfilled.
- **Active—**The request is currently active (fulfilled).
- **Closed—**The request either completed or was not fulfilled within the period specified.

  Depending on conditions, an instance might still be running.
- **Cancelled—**The request is canceled because one of two events took place: You canceled the request, or the bid request went past its expiration date.

  In addition, when your Spot Instance request is canceled, either because the bid request went beyond its expiration date or you manually canceled it, the Spot Instances that were launched previously through the now-canceled request don't automatically get terminated. The only time that the Spot Instance service terminates a running instance is when the Spot Price equals or exceeds your price. Also, you can always terminate your instances manually.

  Note

  The **Valid Until** option that you specify when you submit a Spot Instance request applies only to the *request*; this deadline doesn't apply to Spot Instances that are *launched* by the request.

After placing a request, you can check its state and the state of your other requests.

Note

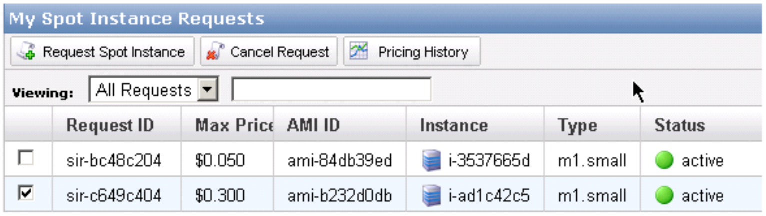
Make sure you have set up the prerequisites for working with Amazon EC2. If you haven't, go to Prerequisites for Using Spot Instances (p. 238).

AWS Management Console

To cancel Spot Instance requests

1. From the AWS Management Console for EC2, click **Spot Requests** in the navigation pane.

The console displays a list of Spot Instances requests.



	Request ID	Max Price	AMI ID	Instance	Type	Status
□	sir-bc48c204	$0.050	ami-84db39ed	i-3537665d	m1.small	● active
☑	sir-c649c404	$0.300	ami-b232d0db	i-ad1c42c5	m1.small	● active

2. Select the spot instances you want to cancel and click the **Cancel Request** button.

Command Line Tools

To cancel Spot Instance requests

1. Enter the following command:

```
PROMPT> ec2-describe-spot-instance-requests
```

Amazon EC2 returns output similar to the following:

```
SPOTINSTANCEREQUEST   sir-09fb0a04   0.04   one-time   Linux/UNIX   closed

2010-04-06T10:03:09+0200   ami-b232d0db   m1.small   gsg-keypair   default
  monitoring-disabled
```

2. To cancel the request, enter the following:

```
PROMPT> ec2-cancel-spot-instance-requests sir-09fb0a04
```

Amazon EC2 returns output similar to the following:

```
SPOTINSTANCEREQUEST     sir-09fb0a04    cancelled
```

Tip

You can filter the list of Spot Instance requests to return only certain types of interest to you. For more information about how to filter the results, go to ec2-describe-spot-instance-requests in the *Amazon Elastic Compute Cloud Command Line Reference*.

API

To cancel Spot Instance requests

1. Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeSpotInstanceRequests
&...auth parameters...
```

Following is an example response.

```
<DescribeSpotInstanceRequestsResponse xmlns="http://ec2.amazon
aws.com/doc/2011-12-01/">
  <spotInstanceRequestSet>
      <item>
         <spotInstanceRequestId>sir-8675309a</spotInstanceRequestId>
         <spotPrice>0.32</spotPrice>
         <type>one-time</type>
         <state>open</state>
         <fault/>
         <validFrom/>
         <validUntil/>
         <launchGroup/>
         <availabilityZoneGroup>MyAzGroup</availabilityZoneGroup>
         <launchSpecification>
            <imageId> i-43a4412a</imageId>
            <keyName>MyKeypair</keyName>
            <groupSet>websrv</groupSet>
            <instanceType>m1.small</instanceType>
         </launchSpecification>
         <instanceId>i-123345678</instanceId>
         <createTime>2009-10-19T00:00:00+0000</createTime>
       <productDescription>Linux/UNIX</productDescription>
       <launchedAvailabilityZone>us-east-1c</launchedAvailabilityZone>
      </item>
   </spotInstanceRequestSet>
</DescribeSpotInstanceRequestsResponse>
```

2. Construct a Query request to cancel the Spot Instance requests.

```
https://ec2.amazonaws.com/
?Action=CancelSpotInstanceRequests
&SpotInstanceRequestId.1=sir-8675309a
&...auth parameters...
```

Following is an example response.

```
<CancelSpotInstanceRequestsResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <spotInstanceRequestId>sir-8675309a</spotInstanceRequestId>
</CancelSpotInstanceRequestsResponse>
```

Tip

You can filter the list of Spot Instance requests to return only certain types of interest to you. For more information about how to filter the results, go to DescribeSpotInstanceRequests in the *Amazon Elastic Compute Cloud API Reference*.

What do you want to do next?

- Create a Spot Instance Request (p. 244)
- View Spot Price History (p. 241)

- Find Running Spot Instances (p. 249)
- Launch Spot Instances in Amazon Virtual Private Cloud (p. 259)

Plan for Interruptions

Because Spot Instances can terminate at any time, applications that run on Spot Instances must terminate cleanly. Although we attempt to cleanly terminate your instances, your application should be prepared to deal with an immediate shutdown.

To test your application, launch and terminate it as an on-demand instance.

⚠ **Important**

Although Spot Instances can use Amazon EBS-backed AMIs, be aware that the EBS-backed AMIs don't support Stop/Start. In other words, you can't stop and start Spot Instances launched from an AMI with an Amazon EBS root device.

Quick Look: Managing Interruptions Video

The following video shows how some customers manage the interruption of their Spot instances. How to Manage Spot Instance Interruptions

For more information about strategies to manage interruptions, go to the following sections:

- Launching Amazon Elastic MapReduce Job Flows with Spot Instances (p. 256)
- Starting Clusters on Spot Instances (p. 256)

Launching Amazon Elastic MapReduce Job Flows with Spot Instances

You can launch an Amazon Elastic MapReduce job flow in Spot Instances. Amazon Elastic MapReduce is a data analysis tool that simplifies the set up and management of a computer cluster, the source data, and the computational tools that help you implement sophisticated data processing jobs quickly. For more information, see Introduction to Amazon Elastic MapReduce.

Quick Look: Using Spot Instances with Amazon Elastic MapReduce Video

The following video describes how Spot Instances work in Amazon Elastic MapReduce and walks you through the process of launching a job flow on Spot Instances from the AWS Management Console: Using Spot Instances with Amazon ElasticMapReduce

Starting Clusters on Spot Instances

Grids are a form of distributed computing that enable a user to leverage multiple instances to perform parallel computations. Customers—such as Numerate, Scribd, and the University of Barcelona/University of Melbourne—use Grid Computing with Spot Instances because this type of architecture can take advantage of Spot Instance’s built-in elasticity and low prices to get work done faster at a more cost-effective price.

To get started, a user will break down the work into discrete units called jobs, and then submit that work to a “master node.” These jobs will be queued up, and a process called a “scheduler” will distribute that work out to other instances in the grid, called “worker nodes.” After the result is computed by the worker node, the master node is notified, and the worker node can take the next operation from the queue. If the job fails or the instance is interrupted, the job will automatically be re-queued by the scheduler process.

As you work to architect your application, it is important to choose the appropriate amount of work to be included in your job. We recommend breaking your jobs down into a logical grouping based on the time it would take to process. Typically, you will want to create a workload size less than an hour, so that if you have to process the workload again, it doesn't cost you additional money (you don't pay for the hour if we interrupt your instance).

Many customers use a Grid scheduler, such as Oracle Grid Engine or UniCloud, to set up a cluster. If you have long-running workloads, the best practice is to run the master node on On-Demand or Reserved Instances, and run the worker nodes on Spot or a mixture of On-Demand, Reserved, and Spot Instances. Alternatively, if you have a workload that is less than an hour or you are running a test environment, you may want to run all of your instances on Spot. No matter the setup, we recommend that you create a script to automatically re-add instances that may be interrupted. Some existing tools—StarCluster, for example— can help you manage this process.

Quick Look: How to Launch a Cluster on Spot Video

Chris Dagdigian, from AWS Solution Provider BioTeam, provides a quick overview of how to start a cluster from scratch in about 10 to 15 minutes on Amazon EC2 Spot Instances using StarCluster. StarCluster is an open source tool created by a lab at MIT that makes it easy to set up a new Oracle Grid Engine cluster. In this video, Chris walks through the process of installing, setting up, and running simple jobs on a cluster. Chris also leverages Spot Instances, so that you can potentially get work done faster and potentially save between 50 percent to 66 percent. How to Launch a Cluster on Spot

Bidding Strategies

Depending on how soon you want your Spot Instance request fulfilled and how long you want your Spot Instances to continue running, there are a number of factors you need to consider when you make a bid for Spot Instances. The requirements you include in your Spot Instance request can affect the chances that it will be fulfilled. If you have a number of requirements that must be met, your request can take a little longer to fulfill.

The maximum Spot Instance price impacts bid fulfillment and how long your Spot Instances run, as well. If you want your instances to run continuously for longer periods, select a higher price. If your instances don't need to run continuously, select a lower price.

The maximum Spot Instance price you specify is not necessarily the price that you pay. For example, if you specify $.50 as your maximum price, and the Spot Price is $.30 for the period, you pay $.30. If the Spot Price drops, you pay less. If the Spot Price increases, you pay the new price (up to your specified maximum price).

Quick Look: Bidding Strategies Video

The following video describes strategies to use when bidding for Spot Instances. Deciding on Your Spot Bidding Strategies

Advanced Tasks

Topics

Now that you have created Spot Instance requests and worked with Spot Instances, this section discusses some advanced tasks.

Persist Your Root EBS Partition

Amazon Elastic Block Store (Amazon EBS) can be an effective way to store data that you otherwise might lose when your Spot Instance terminates.

To set up the persistence of Spot Instance data, you map the Spot Instances that will be launched to an existing Amazon Elastic Block Store (Amazon EBS) snapshot. Set the *`delete-on-termination`* flag to `false`; this indicates that Amazon EC2 shouldn't delete the Amazon EBS volume when the spot instance terminates.

Let's walk through making an example Spot Request with the following specifications:

- Bid price of $0.5
- One instance of the m1.xlarge instance type
- Block device mapping to a snapshot that shouldn't be deleted when the Spot Instance is terminated

You can do this example using either the CLI or API tools. Using the CLI, your example request should look like this:

```
PROMPT> ec2-request-spot-instances -p 0.5 -t m1.xlarge -n 1 -b '/dev/sdb=snap-a123bcde:20:false' ami-8e1fece7
```

For more information, see:

- Block Device Mapping (p. 167)
- Basics of Amazon EBS-Backed AMIs and Instances (p. 177)
- ec2-request-spot-instances
- RequestSpotInstances

Tag Spot Instance Requests

To help categorize and manage your Spot Instance requests, you can tag them with metadata of your choice. You tag your Spot Instance requests in the same way that you tag other Amazon EC2 resources. Create a tag by specifying a key and value, and assigning the pair to the Spot Instance request. You use the CreateTags action or the ec2-create-tags command. For information about how to use tags, see Using Tags (p. 275).

The tags you create for your Spot Instance requests only apply to the requests. These tags don't propagate to the instances that the requests launch. To categorize the Spot Instances that are launched from your tagged request, you must create a separate set of tags for the instances using the same commands.

For example, you have Spot Instance request *sir-69047e11* and you want to label it with the tag `Spot=Test`. To do this, use the following command:

```
PROMPT> ec2-create-tags sir-69047e11 --tag "Spot=Test"
```

Amazon EC2 returns this information:

```
TAG     spot-instance-request     sir-69047e11    Spot    Test
```

You can also confirm the tag information by using the `ec2-describe-tags` command.

When your request is fulfilled and a Spot Instance launches, you will see that the tag you used for your Spot Instance request is not applied to the Spot Instance. In the following example command, we are obtaining information about the Spot Instance *i-b8ca48d8* that was launched as a result of your Spot Instance request *sir-69047e11*, tagged Spot=Test.

```
PROMPT> ec2-describe-instances i-b8ca48d8
```

The call returns details about the instance with no tag information, showing that the tag for your Spot Instance request does not apply to the Spot Instance that the request launched. To tag your Spot Instance, use the following command:

```
PROMPT> ec2-create-tags i-b8ca48d8 --tag "SpotI=Test1"
```

Amazon EC2 returns this information:

```
TAG     instance     i-b8ca48d8    SpotI    Test1
```

You can create similar tags using the API. For more information, see the API section of Using Tags in the *Amazon Elastic Compute Cloud User Guide*.

Launch Spot Instances in Amazon Virtual Private Cloud

Amazon Virtual Private Cloud (Amazon VPC) lets you define a virtual networking environment in a private, isolated section of the Amazon Web Services (AWS) cloud. You have complete control of this virtual network and you can use advanced security features and network access control at the instance level and subnet level. For more information about Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

If you want to take advantage of the features of Amazon VPC when you use Spot Instances, specify in your Spot Request that your instances are to be launched in Amazon VPC. Before you can specify that your Spot Instances are to be launched in Amazon VPC, you must set up a VPC. For information about setting up an Amazon VPC, see Getting Started with Amazon VPC in the *Amazon Virtual Private Cloud Getting Started Guide*.

Quick Look: Launching Spot Instances in Amazon VPC Video

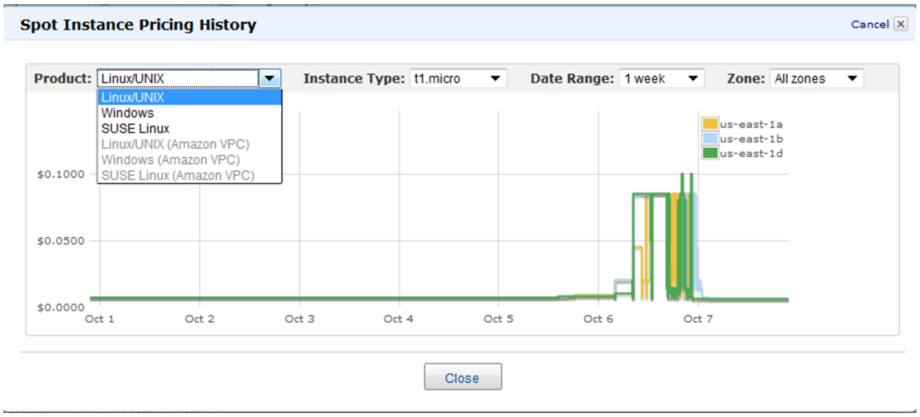
The following video shows how to launch your first Spot Instance in the Amazon Virtual Private Cloud (Amazon VPC) using the AWS Management Console. This video includes instructions for creating your Amazon VPC subnet, placing a bid, determining when the instance is fulfilled, and canceling the instance. Launching Spot Instances in Amazon VPC Video

The process for making a Spot Instance request that launches in Amazon VPC is the same as the process you follow when you make a Spot Instance request in a non-VPC portion of Amazon EC2. The main differences are that you:

- Base your Spot Price bid on Spot Price history of Spot Instances in VPCs.
  When you use the `DescribeSpotPriceHistory` action or the `ec2-describe-spot-price-history` command, add *Amazon VPC* to the *`product-description`* filter. For example:

```
PROMPT> ec2-describe-spot-price-history -s '2011-09-08T00:00:00Z' -t m1.xlarge
 -d "Linux/UNIX (Amazon VPC)"
```

Using the Amazon EC2 dashboard in the AWS Management Console, check the **Spot Price History** page to see the Spot pricing history for Amazon EC2 instances running in both Amazon EC2 and Amazon VPC.



Spot Instance Pricing History

Product: Linux/UNIX

Instance Type: t1.micro

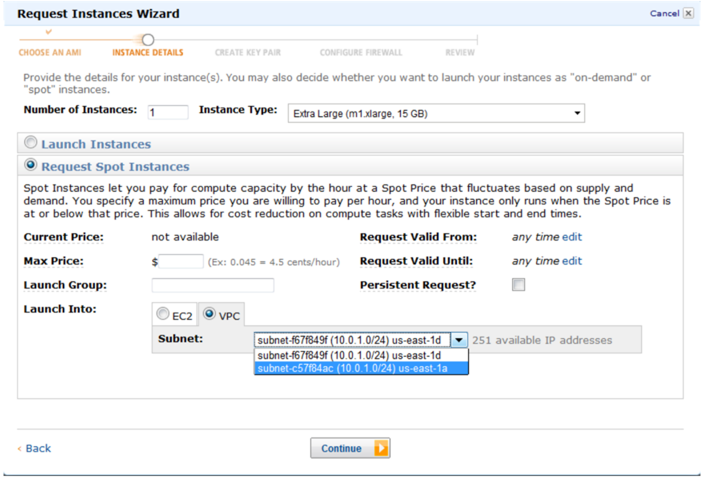
Date Range: 1 week

Zone: All zones

- Specify the VPC subnet in which you want to launch your Spot Instance.
  When you use the `RequestSpotInstances` action or the `ec2-request-spot-instances` command, specify the ID of the Amazon VPC subnet in which you want to launch the Spot Instance.

```
PROMPT> ec2-request-spot-instances ami-8e1fece7 -t m1.xlarge -p '$0.01' -n 5
 -r 'one-time' -s 'subnet-baab943d3'
```

When you launch the **Request Instances Wizard** from the Spot Instance page of the EC2 dashboard in the AWS Management Console, select a subnet after specifying that you're launching the Spot Instance into a VPC.



Request Instances Wizard

Provide the details for your instance(s). You may also decide whether you want to launch your instances as "on-demand" or "spot" instances.

Number of Instances: 1   Instance Type: Extra Large (m1.xlarge, 15 GB)

○ Launch Instances

● Request Spot Instances

Spot Instances let you pay for compute capacity by the hour at a Spot Price that fluctuates based on supply and demand. You specify a maximum price you are willing to pay per hour, and your instance only runs when the Spot Price is at or below that price. This allows for cost reduction on compute tasks with flexible start and end times.

Current Price: not available

Request Valid From: *any time* edit

Max Price: $ (Ex: 0.045 = 4.5 cents/hour)

Request Valid Until: *any time* edit

Launch Group:

Persistent Request? □

Launch Into: ○ EC2 ● VPC

Subnet: subnet-f67f849f (10.0.1.0/24) us-east-1d   251 available IP addresses

subnet-f67f849f (10.0.1.0/24) us-east-1d
subnet-c57f84ac (10.0.1.0/24) us-east-1a

Note

The **Request Spot Instances** page will show the **VPC** option only if you have created a VPC and subnet before making the Spot Instance request.

For more information about using Amazon VPC, go to the Amazon Virtual Private Cloud User Guide.

Subscribe to Your Spot Instance Data Feed

If you want to monitor your Spot Instance usage, you can subscribe to your Spot Instance data feed, which stores usage logs in Amazon Simple Storage Service (Amazon S3).

This section describes the data feed content and how to create the data feed for Spot Instances. You can create one data feed per account.

Spot Instance Data Feed Overview

To help you understand the charges for your Spot Instances, Amazon EC2 provides access to a data feed that details your Spot Instance usage and pricing. This data feed is sent to the Amazon S3 bucket of your choice.

To have a data feed delivered to an Amazon S3 bucket, you need to create a Spot Instances data feed subscription using the Amazon EC2 API. When you create this subscription, you can specify an Amazon S3 bucket to deliver the data feed files to, and a filename prefix to use to avoid collisions.

Data Feed Filename and Format

The Spot Instance data feed filename uses the following format (with the date and hour in UTC):

```
{Bucket}.s3.amazonaws.com/{Optional Prefix}/{AWS Account ID}.{YYYY}-{MM}-{DD}-
{HH}.{n}.{Unique ID}.gz
```

For example, if your bucket name is `mydata`, and you name your prefix `myprefix`, your filenames look similar to this:

```
mydata.s3.amazonaws.com/myprefix/999988887777.2010-03-17-20.001.pwBdGTJG.gz
```

Data feed files arrive in your bucket typically once an hour and each hour of usage is typically covered in a single data file. These files are compressed (gzip) before delivery into your bucket. We can write multiple files for a given hour of usage where files are very large (for example, when file contents for the hour exceed 50 MB before compression).

Note

If you don't have a Spot Instance running during a certain hour, you won't receive a data feed file for that hour.

The Spot Instance data feed files are tab-delimited. Each line in the data file corresponds to one instance-hour. Each line contains the fields listed in the following table.

Field	Description
`Timestamp`	The timestamp used to determine the price charged for this instance-hour.
`UsageType`	Indicates the type of usage and instance type being charged for. For m1.small Spot Instances, this field is set to "SpotUsage." For all other instance types, this field is set to "SpotUsage:{instance-type}," for example, “SpotUsage:c1.medium.”

Field	Description
`Operation`	Indicates the product being charged for. For Linux/UNIX Spot Instances, this field is set to “RunInstances.” For Microsoft Windows, this field is set to "RunInstances:0002." Spot usage is grouped according to Availability Zone.
`InstanceID`	The instance ID for the Spot Instance that generated this instance-hour.
`MyBidID`	The Spot Instance request ID for the request that generated this instance-hour.
`MyMaxPrice`	The maximum price specified for this Spot Instance request.
`MarketPrice`	The Spot Price at the time specified in the Timestamp field.
`Charge`	The price charged for this instance-hour.
`Version`	The version included in the data feed filename for this record.

Preparing Amazon S3 for Data Feeds

When you subscribe to data feeds, you tell Amazon EC2 which bucket you want to store the data feed file in. Before you subscribe to the data feed, consider the following when choosing your S3 bucket:

- You must have Amazon S3 FULL_CONTROL permission on the bucket you provide.
  If you're the bucket owner, you have this permission by default. If you're not the bucket owner, the bucket owner must grant your AWS account FULL_CONTROL permission.
- When you create your data feed subscription, Amazon EC2 updates the designated bucket's ACL to allow read and write permissions for the AWS data feeds account.
- Each data feed file has its own ACL (separate from the bucket's ACL).
  The bucket owner has FULL_CONTROL permission for the data files. The data feed account has read and write permission.
- Removing the permissions for the data feed account does not disable the data feed.
  If you remove those permissions but don't disable the data feed (which you do with the control API), we reinstate those permissions the next time the data feeds account needs to write a data file to your bucket.
- If you delete your data feed subscription, Amazon EC2 doesn't remove the read/write permissions for the data feed account on either the bucket or the data files.
  You must perform remove the read/write permissions yourself.

Subscribe to Your Spot Instance Data Feed

Command Line Tools

To subscribe to your Spot Instance data feed

- Enter the following command:

```
PROMPT>  ec2-create-spot-datafeed-subscription --bucket bucket [--prefix prefix
 ]
```

Amazon EC2 returns output similar to the following:

```
SPOTDATAFEEDSUBSCRIPTION        999988887777    bucket        prefix   Active
```

API

To subscribe to your Spot Instance data feed

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateSpotDatafeedSubscription
&Bucket=my-bucket
&Prefix=my-spot-subscription
&...auth parameters...
```

Following is an example response.

```
<CreateSpotDatafeedSubscriptionResponse xmlns="http://ec2.amazon
aws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <spotDatafeedSubscription>
        <ownerId>999988887777</ownerId>
        <bucket>my-bucket</bucket>
        <prefix>my-spot-subscription</prefix>
        <state>Active</state>
        <fault/></fault>
   </spotDatafeedSubscription>
</CreateSpotDatafeedSubscriptionResponse>
```

Delete a Spot Instance Data Feed

Command Line Tools

To delete a Spot Instance data feed

- To delete a data feed, enter the following command:

```
PROMPT>  ec2-delete-spot-datafeed-subscription
```

If the request is successful, the output is empty.

API

To delete a Spot Instance data feed

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DeleteSpotDatafeedSubscription
&...auth parameters...
```

Following is an example response. It confirms that the subscription was deleted.

```
<DeleteSpotDatafeedSubscriptionResponse xmlns="http://ec2.amazon
aws.com/doc/2011-12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <return>true</return>
</DeleteSpotDatafeedSubscriptionResponse>
```

Failure Resilient Application Concepts

You need to think about is how to design your application for *failure resilience*. It's inevitable that EC2 instances will fail, and you need to plan for it. An instance failure isn't a problem if your application is designed to handle it. This section points to references and sections in this guide that will help you understand how to design your application for fault tolerance.

Whitepapers

AWS solutions architects and evangelists have written white papers to help you think about designing your application. For more information, go to AWS Cloud Computing Whitepapers.

Features of EC2 for Failure Resilience

Within this guide, you should read the following sections to understand features of EC2 that help you build fault tolerant applications:

- Using Elastic IP Addresses (p. 292)
- Amazon Elastic Block Store (p. 126)
- Auto-Scaling and Load Balancing Your Instances (p. 345)
- Monitoring Your Instances and Volumes with CloudWatch (p. 348)

Amazon EC2 Credentials

Topics

Types of Credentials You Need

Amazon EC2 uses a variety of credentials for different purposes. This section describes major tasks you might perform with EC2 and the credentials required to perform them.

Credentials to Access Your Account

To sign up for services, view your bills, perform account-based tasks, and get many of your security credentials, you will need your standard Amazon login and password. For more information, see How to Log In with Your Amazon Login and Password (p. 266).

Note

We also provide AWS Multi-Factor Authentication, which requires a physical device and passcode to login to your AWS account. For more information, go to http://aws.amazon.com/mfa.

Credentials to Launch and Administer Amazon EC2 Instances

The credentials you need to launch and administer instances depend on the interface you're using.

AWS Management Console

To launch and administer Amazon EC2 instances through the AWS Management Console, you only need the Amazon login and password. For more information, see How to Log In with Your Amazon Login and Password (p. 266).

Query and Third-Party UI Tools

Launching and administering Amazon EC2 instances through the Query API and many UI-based tools (e.g., ElasticFox) requires the Access Key ID and Secret Access Key. For more information, see How to Get Your Access Key ID and Secret Access Key (p. 267).

SOAP and EC2 Command Line Tools

Launching and administering Amazon EC2 instances through the SOAP API and command line interface (i.e., API tools) requires an X.509 certificate and private key, which can be generated by AWS. For more information, see How to Create an X.509 Certificate and Private Key (p. 267).

Credentials to Connect to Amazon EC2 Instances

To connect to your instances, you need the following:

- **Amazon EC2 Key Pair—**Enables you to connect to Linux/UNIX instances through SSH. For more information, see SSH Key Pair (p. 268).
- **Windows administrator password (Windows only)—**Provides the "first-use" password that enables you to connect to a Windows instance through Remote Desktop. For more information, see Windows Administrator Password (p. 268).

Credentials Needed for Sharing with Others

To enable other AWS accounts to use your Amazon EC2 AMIs and Amazon EBS snapshots, you need their AWS Account IDs. For information on how to get the AWS Account ID associated with your account, see Viewing Your Account ID (p. 268).

Credentials Needed for Bundling Amazon EC2 instance store-backed Instances

To bundle Amazon EC2 instance store-backed Linux/UNIX instances, you need your AWS Account ID, and your X.509 certificate and private key. For information about viewing your Account ID, see Viewing Your Account ID (p. 268). For information about getting an X.509 certificate and private key, see How to Create an X.509 Certificate and Private Key (p. 267).

To bundle Amazon EC2 instance store-backed Windows instances, you need your Amazon login and password to access the AWS Management Console. For information about the Amazon login and password, see How to Log In with Your Amazon Login and Password (p. 266).

How to Log In with Your Amazon Login and Password

The Amazon login and password enable you to sign up for services, view your bills, perform account-based tasks, and get many of your security credentials. You also use the login and password to perform Amazon EC2 tasks through the AWS Management Console.

This section describes how to log in with your login and password.

To log in with your login and password (if you have an existing account)

1. Go to the AWS Web Site.
2. Select an option from the Your Account menu. The **Amazon Web Services Sign In** page appears.
3. Enter your email address, select I am a returning user and my password is, enter your password, and click the **Sign In** button.

To get a new Amazon login and password (create a new AWS account)

1. Go to the AWS Web Site.
2. Click **Create an AWS Account**.

   The **Amazon Web Services Sign In** page appears.
3. Select I am a new user and click the **Sign In** button.
4. Follow the on-screen prompts to create a new account.

Note

It is important to keep your Amazon login and password secret as they can be used to view and create new credentials. As an increased security measure, we offer Multi-Factor Authentication, which uses the combination of a physical device and passcode to login to your AWS account. For more information, go to http://aws.amazon.com/mfa.

How to Get Your Access Key ID and Secret Access Key

The Access Key ID and Secret Access Key are the most commonly used set of AWS credentials. They are used to make Query and REST-based requests and are commonly used by UI-based tools, such as ElasticFox. You can use up to two sets of Access Keys at a time. You can generate new keys at any time or disable existing keys.

To get your Access Key ID and Secret Access Key

1. Go to the AWS Web Site.
2. Point to **Your Account** and select **Security Credentials**.
   If you are not already logged in, you are prompted to do so.
3. Scroll down to the **Access Credentials** section and verify the **Access Keys** tab is selected.
4. Locate an active Access Key in the **Your Access Keys** list.
5. To display the Secret Access Key, click **Show** in the **Secret Access Key** column.
6. Write down the keys or save them.
7. If there are no Access Keys in the list, click **Create a New Access Key** and follow the on-screen prompts.

How to Create an X.509 Certificate and Private Key

The X.509 Certificate and Private Key are used by the command line tools and the SOAP API. You can download the private key file once. If you lose it, you will need to create a new certificate. Up to two certificates (active or inactive) are allowed at anytime.

This section describes how to create a new certificate.

To create a certificate

1. Go to the AWS Web Site.
2. Point to **Your Account** and select **Security Credentials**.
   If you are not already logged in, you are prompted to do so.
3. Click the **X.509 Certificates** tab
4. Click **Create a New Certificate** and follow the on-screen prompts.
   The new Certificate is created and appears in the X.509 Certificate list. You are prompted to download the certificate and private key files.
5. Create a .ec2 directory in your home directory, and save these files to it with the filenames offered by your browser.
   You should end up with a PEM-encoded X.509 certificate and a private key file.

SSH Key Pair

You must create an RSA public/private key pair, which you use to ensure that only you have access to instances that you launch.

You have two options for getting this key pair:

- Generate it yourself with a third-party tool such as OpenSSH, and then import the public key to AWS using either the `ec2-import-keypair` command or the `ImportKeyPair` action
- Have AWS generate the key pair for you using the AWS Management Console, the `ec2-add-keypair` command, or the `CreateKeyPair` action

With either option, AWS doesn't store a copy of the private key. Amazon EC2 only stores the public key, and associates it with a friendly key pair name you specify. Whenever you launch an instance using the key pair name, the public key is copied to the instance metadata. This allows you to access the instance securely using your private key.

For information on how to create key pairs, see Getting an SSH Key Pair (p. 77).

Windows Administrator Password

The Windows administrator password is used to access a Windows instance through Remote Desktop (RDP) for the first time only. If you change the password or rebundle the AMI, the instance will use the last set password. For information on how to get the Windows administrator password to connect to a Windows instance through RDP using the AWS Management Console, see Connecting to Windows Instances (p. 122).

Viewing Your Account ID

The Account ID identifies your account to AWS and enables other accounts to access resources that you want to share, such as Amazon EC2 AMIs and Amazon EBS snapshots.

To view your Account ID

1. Go to the AWS Web Site.
2. Point to **Your Account** and select **Security Credentials**.

   If you are not already logged in, you are prompted to do so.
3. Scroll down to the **Account Identifiers** section.
4. Locate your AWS Account ID.

For information on how to share AMIs, see Sharing AMIs Safely (p. 47). For information on how to share snapshots, see Modifying Snapshot Permissions (p. 143).

Note

The Account ID number is not a secret. When granting access to resources, make sure to specify the Account ID without hyphens.

The canonical user ID is used by Amazon S3.

Related Topics

- Bundling Amazon EC2 instance store-backed Windows AMIs (p. 21)
- Launching and Using Instances (p. 75)
- Connecting to Instances (p. 111)

Listing and Filtering Your Resources

Topics

As part of using Amazon EC2, you might use different *resources*. These includes images, instances, Amazon EBS volumes, snapshots, etc. You can list each type of resource with the corresponding *describe* action (e.g., `ec2-describe-images`, `ec2-describe-instances`). The resulting list can be long, so you might want to filter the results to include only resources that match certain criteria. For example, you could list only the public 64-bit Windows AMIs that use an Amazon EBS volume as the root device. You could list only instances that have an attached Amazon EBS volume that is set to delete whenever the instance terminates. You could list only Amazon EBS snapshots that have been tagged with a certain value.

The EC2 command line interface (also called the API tools) and API let you specify filters when listing resources. The specific filters you can use are listed in the topic covering the particular describe call. The examples that follow show some of the filters.

You can specify multiple values for a filter. For example, you can list all the instances whose type is either m1.small or m1.large. The resource must match at least one of the values to be included in the resulting resource list. You can also specify multiple filters. For example, you can list all the instances whose type is either m1.small or m1.large, and that have an attached EBS volume that is set to delete when the instance terminates. The instance must match all your filters to be included in the results.

You can also use wildcards with the filter values. An asterisk (*) matches zero or more characters, and a question mark (?) matches exactly one character. For example, you can use `*database*` as a filter value to get all EBS snapshots that include `database` in the description. If you were to specify `database` as the filter value, then only snapshots whose description equals `database` would be returned. Filter values are case sensitive. We support only exact string matching, or substring matching (with wildcards).

Note

Your search can include the literal values of the wildcard characters; you just need to escape them with a backslash before the character. For example, a value of `\*amazon\?\\` searches for the literal string `*amazon?\`.

Listing Resources

Each resource type has a corresponding *describe* action that you use to list your resources of that type. For example, you use `ec2-describe-images` or `ec2-describe-instances` to list your images or instances, respectively.

Command Line Tools

The following example lists your instances.

```
PROMPT> ec2-describe-instances
```

The response contains information about all your instances. Following is a sample response:

```
RESERVATION     r-c482cbaf      999988887777    default
INSTANCE        i-8f59c3e5      ami-c5e40dac    ec2-184-73-22-106.compute-
1.amazonaws.com       ip-10-194-250-85.ec2.internal   running   GSG_Keypair
0               m1.small        2010-08-17T14:26:03+0000        us-east-1d
       windows   monitoring-disabled     184.73.22.106   10.194.250.85
  ebs        hvm
BLOCKDEVICE     /dev/sda1       vol-6214560b    2010-08-17T14:26:07.000Z
RESERVATION     r-f0ddf49b      999988887777    default
INSTANCE        i-59d69933      ami-b232d0db    ec2-75-101-210-167.compute-
1.amazonaws.com      domU-12-31-39-0F-CD-55.compute-1.internal       running
     GSG_Keypair     0               m1.small        2010-09-03T16:09:09+0000
       us-east-1b  aki-94c527fd        ari-96c527ff            monitoring-dis
abled   75.101.210.167  10.193.206.163                  ebs     spot    sir-
e1aa2e03            paravirtual
BLOCKDEVICE     /dev/sda1       vol-28421e41    2010-09-03T16:09:16.000Z
```

API

The following Query example lists your instances.

```
https://ec2.amazonaws.com/?Action=DescribeInstances
&AuthParams
```

Here is a sample response to the list request:

```
<DescribeInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>98e3c9a4-848c-4d6d-8e8a-b1bdEXAMPLE</requestId>
  <reservationSet>
    <item>
      <reservationId>r-b27e30d9</reservationId>
      <ownerId>999988887777</ownerId>
      <groupSet>
        <item>
          <groupId>sg-2eac845a</groupId>
          <groupName>default</groupName>
        </item>
      </groupSet>
      <instancesSet>
        <item>
          <instanceId>i-c5cd56af</instanceId>
          <imageId>ami-1a2b3c4d</imageId>
          <instanceState>
            <code>16</code>
            <name>running</name>
          </instanceState>
         <privateDnsName>domU-12-31-39-10-56-34.compute-1.internal</privateDns
Name>
          <dnsName>ec2-174-129-165-232.compute-1.amazonaws.com</dnsName>
          <reason/>
          <keyName>GSG_Keypair</keyName>
          <amiLaunchIndex>0</amiLaunchIndex>
          <productCodes/>
          <instanceType>m1.small</instanceType>
          <launchTime>2010-08-17T01:15:18.000Z</launchTime>
          <placement>
            <availabilityZone>us-east-1b</availabilityZone>
            <groupName/>
          </placement>
          <kernelId>aki-94c527fd</kernelId>
          <ramdiskId>ari-96c527ff</ramdiskId>
          <monitoring>
            <state>disabled</state>
          </monitoring>
          <privateIpAddress>10.198.85.190</privateIpAddress>
          <ipAddress>174.129.165.232</ipAddress>
          <sourceDestCheck>true</sourceDestCheck>
          <groupSet>
             <item>
                <groupId>sg-2eac845a</groupId>
                <groupName>default</groupName>
             </item>
          </groupSet>
          <architecture>i386</architecture>
          <rootDeviceType>ebs</rootDeviceType>
          <rootDeviceName>/dev/sda1</rootDeviceName>
          <blockDeviceMapping>
            <item>
              <deviceName>/dev/sda1</deviceName>
              <ebs>
                <volumeId>vol-a082c1c9</volumeId>
                <status>attached</status>
                <attachTime>2010-08-17T01:15:21.000Z</attachTime>
                <deleteOnTermination>false</deleteOnTermination>
              </ebs>
            </item>
          </blockDeviceMapping>
          <instanceLifecycle>spot</instanceLifecycle>
          <spotInstanceRequestId>sir-7a688402</spotInstanceRequestId>
          <virtualizationType>paravirtual</virtualizationType>
          <clientToken/>
          <tagSet/>
          <hypervisor>xen</hypervisor>
       </item>
      </instancesSet>
      <requesterId>854251627541</requesterId>
    </item>
    <item>
      <reservationId>r-b67e30dd</reservationId>
      <ownerId>999988887777</ownerId>
      <groupSet>
        <item>
           <groupId>sg-2eac845a</groupId>
          <groupId>default</groupId>
        </item>
      </groupSet>
      <instancesSet>
        <item>
          <instanceId>i-d9cd56b3</instanceId>
          <imageId>ami-1a2b3c4d</imageId>
          <instanceState>
            <code>16</code>
            <name>running</name>
          </instanceState>
```

```
         <privateDnsName>domU-12-31-39-10-54-E5.compute-1.internal</privateDns
Name>
          <dnsName>ec2-184-73-58-78.compute-1.amazonaws.com</dnsName>
          <reason/>
          <keyName>GSG_Keypair</keyName>
          <amiLaunchIndex>0</amiLaunchIndex>
          <productCodes/>
          <instanceType>m1.large</instanceType>
          <launchTime>2010-08-17T01:15:19.000Z</launchTime>
          <placement>
            <availabilityZone>us-east-1b</availabilityZone>
            <groupName/>
          </placement>
          <kernelId>aki-94c527fd</kernelId>
          <ramdiskId>ari-96c527ff</ramdiskId>
          <monitoring>
            <state>disabled</state>
          </monitoring>
          <privateIpAddress>10.198.87.19</privateIpAddress>
          <ipAddress>184.73.58.78</ipAddress>
          <sourceDestCheck>true</sourceDestCheck>
          <groupSet>
             <item>
                <groupId>sg-2eac845a</groupId>
                <groupName>default</groupName>
             </item>
          </groupSet>
          <architecture>i386</architecture>
          <rootDeviceType>ebs</rootDeviceType>
          <rootDeviceName>/dev/sda1</rootDeviceName>
          <blockDeviceMapping>
            <item>
              <deviceName>/dev/sda1</deviceName>
              <ebs>
                <volumeId>vol-a282c1cb</volumeId>
                <status>attached</status>
                <attachTime>2010-08-17T01:15:23.000Z</attachTime>
                <deleteOnTermination>false</deleteOnTermination>
              </ebs>
            </item>
          </blockDeviceMapping>
          <instanceLifecycle>spot</instanceLifecycle>
          <spotInstanceRequestId>sir-55a3aa02</spotInstanceRequestId>
          <virtualizationType>paravirtual</virtualizationType>
          <clientToken/>
          <tagSet/>
          <hypervisor>xen</hypervisor>
       </item>
      </instancesSet>
      <requesterId>854251627541</requesterId>
    </item>
  </reservationSet>
</DescribeInstancesResponse>
```

Filtering Resources

You can reduce the results of a *describe* action by adding one or more filters to the request.

Command Line Tools

The following example uses filters to list only your volumes that are in the us-east-1b Availability Zone and have a status of `available`.

```
PROMPT>  ec2-describe-volumes --filter availability-zone=us-east-1b --filter
status=available
```

Note

If you're using the command line tools on a Windows system, you might need to use quotation marks (i.e., `--filter "status=available"`).

The following example uses filters to list only public 64-bit Windows AMIs that use an Amazon EBS volume as the root device.

```
PROMPT> ec2-describe-images --filter is-public=true --filter architecture=x86_64
 --filter platform=windows
```

The following example uses filters to list only your m1.small or m1.large instances that have an attached Amazon EBS volume that is set to delete on instance termination.

```
PROMPT> ec2-describe-instances --filter instance-type=m1.small --filter instance-
type=m1.large --filter block-device-mapping.status=attached --filter block-
device-mapping.delete-on-termination=true
```

For an example of filtering resources by tags, see Using Tags (p. 275).

For a list of the available filters for a given EC2 resource, go to the *describe* topic for that resource in the Amazon Elastic Compute Cloud Command Line Reference.

API

The following example Query request uses filters to list only your volumes that are in the us-east-1b Availability Zone and have a status of `available`.

```
https://ec2.amazonaws.com/?Action=DescribeVolumes
&Filter.1.Name=availability-zone
&Filter.1.Value.1=us-east-1b
&Filter.2.Name=status
&Filter.2.Value.1=available
&AuthParams
```

The following example Query request uses filters to list only public 64-bit Windows AMIs that use an Amazon EBS volume as the root device.

```
https://ec2.amazonaws.com/?Action=DescribeImages
&Filter.1.Name=is-public
&Filter.1.Value.1=true
&Filter.2.Name=architecture
&Filter.2.Value.1=x86_64
&Filter.3.Name=platform
```

```
&Filter.3.Value.1=windows
&AUTHPARAMS
```

The following example Query request uses filters to list only the m1.small or m1.large instances that have an attached Amazon EBS volume that is set to delete on instance termination.

```
https://ec2.amazonaws.com/?Action=DescribeInstances
&Filter.1.Name=instance-type
&Filter.1.Value.1=m1.small
&Filter.1.Value.2=m1.large
&Filter.2.Name=block-device-mapping.status
&Filter.2.Value.1=attached
&Filter.3.Name=block-device-mapping.delete-on-termination
&Filter.3.Value.1=true
&AUTHPARAMS
```

For an example of filtering resources by tags, see Using Tags (p. 275).

For a list of the available filters for a given EC2 resource, go to the *describe* topic for that resource in the Amazon Elastic Compute Cloud API Reference.

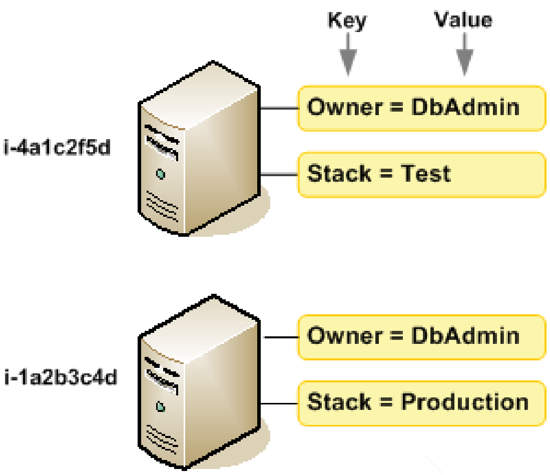
Using Tags

Topics

To help you manage your Amazon EC2 instances, images, and other Amazon EC2 resources, you can assign your own metadata to each resource in the form of *tags*. This section describes tags and how to create them.

Tag Basics

Each EC2 tag consists of a key and a value, both of which you define. For example, you could define a set of tags for your account's instances that helps you track each instance's owner and stack level. The following diagram illustrates the concept. In the diagram, you've assigned two tags to each of your instances, one called *Owner* and another called *Stack*. Each of the tags also has an associated value.



Tags let you categorize your EC2 resources in different ways, e.g., by purpose, owner, environment, etc. We recommend you devise a set of tag keys that meets your needs for each resource type. Using a consistent set of tag keys will make it easier to manage your resources. You can search and filter the resources based on the tags.

AWS doesn't apply any semantic meaning to your tags; they're interpreted strictly as strings of characters. AWS doesn't automatically set any tags on resources.

Note

You must not prefix your tag names or values with `aws:`. Tags prefixed with `aws:` have been created by AWS and cannot be edited or deleted.

You can assign tags only to resources that already exist. Exception: You can add tags when you launch an instance in the AWS Management Console. If you add a tag that has the same key as an existing tag on that resource, the new value overwrites the old value. You can edit tag keys and values, and you can remove tags from a resource at any time. You can set a tag's value to the empty string, but you can't set a tag's value to null.

If you're using AWS Identity and Access Management (IAM), you can control which Users in your AWS Account have permission to create, edit, or delete tags. For more information about IAM, see AWS Identity and Access Management (p. 10).

Tag Restrictions

The following basic restrictions apply to tags:

- **Maximum Number of Tags Per Resource—**10
- **Maximum Key Length—**128 Unicode characters
- **Maximum Value Length—**256 Unicode characters
- **Unavailable Prefix—**`aws:` (we have reserved it for tag names and values)

Tag keys and values are case sensitive.

You can't terminate, stop, or delete a resource based solely on its tags; you must specify the resource identifier instead (e.g., i-1a2b3c4d). For example, to delete snapshots that have been tagged with a tag key called `DeleteMe`, you must first get a list of the those snapshots by using a `DescribeSnapshots` call with a filter on the tag. Then you would call `DeleteSnapshots` with the IDs of the snapshots (e.g., snap-1a2b3c4d). You can't call `DeleteSnapshots` with a filter on the tag. For more information about using filters when listing your resources, see Listing and Filtering Your Resources (p. 270).

An instance can't retrieve its own tags from its metadata (for more information about metadata, see Using Instance Metadata (p. 96)). Tag information is available only through the EC2 API.

Resources You Can Tag

You can tag the following Amazon EC2 and Amazon VPC resources:

- images (AMIs, kernels, RAM disks)
- instances
- security groups
- Amazon EBS volumes
- Amazon EBS snapshots
- Reserved Instances
- Spot Instance requests
- VPCs
- subnets
- Internet gateways
- VPN connections
- virtual private gateways
- customer gateways
- route tables
- network ACLs
- DHCP options sets

Following are the Amazon EC2 resources you cannot tag:

- Elastic IP addresses
- Key pairs
- Placement groups

You can tag public or shared resources, but the tags you assign are available only to your AWS account and not to the other accounts sharing the resource.

AWS Management Console

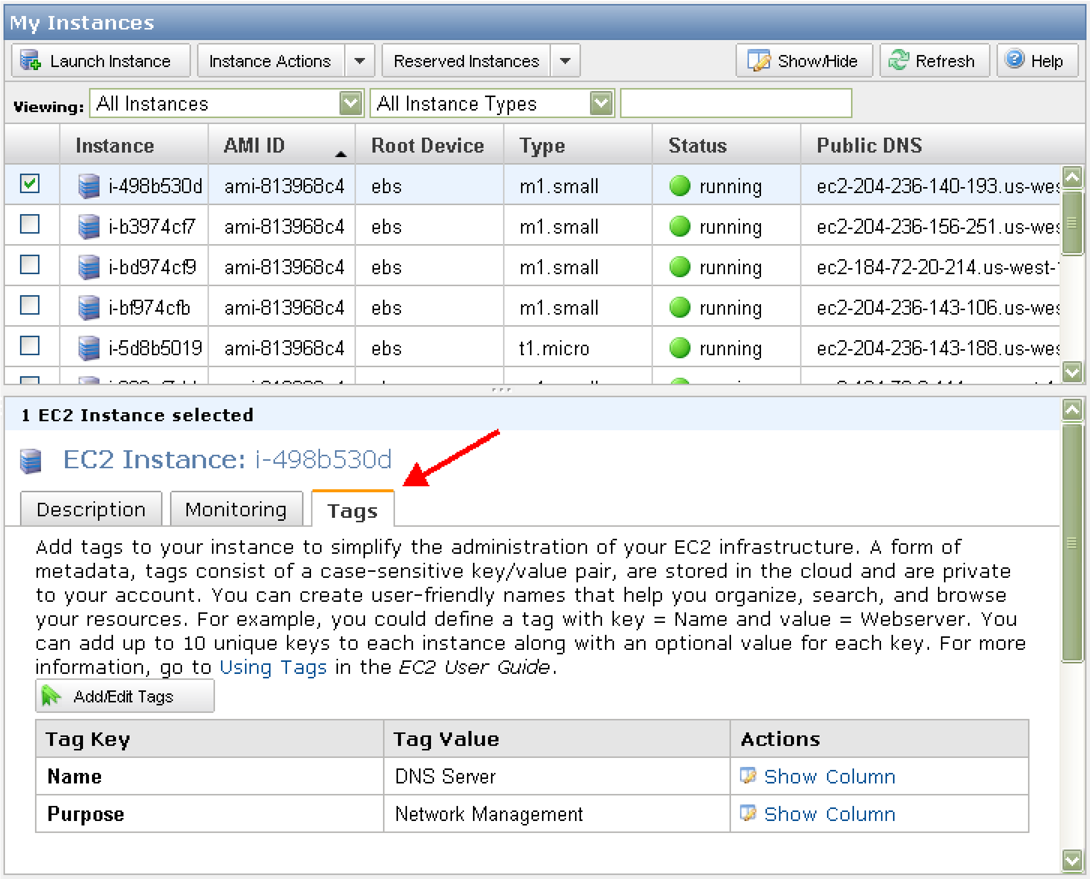
In this section, we'll show you how tags work in the AWS Management Console.

Note

Currently in the AWS Management Console, you can add tags only to instances, images, volumes, and snapshots.

The AWS Management Console is designed to let you add, edit, or remove tags only from a single resource at a time, whereas the EC2 API and command line tools let you do that with multiple resources in one call.

The AWS Management Console displays lists of your EC2 instances, images, and other resources. When you select a resource in one of these lists, a **Description** tab provides detailed information about the resource. Adjacent to that tab is a **Tags** tab where you can manage tags for the resource. The following image shows the tab for an instance with two tags (Name = DNS Server and Purpose = Network Management).



Viewing: All Instances | All Instance Types

	Instance	AMI ID	Root Device	Type	Status	Public DNS
☑	i-498b530d	ami-813968c4	ebs	m1.small	● running	ec2-204-236-140-193.us-wes
□	i-b3974cf7	ami-813968c4	ebs	m1.small	● running	ec2-204-236-156-251.us-wes
□	i-bd974cf9	ami-813968c4	ebs	m1.small	● running	ec2-184-72-20-214.us-west-1
□	i-bf974cfb	ami-813968c4	ebs	m1.small	● running	ec2-204-236-143-106.us-wes
□	i-5d8b5019	ami-813968c4	ebs	t1.micro	● running	ec2-204-236-143-188.us-wes

1 EC2 Instance selected

EC2 Instance: i-498b530d

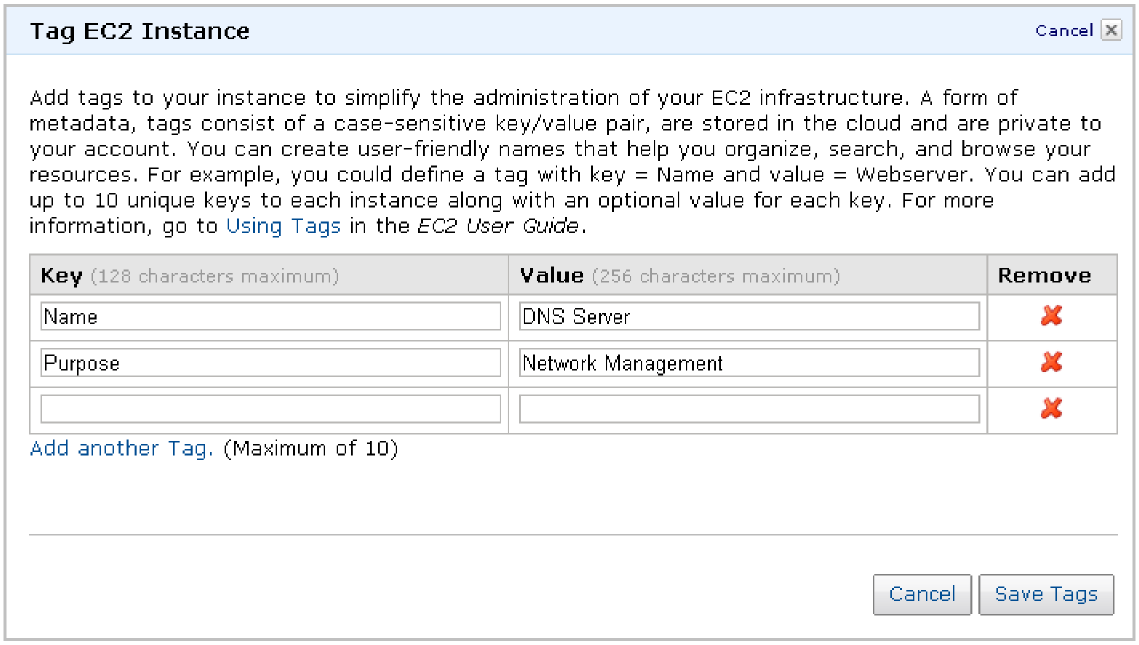
Add tags to your instance to simplify the administration of your EC2 infrastructure. A form of metadata, tags consist of a case-sensitive key/value pair, are stored in the cloud and are private to your account. You can create user-friendly names that help you organize, search, and browse your resources. For example, you could define a tag with key = Name and value = Webserver. You can add up to 10 unique keys to each instance along with an optional value for each key. For more information, go to Using Tags in the *EC2 User Guide*.

Tag Key	Tag Value	Actions
Name	DNS Server	Show Column
Purpose	Network Management	Show Column

Adding or Deleting Tags

To add a tag, click **Add/Edit Tags**, which opens a separate dialog box like the following. The first tag on the page automatically uses *Name* as the key. However, you can change it; using *Name* is only a suggestion.

To delete a tag, click the X for the tag in the **Remove** column.



Tag EC2 Instance

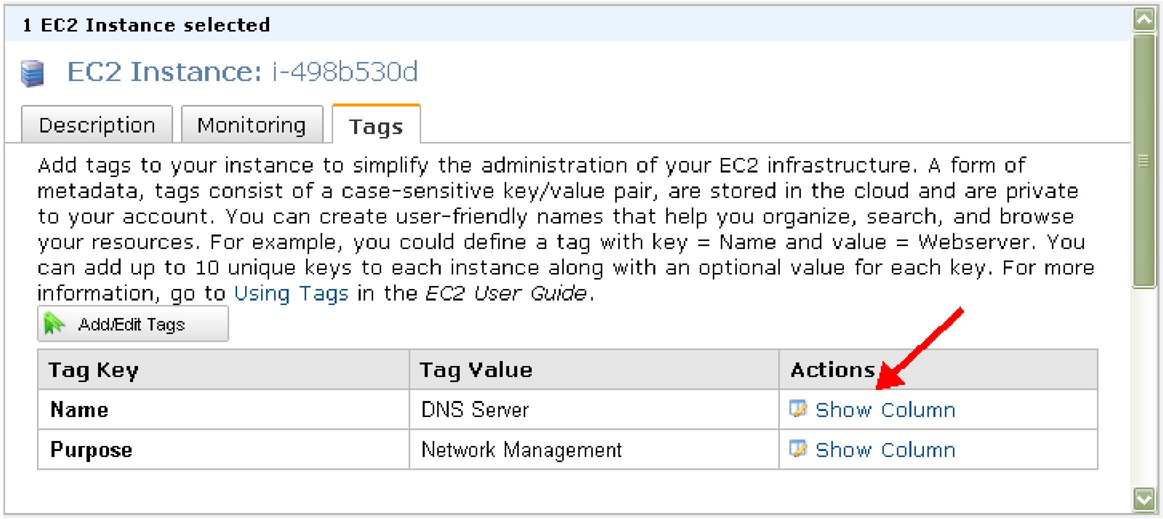
Add tags to your instance to simplify the administration of your EC2 infrastructure. A form of metadata, tags consist of a case-sensitive key/value pair, are stored in the cloud and are private to your account. You can create user-friendly names that help you organize, search, and browse your resources. For example, you could define a tag with key = Name and value = Webserver. You can add up to 10 unique keys to each instance along with an optional value for each key. For more information, go to Using Tags in the *EC2 User Guide*.

Key (128 characters maximum)	**Value** (256 characters maximum)	**Remove**
Name	DNS Server	✗
Purpose	Network Management	✗
		✗

Add another Tag. (Maximum of 10)

Displaying Tags as a Column in the List

Once you've added tags, you'll probably want them to appear in the resource list, so that you can sort and filter the resource list by tag. On the **Tags** tab, just click **Show Column** for the tag.



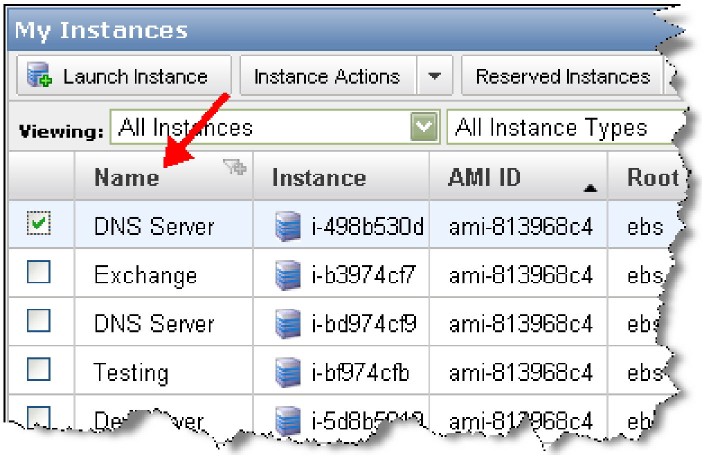
1 EC2 Instance selected

EC2 Instance: i-498b530d

Add tags to your instance to simplify the administration of your EC2 infrastructure. A form of metadata, tags consist of a case-sensitive key/value pair, are stored in the cloud and are private to your account. You can create user-friendly names that help you organize, search, and browse your resources. For example, you could define a tag with key = Name and value = Webserver. You can add up to 10 unique keys to each instance along with an optional value for each key. For more information, go to Using Tags in the *EC2 User Guide*.

Tag Key	Tag Value	Actions
Name	DNS Server	Show Column
Purpose	Network Management	Show Column

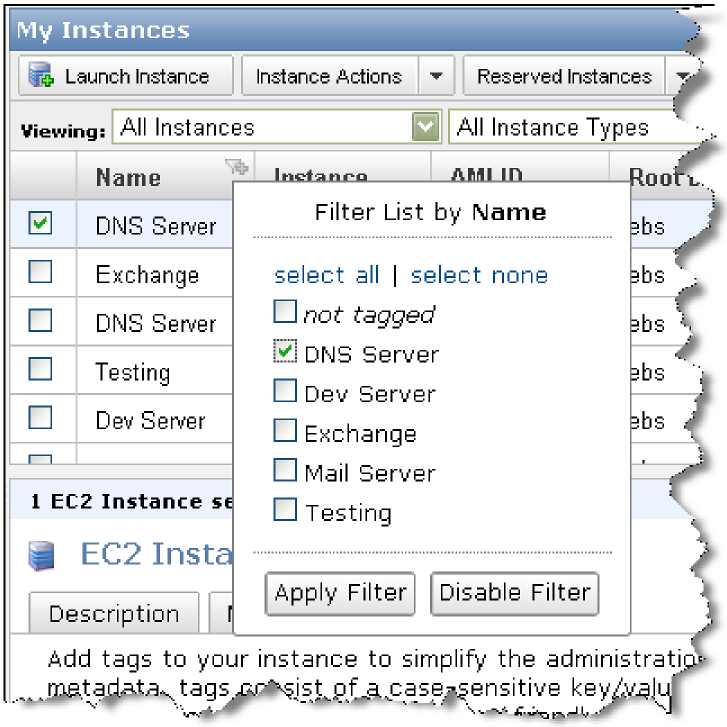
The tag appears in the resource list as a new **Name** column.



	Name	Instance	AMI ID	Root
☑	DNS Server	i-498b530d	ami-813968c4	ebs
□	Exchange	i-b3974cf7	ami-813968c4	ebs
□	DNS Server	i-bd974cf9	ami-813968c4	ebs
□	Testing	i-bf974cfb	ami-813968c4	ebs

Filtering the List by Tag

You can filter your list of resources by a particular tag's value. Just click the filter icon in the top right corner of the tag's column. The following image shows filtering by *Name = DNS Server*. You can filter the list based on multiple tag keys and multiple tag values.



	Name
☑	DNS Server
□	Exchange
□	DNS Server
□	Testing
□	Dev Server

Filter List by Name

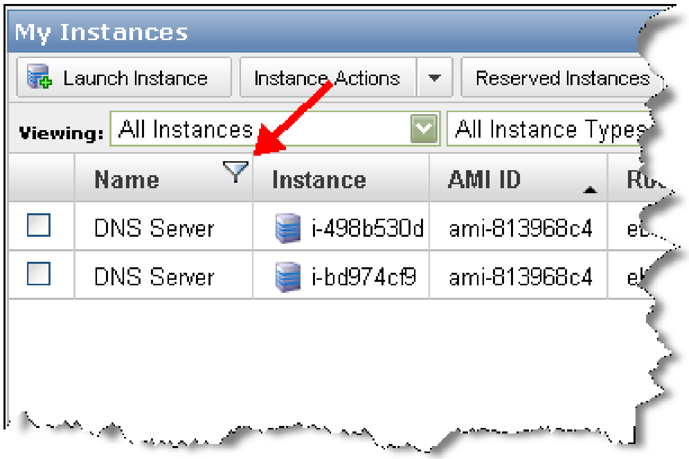
select all | select none

- □ *not tagged*
- ☑ DNS Server
- □ Dev Server
- □ Exchange
- □ Mail Server
- □ Testing

Apply Filter | Disable Filter

Add tags to your instance to simplify the administratio
metadata, tags consist of a case-sensitive key/valu

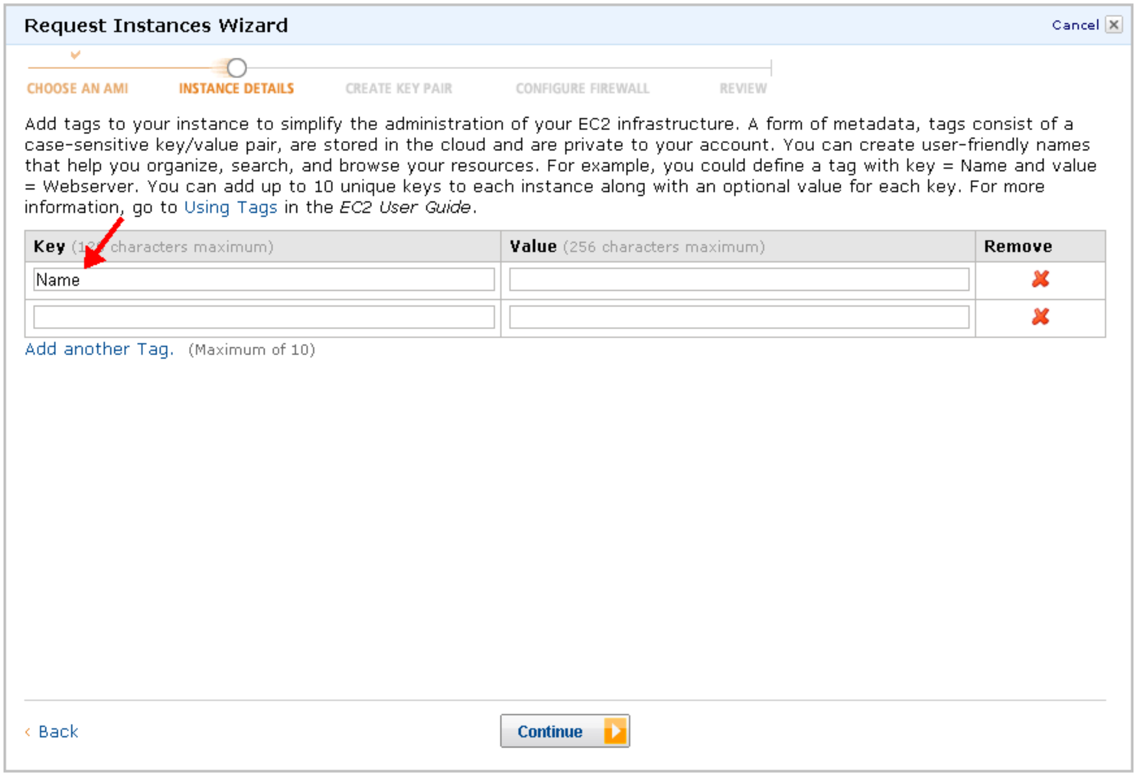
When you apply the filter, the filter icon changes to indicate the filtering.



Viewing: All Instances | All Instance Types

	Name	Instance	AMI ID	Ro
□	DNS Server	i-498b530d	ami-813968c4	eb
□	DNS Server	i-bd974cf9	ami-813968c4	eb

So far, you've seen how to add tags to an existing resource. The console enables you to add tags when you launch instances. There's a page in the launch wizard where you specify tags. As mentioned previously, the first tag on the page automatically uses *Name* as the key. However, you can change it; using *Name* is only a suggestion.



Add tags to your instance to simplify the administration of your EC2 infrastructure. A form of metadata, tags consist of a case-sensitive key/value pair, are stored in the cloud and are private to your account. You can create user-friendly names that help you organize, search, and browse your resources. For example, you could define a tag with key = Name and value = Webserver. You can add up to 10 unique keys to each instance along with an optional value for each key. For more information, go to Using Tags in the *EC2 User Guide*.

Key (128 characters maximum)	Value (256 characters maximum)	Remove
Name		✘
		✘

Add another Tag. (Maximum of 10)

Command Line Tools

For information about getting the command line tools, see Getting Started with the Command Line Tools (p. 384).

Summary of available commands:

- **ec2-create-tags—**Adds a set of tags to a set of resources. You also use this to update a tag's value (overwrites the existing value).
- **ec2-delete-tags—**Deletes a set of tags from a set of resources.
- **ec2-describe-tags—**Lists your tags, or just certain ones you specify.

For details about each command, go to the Amazon Elastic Compute Cloud Command Line Reference.

Adding Tags

Let's say you want to add the same two tags to an AMI and an instance. One tag is `webserver` (with no value), and the other is `stack=Production`. In this example, the value of the `webserver` tag is an empty string.

```
PROMPT>  ec2-create-tags ami-1a2b3c4d i-6f5d4e3a --tag webserver --tag
stack=Production
TAG  ami-1a2b3c4d  image  webserver
TAG  ami-1a2b3c4d  image  stack  Production
TAG  i-6f5d4e3a  image  webserver
TAG  i-6f5d4e3a  image  stack  Production
```

Listing Tags

You can list your tags and filter the results different ways, as shown in the following examples.

This example describes all the tags belonging to your account.

```
PROMPT> ec2-describe-tags
TAG  ami-1a2b3c4d  image  webserver
TAG  ami-1a2b3c4d  image  stack  Production
TAG  i-5f4e3d2a  instance  webserver
TAG  i-5f4e3d2a  instance  stack  Production
TAG  i-12345678  instance  database_server
TAG  i-12345678  instance  stack  Test
```

This example describes the tags for the AMI with ID ami-1a2b3c4d.

```
PROMPT> ec2-describe-tags --filter "resource-id=ami-1a2b3c4d"
TAG  ami-1a2b3c4d  image  webserver
TAG  ami-1a2b3c4d  image  stack  Production
```

This example describes the tags for all your instances.

```
PROMPT> ec2-describe-tags --filter "resource-type=instance"
TAG  i-5f4e3d2a  instance  webserver
TAG  i-5f4e3d2a  instance  stack  Production
TAG  i-12345678  instance  database_server
TAG  i-12345678  instance  stack  Test
```

This example describes the tags for all your instances tagged with the name *webserver*.

```
PROMPT> ec2-describe-tags --filter "resource-type=instance" --filter "key=web
server"
TAG  i-5f4e3d2a  instance  webserver
```

This example describes the tags for all your instances tagged with either stack=Test or stack=Production.

```
PROMPT> ec2-describe-tags --filter "resource-type=instance" --filter "key=stack"
 --filter "value=Test" --filter "value=Production"
TAG  i-5f4e3d2a  instance  stack  Production
TAG  i-12345678  instance  stack  Test
```

This example describes the tags for all your instances tagged with Purpose=[empty string].

```
PROMPT> ec2-describe-tags --filter "resource-type=instance" --filter "key=Pur
pose" --filter "value="
```

Changing Tag Values

Let's say you now want to change the value of the `stack` tag for one of the AMIs from `Production` to `Test`. You use the `ec2-create-tags` command to overwrite the original tag value.

```
PROMPT>  ec2-create-tags ami-1a2b3c4d  --tag stack=Test
TAG  ami-1a2b3c4d  image  stack  Test
```

Deleting Tags

Now let's say you want to delete the tags you originally assigned to the AMI and instance. You don't need to specify the value.

```
PROMPT>  ec2-delete-tags ami-1a2b3c4d i-6f5d4e3a --tag webserver --tag stack
```

If you specify a value for the key, the tag is deleted only if the tag's value matches the one you specified. If you specify the empty string as the value, the tag is deleted only if the tag's value is the empty string. The following example specifies the empty string as the value for the tag to delete (notice the equals sign after `Owner`).

```
PROMPT>  ec2-delete-tags snap-1a2b3c4d --tag Owner=
```

Filtering the List of Resources by Tags

You can describe your resources and filter the results based on the tag. Let's say that you've tagged all your Amazon EBS volumes with an `Owner` tag and a `Purpose` tag. You've got a series of teams (TeamA, TeamB, TeamC, etc.), and each has a series of volumes they own. You can get tag descriptions and filter the results by volume.

```
PROMPT>  ec2-describe-tags --filter resource-type=volume
TAG  vol-abcd1234  volume  Owner  TeamA
TAG  vol-abcd1234  volume  Purpose  Project1
TAG  vol-efba9876  volume  Owner  TeamA
TAG  vol-efba9876  volume  Purpose Project2
TAG  vol-4562dabf  volume  Owner  TeamA
TAG  vol-4562dabf  volume  Purpose  RawLogData
TAG  vol-2a3d4b5f  volume  Owner  TeamB
TAG  vol-2a3d4b5f  volume  Purpose Project1
TAG  vol-9f8g7d6c  volume  Owner  TeamB
TAG  vol-9f8g7d6c  volume  Purpose  Project2
TAG  vol-3b3a4c4d  volume  Owner  TeamB
TAG  vol-3b3a4c4d  volume  Purpose Logs
TAG  vol-1234abcd  volume  Owner  TeamC
TAG  vol-1234abcd  volume  Purpose Project1
TAG  vol-7f7g7d7a  volume  Owner  TeamC
TAG  vol-7f7g7d7a  volume  Purpose  Project2
TAG  vol-4a4b4c4d  volume  Owner  TeamC
TAG  vol-4a4b4c4d  volume  Purpose Logs
```

Likewise, you can get volume descriptions, and filter the results by tag. The filter name you use is `tag:`*key*. Let's say you want to get a list of just the volumes belonging to either TeamA or TeamB, and that contain log data. This time you use `ec2-describe-volumes` with a filter based on the tags of interest. You use a wildcard to find the volumes with "Log" in the *Purpose* tag's value.

```
PROMPT>  ec2-describe-volumes --filter tag:Owner=TeamA --filter tag:Owner=TeamB
 --filter tag:Purpose=*Log*
VOLUME  vol-4562dabf  5  us-east-1b  available   2010-02-22T22:50:43+0000
Owner  TeamA  Purpose  RawLogData
VOLUME  vol-3b3a4c4d  12  us-east-1b  available  2010-05-01T13:09:27+0000
Owner  TeamB  Purpose Logs
```

API

Summary of available API actions:

- **CreateTags—**Adds a set of tags to a set of resources. You also use this to update a tag's value (overwrites the existing value).
- **DeleteTags—**Deletes a set of tags from a set of resources.
- **DescribeTags—**Lists your tags, or just certain ones you specify.

For details about each action and examples, go to the Amazon Elastic Compute Cloud API Reference.

Using the API to add, update, list, and delete tags is straightforward. The topics in the API reference show examples. In the following sections, we will demonstrate how you can list tags, or list resources and filter the results based on tags.

Filtering the List of Resources by Tags

You can describe your resources and filter the results based on the tag. Let's say that you've tagged all your Amazon EBS volumes with an `Owner` tag and a `Purpose` tag. You've got a series of teams (TeamA, TeamB, TeamC, etc.), and each has a series of volumes they own. You can get tag descriptions and filter the results by volume.

Example Query request:

```
https://ec2.amazonaws.com/?Action=DescribeTags
&Filter.1.Name=resource-type
```

```
&Filter.1.Value=volume
&AuthParams
```

In this example, the response includes 18 tags covering 9 volumes.

```
<DescribeTagsResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>7a62c49f-347e-4fc4-9331-6e8eEXAMPLE</requestId>
   <tagSet>
       <item>
         <resourceId>vol-abcd1234</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamA</value>
      </item>
      <item>
         <resourceId>vol-abcd1234</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project1</value>
      </item>
      <item>
         <resourceId>vol-efba9876</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamA</value>
      </item>
      <item>
         <resourceId>vol-efba9876</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project2</value>
      </item>
      <item>
         <resourceId>vol-4562dabf</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamA</value>
      </item>
      <item>
         <resourceId>vol-4562dabf</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>RawLogData</value>
      </item>
      <item>
         <resourceId>vol-2a3d4b5f</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamB</value>
      </item>
      <item>
         <resourceId>vol-2a3d4b5f</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project1</value>
      </item>
      <item>
```

```
         <resourceId>vol-9f8g7d6c</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamB</value>
      </item>
      <item>
         <resourceId>vol-9f8g7d6c</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project2</value>
      </item>
      <item>
         <resourceId>vol-3b3a4c4d</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamB</value>
      </item>
      <item>
         <resourceId>vol-3b3a4c4d</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Logs</value>
      </item>
        <item>
         <resourceId>vol-1234abcd</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamC</value>
      </item>
      <item>
         <resourceId>vol-1234abcd</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project1</value>
      </item>
      <item>
         <resourceId>vol-7f7g7d7a</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamC</value>
      </item>
      <item>
         <resourceId>vol-7f7g7d7a</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Project2</value>
      </item>
      <item>
         <resourceId>vol-4a4b4c4d</resourceId>
         <resourceType>volume</resourceType>
         <key>Owner</key>
         <value>TeamC</value>
      </item>
      <item>
         <resourceId>vol-4a4b4c4d</resourceId>
         <resourceType>volume</resourceType>
         <key>Purpose</key>
         <value>Logs</value>
      </item>
  </tagSet>
</DescribeTagsResponse>
```

Likewise, you can get volume descriptions, and filter the results by tag. The filter name you use is `tag:`*`key`*. Let's say you want to get a list of just the volumes belonging to either TeamA or TeamB, and that contain log data. This time you use `DescribeVolumes` with a filter based on the tags of interest. You use a wildcard to find the volumes with "Log" in the `Purpose` tag's value.

Example Query request:

```
https://ec2.amazonaws.com/?Action=DescribeVolumes
&Filter.1.Name=tag:Owner
&Filter.1.Value.1=TeamA
&Filter.1.Value.2=TeamB
&Filter.2.Name=tag:Purpose
&Filter.2.Value.1=*Log*
&AuthParams
```

Because of the filtering, the response includes only two of the volumes that were in the preceding list of tagged volumes. The volume's tags are included in the response.

```
<DescribeVolumesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>7a62c49f-347e-4fc4-9331-6e8eEXAMPLE</requestId>
   <volumeSet>
      <item>
         <volumeId>vol-4562dabf</volumeId>
         <size>5</size>
         <snapshotId/>
         <availabilityZone>us-east-1b</availabilityZone>
         <status>available</status>
         <createTime>2010-02-22T22:50:43+0000</createTime>
         <attachmentSet/>
         <tagSet>
            <item>
               <key>Owner</key>
               <value>TeamA</key>
            </item>
            <item>
               <key>Purpose</key>
               <value>RawLogData</key>
            </item>
         </tagSet>
      </item>
      <item>
         <volumeId>vol-3b3a4c4d</volumeId>
         <size>12</size>
         <snapshotId/>
         <availabilityZone>us-east-1b</availabilityZone>
         <status>available</status>
         <createTime>2010-05-01T13:09:27+0000</createTime>
         <attachmentSet/>
         <tagSet>
            <item>
               <key>Owner</key>
               <value>TeamB</key>
            </item>
             <item>
               <key>Purpose</key>
               <value>Logs</key>
            </item>
         </tagSet>
      </item>
    </volumeSet>
</DescribeVolumesResponse>
```

Using Instance IP Addresses

Topics

This section describes the types of IP addresses available to Amazon EC2 instances, including Elastic IP addresses, which you can remap on demand.

Public and Private Addresses

All Amazon EC2 instances are assigned two IP addresses at launch: a private address (RFC 1918) and a public address that are directly mapped to each other through Network Address Translation (NAT). Private addresses are only reachable from within the Amazon EC2 network. Public addresses are reachable from the Internet.

Amazon EC2 also provides an internal DNS name and a public DNS name that map to the private and public IP addresses respectively. The internal DNS name can only be resolved within Amazon EC2. The public DNS name resolves to the public IP address outside the Amazon EC2 network and the private IP address within the Amazon EC2 network.

Note

If you require persistent Internet routable IP addresses that can be assigned to and removed from instances as necessary, use Elastic IP addresses. For more information, see Elastic IP Addresses Concepts (p. 293).

Private (RFC 1918) Addresses and Internal DNS

All Amazon EC2 instances are allocated a private address by DHCP. These ranges are defined in RFC 1918, are only routable within Amazon EC2, and are used for communication between instances. For more information, go to RFC 1918.

This private address is associated exclusively with the instance for its lifetime and is only returned to Amazon EC2 when the instance is stopped or terminated.

Always use the internal address when you are communicating between Amazon EC2 instances. This ensures that your network traffic follows the highest bandwidth, lowest cost, and lowest latency path through our network.

Each instance is provided an internal DNS name that resolves to the private IP address of the instance from within Amazon EC2; it will not resolve outside of Amazon EC2.

Public Addresses and DNS

At launch, a public address is also associated with each Amazon EC2 instance using Network Address Translation (NAT). For more information about NAT, go to RFC 1631: The IP Network Address Translator (NAT).

This public address is associated exclusively with the instance until it is stopped, terminated or replaced with an Elastic IP address.

Important

Amazon EC2 instances that access other instances through their public NAT IP address are charged for Regional or Internet data transfer, depending on whether the instances are in the same Region.

Each instance is provided an external DNS name that resolves to the public IP address of the instance outside the Amazon EC2 network and the private IP address from within Amazon EC2 network.

IP Addresses for Instances in a VPC

If you're using Amazon Virtual Private Cloud, when you create a subnet in your VPC, you specify the range of private IP addresses in your network to use for the subnet. Each time you launch an instance in the subnet, you can optionally specify an IP address for the instance in the subnet's range. If you don't specify an address, we automatically assign an IP address from the subnet's range. The assigned address stays with the instance until it terminates. Even if you stop and start the instance, it retains the same IP address.

For more information about Amazon VPC, go to the Amazon Virtual Private Cloud Getting Started Guide.

Determining Your IP Addresses

This section describes how to determine the internal and external IP addresses of an instance you own.

AWS Management Console

To determine your instance's public and private IP addresses

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Instances** in the **Navigation** pane.
   The console displays a list of running instances.
3. Locate and select an instance.
   The console displays information about the instance in the lower pane.
4. To determine the public IP address, use the IP address specified within the **Public DNS** field.
5. To determine the private IP address, look at the **Private IP Address** field.

Command Line Tools

To determine your instance's public and private IP addresses

- Use the **ec2-describe-instances** command with the instance ID. The instance must be running.

```
PROMPT>  ec2-describe-instances instance_id
RESERVATION   r-f25e6f9a 999988887777  default
INSTANCE <instance_id> ami-b232d0db ec2-204-236-202-134.compute-1.amazon
aws.com domU-12-31-39-00-86-35.compute-1.internal running gsg-keypair 0
m1.small 2010-03-30T08:43:48+0000 us-east-1a aki-94c527fd ari-96c527ff mon
itoring-disabled 204.236.202.134 10.254.137.191 ebs
BLOCKDEVICE     /dev/sda1       vol-cf13b3a6    2010-03-30T08:01:44.000Z
```

  In the preceding example, the public IP address is 204.236.202.134. The private IP address is 10.254.137.191.

API

To determine your instance's public and private IP addresses

1. Construct the following Query request.

   ```
   https://ec2.amazonaws.com/
   ?Action=DescribeInstances
   &InstanceId.1=instance-id
   &...auth parameters...
   ```

   Following is an example response.

   ```
   <DescribeInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

     <reservationSet>
       <item>
         <reservationId>r-44a5402d</reservationId>
         <ownerId>999988887777</ownerId>
         <groupSet>
           <item>
             <groupId>default</groupId>
           </item>
         </groupSet>
         <instancesSet>
           <item>
             <instanceId>i-28a64341</instanceId>
             <imageId>ami-6ea54007</imageId>
             <instanceState>
                <code>16</code>
                <name>running</name>
             </instanceState>
              <privateDnsName>ip-10-203-46-149.ec2.internal</privateDnsName>
              <dnsName>ec2-75-101-217-237.compute-1.amazonaws.com</dnsName>
              ...
              <privateIpAddress>10.203.46.149</privateIpAddress>
              <ipAddress>75.101.217.237</ipAddress>
               ...
           </item>
         </instancesSet>
       </item>
     </reservationSet>
   </DescribeInstancesResponse>
   ```

2. To determine the public IP address, use the IP address specified within `ipAddress`.
3. To determine the private IP address, use the IP address specified within `privateIpAddress`.

From Within the Instance

To determine your private IP address in Linux/UNIX

1. Connect to the instance.
2. Enter one of the following commands:

   - `#  ifconfig eth0`

-
  ```
  # curl http://169.254.169.254/latest/meta-data/local-ipv4
  ```

  The second option refers to the instance metadata. For more information, see Using Instance Metadata (p. 96).

To determine your private IP address in Windows

1. Connect to the instance.
2. On the taskbar, click **Start**, right-click **My Computer**, and select **Properties**.
3. Click the **Computer Name** tab. The IP address appears in the **Full computer name** field.

To determine your public IP address

1. Connect to the instance.
2. Determine your public IP address from your instance by referring to the instance data.

   ```
   PROMPT>  GET http://169.254.169.254/latest/meta-data/public-ipv4
   ```

Related Topics

- IP Information FAQ (p. 410)

Using Elastic IP Addresses

Topics

Elastic IP addresses are static IP addresses designed for dynamic cloud computing. An elastic IP address is associated with your account, not a particular instance. You control addresses associated with your account until you choose to explicitly release them.

Important

If you're using Amazon Virtual Private Cloud: There's a separate pool of Elastic IP addresses to use with Amazon VPC. Your EC2 Elastic IP addresses won't work with instances in a VPC, and your VPC Elastic IP addresses won't work with instances in EC2.. The following sections describe how to work with EC2 addresses, which have different characteristics than VPC Elastic IP addresses. For information about using VPC Elastic IP addresses, go to Elastic IP Addresses in the *Amazon Virtual Private Cloud User Guide*.

Elastic IP Addresses Concepts

By default, all Amazon EC2 instances are assigned two IP addresses at launch: a private (RFC 1918) address and a public address that is mapped to the private IP address through Network Address Translation (NAT).

If you use dynamic DNS to map an existing DNS name to a new instance's public IP address, it might take up to 24 hours for the IP address to propagate through the Internet. As a result, new instances might not receive traffic while terminated instances continue to receive requests.

To solve this problem, Amazon EC2 provides Elastic IP addresses. Elastic IP addresses are static IP addresses designed for dynamic cloud computing. Elastic IP addresses are associated with your account, not specific instances. Any Elastic IP addresses that you associate with your account remain associated with your account until you explicitly release them. Unlike traditional static IP addresses, however, Elastic IP addresses allow you to mask instance or Availability Zone failures by rapidly remapping your public IP addresses to any instance in your account.

You can associate one Elastic IP address with only one instance at a time.

When you associate an Elastic IP address with an instance, its current public IP address is released to the Amazon EC2 public IP address pool. If you disassociate an Elastic IP address from the instance, the instance is automatically assigned a new public IP address within a few minutes. In addition, stopping the instance also disassociates the Elastic IP address from it.

To ensure our customers are efficiently using Elastic IP addresses, we impose a small hourly charge when these IP addresses are not mapped to an instance. When these IP addresses are mapped to an instance, they are free of charge.

For an example of using Elastic IP addresses, see Example (p. 301).

API and Command Overview

The following table summarizes the available Elastic IP address commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-allocate-address` `AllocateAddress`	Acquires an Elastic IP address for use with your account.
`ec2-associate-address` `AssociateAddress`	Associates an Elastic IP address with an instance.
`ec2-describe-addresses` `DescribeAddresses`	Lists Elastic IP addresses assigned to your account.
`ec2-disassociate-address` `DisassociateAddress`	Disassociates an Elastic IP address from the instance it's associated with.
`ec2-release-address` `ReleaseAddress`	Releases an Elastic IP address associated with your account. After releasing an Elastic IP address, it is released to the IP address pool and might no longer be available to your account.

Allocating Elastic IP Addresses

This section describes how to assign an Amazon EC2 Elastic IP address to your account.

AWS Management Console

To allocate a new IP address for use with your account

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Elastic IPs** in the **Navigation** pane.
   The console displays a list of Elastic IP addresses assigned to your account.
3. Click **Allocate New Address**.
   A confirmation dialog box appears.
4. Ensure the value for **EIP used in** is set to `EC2`, and click **Yes, Allocate**.
   A new standard (EC2) Elastic IP address appears in the list.

Note

The Elastic IP address is associated with your account and billed accordingly until you release it.

Command Line Tools

To allocate a new IP address for use with your account

- Enter the following command:

```
PROMPT> ec2-allocate-address
```

Amazon EC2 returns an Elastic IP address similar to the following:

```
ADDRESS 192.0.2.1   standard
```

The value `standard` means this address is for use only with standard (EC2) instances, and not Amazon VPC instances.

Note

The Elastic IP address is associated with your account and billed accordingly until you release the address with `ec2-release-address` command.

API

To allocate a new IP address for use with your account

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=AllocateAddress
&...auth parameters...
```

Following is an example response.

```
<AllocateAddressResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <publicIp>192.0.2.1</publicIp>
   <domain>standard</domain>
</AllocateAddressResponse>
```

The domain value `standard` means this address is for use only with standard (EC2) instances, and not Amazon VPC instances.

Note

The Elastic IP address is associated with your account and billed accordingly until you release the address with `ReleaseAddress`.

Describing Elastic IP Addresses

This section describes how to view the Elastic IP addresses allocated to your account.

AWS Management Console

To view Elastic IP addresses assigned to your account

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Elastic IPs** in the **Navigation** pane.
   The console displays a list of Elastic IP addresses assigned to your account.
3. To reduce the size of the list, start typing part of the IP address or instance ID to which it is assigned in the search box.

Command Line Tools

To view Elastic IP addresses assigned to your account

1. To view all Elastic IP addresses assigned to your account:

   ```
   PROMPT> ec2-describe-addresses
   ```

   Amazon EC2 returns a list of Elastic IP addresses similar to the following:

   ```
   ADDRESS 192.0.2.1   standard
   ADDRESS 198.51.100.1   standard
   ```

   If the IP address is mapped, it is followed by the instance ID it's mapped to.

   ```
   ADDRESS  203.0.113.1  i-996fc0f2   standard
   ```

2. To verify a specific Elastic IP address:

   ```
   PROMPT> ec2-describe-addresses ip_address
   ```

Amazon EC2 returns the specified Elastic IP address, similar to the following:

```
ADDRESS 192.0.2.1   standard
```

API

To view Elastic IP addresses assigned to your account

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeAddresses
&...auth parameters...
```

Following is an example response.

```
<DescribeAddressesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

   <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
   <addressesSet>
     <item>
       <publicIp>192.0.2.1</publicIp>
       <domain>standard</domain>
       <instanceId/>
     </item>
   </addressesSet>
</DescribeAddressesResponse>
```

Associating an Elastic IP Address with a Running Instance

After you allocate an Elastic IP address, you can map it to a running instance.

AWS Management Console

To associate an Elastic IP address with an instance

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Instances** in the **Navigation** pane.
   The console displays a list of instances.
3. Write down the instance ID to associate with the Elastic IP address.
4. Click **Elastic IPs** in the **Navigation** pane.
   The console displays a list of Elastic IP addresses assigned to your account.
5. Select an address and click **Associate Address**.
   The **Associate Address** dialog box appears.
6. Select the instance from the **Instance** list box and click **Yes, Associate**.
   The Elastic IP address is associated with the instance.

Command Line Tools

To associate an Elastic IP address with an instance

1. Describe running instances:

   ```
   PROMPT> ec2-describe-instances
   ```

   Amazon EC2 returns output similar to the following:

   ```
   RESERVATION      r-ae33c2c7      999988887777    default
   INSTANCE i-afb130c4 ami-3c47a355 ec2-184-73-1-155.compute-1.amazonaws.com
   domU-12-31-39-09-1D-E5.compute-1.internal running gsg-keypair 0 m1.small
   2010-03-30T07:42:51+0000 us-east-1a aki-a71cf9ce ari-a51cf9cc monitoring-
   disabled   184.73.1.155   10.210.34.19  instance-store   paravirtual   xen
   RESERVATION  r-8a3374e2 999988887777   default
   INSTANCE i-85b435ee ami-b232d0db ec2-204-236-202-134.compute-1.amazonaws.com
    domU-12-31-39-00-86-35.compute-1.internal running gsg-keypair 0 m1.small
   2010-03-30T08:43:48+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
   disabled   204.236.202.134 10.254.137.191  ebs   paravirtual   xen
   ```

   Write down the instance ID to associate with an Elastic IP address.

2. Describe Elastic IP addresses assigned to the account:

   ```
   PROMPT> ec2-describe-addresses
   ```

   Amazon EC2 returns a list of Elastic IP addresses similar to the following:

   ```
   ADDRESS 75.101.157.145   standard
   ADDRESS 198.51.100.1   standard
   ```

   Write down the Elastic IP address to associate with an instance.

3. To associate the instance and Elastic IP address:

   ```
   PROMPT> ec2-associate-address -i instance_id ip_address
   ```

   Amazon EC2 returns output similar to the following:

   ```
   ADDRESS 75.101.157.145   i-afb130c4
   ```

4. Associations take a few minutes to complete. To verify the association using `ec2-describe-addresses`:

   ```
   PROMPT> ec2-describe-addresses
   ```

   Amazon EC2 returns output similar to the following:

   ```
   ADDRESS 75.101.157.145   i-afb130c4   standard
   ```

5. To verify the association using `ec2-describe-instances`:

```
PROMPT> ec2-describe-instances instance_id
```

Amazon EC2 returns output similar to the following:

```
RESERVATION r-9284a1fa 999988887777  default
INSTANCE i-afb130c4 ami-3c47a355 ec2-75-101-157-145.compute-1.amazonaws.com
 domU-12-31-39-09-25-62.compute-1.internal running 5fmorgin-ami    0
         m1.small   2010-03-17T13:17:41+0000        us-east-1a      aki-
a71cf9ce    ari-a51cf9cc      monitoring-disabled      75.101.157.145
10.210.42.144                   instance-store   paravirtual   xen
```

API

To associate an Elastic IP address with an instance

1. Describe running instances and write down the instance ID of the instance that you will associate with the Elastic IP address. For more information, see API (p. 291).
2. Describe Elastic IP addresses assigned to the account and write down the Elastic IP address that you will associate with the instance. For more information, see API (p. 296).
3. Associate the instance and Elastic IP address using the following Query request.

```
https://ec2.amazonaws.com/
?Action=AssociateAddress
&InstanceId=instance-id
&PublicIp=elastic-ip-address
&...auth parameters...
```

Following is an example response.

```
<AssociateAddressResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

  <requestId>ce410ee1-068c-4026-a36e-86cefd453c28</requestId>
  <return>true</return>
</AssociateAddressResponse>
```

4. Associations take a few minutes to complete. You can verify the association using `DescribeAddresses` or `DescribeInstances`.

Associating an Elastic IP Address with a Different Running Instance

Once an Elastic IP Address is allocated, you can map it to a different running instance.

Note

It is highly unlikely that an instance will be configured with its original public IP address that it used prior to being mapped.

AWS Management Console

To remap an IP address

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Instances** in the **Navigation** pane.

   The console displays a list of running instances.
3. Write down the new instance ID to associate with the Elastic IP address.
4. Click **Elastic IPs** in the **Navigation** pane.

   The console displays a list of Elastic IP addresses assigned to your account.
5. Locate the IP address to remap and click **Disassociate**.

   A confirmation dialog box appears.
6. Click **Yes, Disassociate**.

   The Elastic IP address is disassociated from the instance and you are returned to the list of Elastic IP addresses assigned to your account.
7. Locate the IP address in the list and click **Associate**.

   The **Associate Address** dialog box appears.
8. Select the new instance from the **Instance ID** list box and click **Associate**.

   The Elastic IP address is associated with the new instance.

Command Line Tools

To remap an IP address

1. Describe running instances:

   ```
   PROMPT> ec2-describe-instances
   ```

   Amazon EC2 returns output similar to the following:

   ```
   RESERVATION r-9284a1fa 999988887777  default
   INSTANCE i-b2e019da ami-3c47a355 ec2-75-101-157-145.compute-1.amazonaws.com
    domU-12-31-39-09-25-62.compute-1.internal running gsg-keypair    0
           m1.small   2010-03-17T13:17:41+0000        us-east-1a      aki-
   a71cf9ce    ari-a51cf9cc      monitoring-disabled      75.101.157.145
   10.210.42.144                   instance-store   paravirtual   xen
   RESERVATION     r-f25e6f9a      999988887777    default
   INSTANCE        i-b2e019db ami-3c47a355    ec2-184-73-1-155.compute-
   1.amazonaws.com  domU-12-31-39-09-1D-E5.compute-1.internal        running
   gsg-keypair     0               m1.small   2010-03-30T07:42:51+0000
    us-east-1a      aki-a71cf9ce    ari-a51cf9cc      monitoring-disabled
     184.73.1.155    10.210.34.19                   instance-store   paravir
   tual   xen
   ```

   Write down the instance ID to associate with an Elastic IP address.
2. Associate the address with a new instance:

   ```
   PROMPT> ec2-associate-address -i instance_id ip_address
   ```

   Amazon EC2 returns output similar to the following:

```
ADDRESS 75.101.157.145   i-b2e019db   standard
```

3. Verify the changes:

   ```
   PROMPT> ec2-describe-instances
   ```

   Amazon EC2 returns output similar to the following:

   ```
   RESERVATION r-9284a1fa 999988887777  default
   INSTANCE i-b2e019da ami-3c47a355 ec2-184-73-1-123.compute-1.amazonaws.com
   domU-12-31-39-09-25-62.compute-1.internal running gsg-keypair 0  m1.small
     2010-03-17T13:17:41+0000       us-east-1a aki-a71cf9ce ari-a51cf9cc mon
   itoring-disabled  184.73.1.123 10.210.42.144  instance-store   paravirtual
      xen
   RESERVATION  r-f25e6f9a 999988887777  default
   INSTANCE i-b2e019db ami-3c47a355 ec2-75-101-157-145.compute-1.amazonaws.com
    domU-12-31-39-09-1D-E5.compute-1.internal running gsg-keypair 0  m1.small
    2010-03-30T07:42:51+0000 us-east-1a aki-a71cf9ce ari-a51cf9cc monitoring-
   disabled  75.101.157.145 10.210.34.19  instance-store   paravirtual   xen
   ```

API

To remap an IP address

1. Describe running instances and write down the instance ID of the instance that you will associate with the Elastic IP address.
2. Describe Elastic IP addresses assigned to the account and write down the Elastic IP address to remap.
3. Associate the instance and Elastic IP address using the following Query request.

   ```
   https://ec2.amazonaws.com/
   ?Action=AssociateAddress
   &InstanceId=instance-id
   &PublicIp=elastic-ip-address
   &...auth parameters...
   ```

   Following is an example response.

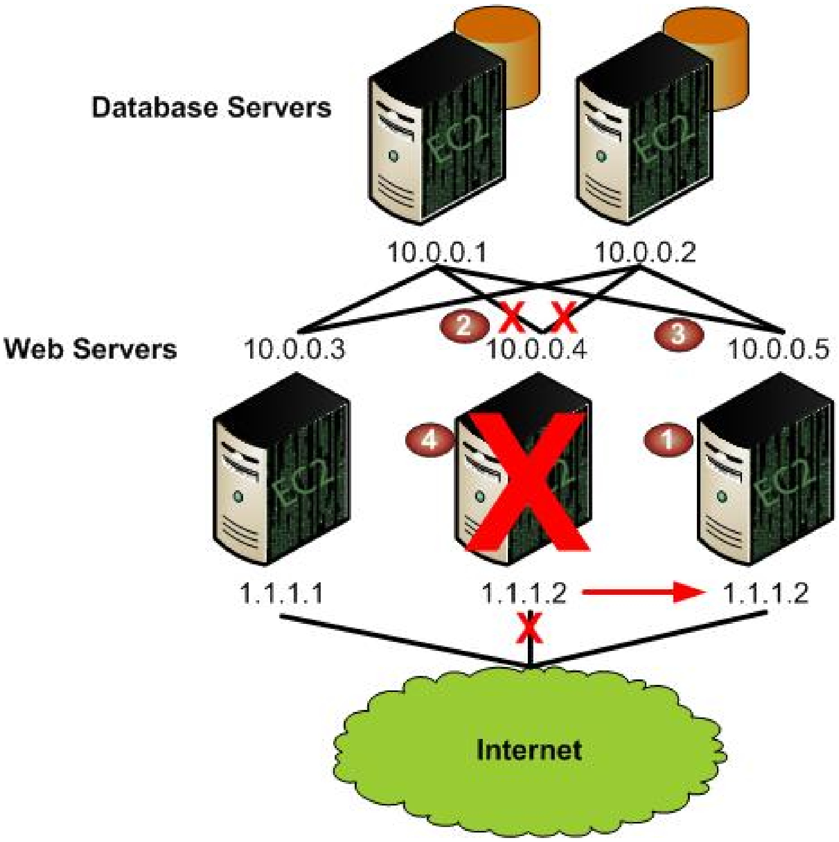
   ```
   <AssociateAddressResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">

     <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
     <return>true</return>
   </AssociateAddressResponse>
   ```

   Associations take a few minutes to complete.
4. Verify the association using `DescribeAddresses` or `DescribeInstances`.

Example

The following diagram and table shows an example of using Elastic IP addresses.



1	Web servers are connected to the Internet through Elastic IP addresses (1.1.1.1 and 1.1.1.2) and to database servers through their private IP addresses.
2	The administrator decides to replace a web server with a larger instance type. To do this, the administrator starts a new instance using a larger instance type, disassociates the 1.1.1.2 Elastic IP address from its running instance, and associates the address with the new instance.
3	The administrator terminates the old instance.

AWS Management Console

The following table describes how you set up the preceding tasks using the AWS Management Console.

Launch Process

1	Click **Instances** in the **Navigation** pane. The console displays a list of running instances.
2	Click **Launch Instance**, specify settings that meet the new requirements, and click **Launch**.
3	Click **Elastic IPs** in the **Navigation** pane. The console displays a list of Elastic IP addresses assigned to the account.

4	Select the desired IP address that is assigned to another instance, click **Disassociate**, and confirm the disassociation. You're returned to the list of Elastic IP addresses.
5	Locate the IP address in the list, click **Associate**, select the new instance, and confirm the association. The Elastic IP address is assigned to the new instance.
6	Click **Instances** in the **Navigation** pane, select the old instance, and clicks **Terminate**. Amazon EC2 begins shutting down the old instance.

Command Line Tools

The following example shows how to set up the preceding tasks using the command line tools (assuming the Elastic IP address is 67.202.55.255).

Note

The command to disassociate the address is optional because `ec2-associate-address` automatically disassociates the Elastic IP address if it is assigned to another instance.

```
PROMPT> ec2-run-instances ami-6ba54002 -n 1 --availability-zone us-east-1a
RESERVATION r-a034c7c9 924417782495 default
INSTANCE i-3ea74257 ami-6ba54002 pending 0 m1.large 2007-07-11T16:40:44+0000
us-east-1a

PROMPT> ec2-disassociate-address 67.202.55.255
ADDRESS 67.202.55.255

PROMPT> ec2-associate-address -i i-3ea74257 67.202.55.255
ADDRESS 67.202.55.255   i-43a4412a

PROMPT> ec2-terminate-instances i-4bc32334
INSTANCE i-4bc32334 running shutting-down
```

API

The following demonstrates how to set up these tasks using the API.

1. Run a new larger instance type using `RunInstances`.
2. Disassociate the Elastic IP address from the old instance using `DisassociateAddress`.

   Note

   This step is optional because `AssociateAddress` automatically disassociates the Elastic IP address if it is assigned to another instance.

3. Associate the Elastic IP address with the new instance using `AssociateAddress`.
4. Verify the association using `DescribeInstances`.
5. Terminate the old instance using `TerminateInstances`.

Elastic IP Address Limit

By default, all accounts are limited to 5 Elastic IP addresses because public (IPV4) Internet addresses are a scarce public resource. We strongly encourage you to use Elastic IPs primarily for load balancing use cases, and private IP addresses for all other inter-node communication.

If you feel your architecture warrants an exception, please complete the Amazon EC2 Elastic IP Address Request Form. We will ask you to think through your use case and help us understand your need for additional addresses.

Using Reverse DNS for EMail Applications

If you intend to send email to third parties from Amazon EC2 instances, we suggest you provision one or more elastic IP addresses and provide them to us in the Request to Remove Email Sending Limitations form. AWS works with ISPs and Internet anti-spam organizations (such as Spamhaus) to reduce the chance that your email sent from these addresses will be flagged as spam.

In addition, assigning a static reverse DNS record to your Elastic IP address used to send email can help avoid having email flagged as spam by some anti-spam organizations. You can provide us with a reverse DNS record (for example: foo.yourcompany.com) to associate with your addresses through the aforementioned form. Note that a corresponding forward DNS record (A Record) pointing to your Elastic IP address must exist before we can create your reverse DNS record.

Using Security Groups

Topics

This section describes Amazon EC2 security groups and how to use them.

Important

For information about Amazon VPC security groups, go to Security Groups in the *Amazon Virtual Private Cloud User Guide*.

Security Group Concepts

A security group acts as a firewall that controls the traffic allowed into a group of instances. When you launch an Amazon EC2 instance, you can assign it to one or more security groups. For each security group, you add rules that govern the allowed inbound traffic to instances in the group. All other inbound traffic is discarded. You can modify rules for a security group at any time. The new rules are automatically enforced for all existing and future instances in the group.

Note

You can create up to 500 EC2 security groups, with up to 100 rules per group.

Caution

Because an Amazon EC2 instance can belong to multiple security groups, more than 100 rules can apply to an instance. Associating hundreds of rules with an instance might cause problems when you access the instance. We recommend you condense your rules as much as possible.

Default Security Group

Your AWS account automatically comes with a *default security group* for your Amazon EC2 instances. If you don't specify another security group at instance launch time, the instance is automatically launched into this default group. These are the initial settings for this group:

- Allow no inbound traffic
- Allow all outbound traffic
- Allow the instances in the group to talk to each other

You can change the default group's inbound rules. For example, you might change the rules to allow SSH or Remote Desktop connections from specific hosts for the purposes of instance management. For another example, see Modifying the Default Group (p. 315).

The default security group is automatically named *default*, and it has an AWS-assigned ID. You can't delete the default security group.

Creating Your Own Security Groups

If you don't want all your instances to use the default security group, you can create your own (up to 500 total). You can create groups that reflect the different roles your EC2 instances play in your system (e.g., web server, database server). For an example, see Creating a Three-Tier Web Service (p. 317).

When you create a new security group, you must provide a friendly name and a description of the group. AWS assigns each group a unique ID (e.g., sg-1a2b3c4d). These are the initial settings for a new group that you create:

- Allow no inbound traffic
- Allow all outbound traffic

After you've created the security group, you can change its rules to reflect the type of inbound traffic you want to allow into the group. You can change only the inbound rules; EC2 security groups don't have modifiable outbound rules. If you want the instances in the group to talk to each other, you must explicitly add rules to allow that.

Instance Group Membership

When you launch an instance, you can assign it to as many groups as you like. When deciding whether to allow traffic to a given instance, we evaluate all the rules from all the groups the instance is in.

If you don't specify any groups at launch time, the instance is automatically assigned to the *default* group and uses the rules you've set up for that group.

After an instance is running, you can't change which EC2 security groups it belongs to. However, you can change the rules of an existing group, and those changes propagate to the instances in the group.

Security Group Rules

An EC2 security group's rules control the inbound traffic allowed in to the instances in the group. All outbound traffic is automatically allowed. You can't change the outbound behavior.

Each EC2 security group rule enables a specific source to access the instances in the group using a certain protocol (TCP, UDP, or ICMP) and destination port or ports (if the protocol is TCP or UDP). For example, a rule could allow IP address 203.0.113.1 (the source) to access the instances in the group on TCP port 22 (the protocol and destination port). If you specify ICMP as the protocol for the rule, you must also specify an ICMP type and code.

The source can be an individual IP address (203.0.113.1), a range of addresses (e.g., 203.0.113.0/24), or an EC2 security group. The security group can be another group in your AWS account, a group in another AWS account, or the security group itself.

By specifying a security group as the source, you allow incoming traffic from all instances that belong to the source security group. The incoming traffic that you allow is based on the private IP addresses of the instances in the source security group. You might specify another security group in your account if you're creating a three-tier web service (see Creating a Three-Tier Web Service (p. 317)).

If you specify the security group itself as the source, each instance in the group will accept inbound traffic from fellow group members. For example, the default security group specifies itself as a source security group in its inbound security group rules. This is why members of the default security group allow inbound traffic from other members of the default security group.

You can't modify an existing rule in a group. However, you can add and remove rules to a group at any time. Your changes propagate to existing instances in the group after a short period.

Each EC2 security group can have up to 100 rules.

Understanding CIDR Notation

To specify a range of IP addresses using CIDR notation, you specify a base IP address and a suffix that indicates the number of significant bits in the IP address used to identify the network. The larger the suffix, the fewer number of hosts will be in the CIDR block. For example, say you specify a source as *`10.10.1.32/27`*. This specifies `10.10.1.32` as the base IP address and `27` as the suffix. Because 27 of the 32 possible bits in the IP4 address are allocated to the network address, there are only 5 (32 minus 27) bits available to specify host addresses. You can only represent 32 distinct numbers with 5 bits (2 to the 5th power = 32), which means 32 hosts on that network match the criteria and are in the CIDR block.

To better understand how this works, let’s look at it in binary notation. The first 27 bits have to be an exact match in order for the range to match. It doesn’t matter what happens after the 27 bits. For this example, any IP address that has `00001010.00001010.00000001.001` in the first 27 bits will match the address.

Here's how it works:

```
10.10.1.32 =
00001010.00001010.00000001.00100000
First 27 bits==
00001010.00001010.00000001.001
10.10.1.44 matches 10.10.1.32/27 =
00001010.00001010.00000001.00101100
But 10.10.1.90 doesn’t match 10.10.1.32/27 =
00001010.00001010.00000001.01011010
```

For more information on CIDR, see http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing.

VPC Security Groups

If you're using Amazon Virtual Private Cloud, you must create *VPC security groups* specifically for your VPC instances. These groups work only inside the VPC where you've created them. Any EC2 security groups you have don't work inside your VPC. You can't create VPC security groups that reference EC2 groups, and you can't create EC2 security groups that reference VPC security groups..

You can have a VPC security group with the same name as an EC2 security group (it's possible because the groups have unique IDs). When working with VPC security groups, you must use the ID and not the name to identify the group.

VPC security groups have additional capabilities that EC2 security groups don't have. The following sections describe how to work with EC2 groups. To learn more about VPC security groups and how to work with them, go to Security Groups in the *Amazon Virtual Private Cloud User Guide*.

API and Command Overview

The following table summarizes the available commands and corresponding API actions for EC2 security groups. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-create-group` `CreateSecurityGroup`	Creates a new security group for use with your account.
`ec2-authorize` `AuthorizeSecurityGroupIngress`	Adds one or more rules to a security group.
`ec2-describe-group` `DescribeSecurityGroups`	Returns information about security groups associated with your account.
`ec2-revoke` `RevokeSecurityGroupIngress`	Removes one or more rules from a security group.
`ec2-delete-group` `DeleteSecurityGroup`	Deletes security groups associated with your account.

Note

VPC security groups have additional API actions and commands that apply to them. For more information, go to Security Groups in the *Amazon Virtual Private Cloud User Guide*.

Creating a Security Group

This section describes how to create a security group.

Note

You can create up to 500 EC2 security groups, with up to 100 rules per group.

Caution

Because an Amazon EC2 instance can belong to multiple security groups, more than 100 rules can apply to an instance. Associating hundreds of rules with an instance might cause problems when you access the instance. We recommend you condense your rules as much as possible.

AWS Management Console

To create a security group

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.

   The console displays a list of current security groups.

3. Click **Create Security Group**.
   The **Create Security Group** dialog box appears.
4. Configure the following settings and click **Yes, Create**.

   - Name
   - Description
   - For **VPC**, select `No VPC`.

   Amazon EC2 creates the security group and adds it to your list of groups.

Command Line Tools

To create a security group

- Enter the following command:

```
PROMPT>  ec2-create-group groupname -d "group_description"
```

Amazon EC2 returns information similar to the following example.

```
GROUP   sg-43de5aab   webservers   My Web Server Group
```

API

To create a security group

- Construct a Query request similar to the following example.

```
https://ec2.amazonaws.com/
?Action=CreateSecurityGroup
&GroupName=security-group-name
&GroupDescription=security-group-description
&...auth parameters...
```

Following is an example response.

```
<CreateSecurityGroupResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-
01/">
  <requestId>53effa54-17f1-47ae-ba5c-118b1d2617c2</requestId>
  <return>true</return>
  <groupdId>sg-43de5aab</groupId>
</CreateSecurityGroupResponse>
```

Describing Security Groups

This section describes how to view your security groups.

AWS Management Console

To view security groups

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. To view more information about a security group, including its rules, select it.
   The group's information is displayed in the lower pane.

Command Line Tools

To view security groups

- Enter the following command:

```
PROMPT>  ec2-describe-group [group ...]
```

  Amazon EC2 returns output similar to the following example.

```
GROUP   sg-455b6c31     999988887777    WebServers   web
PERMISSION      999988887777    WebServers ALLOWS  tcp     80      80
 FROM    CIDR    0.0.0.0/0       ingress
```

Tip

You can filter this list to return only certain security groups of interest to you. For more information about how to filter the results, go to ec2-describe-group in the *Amazon Elastic Compute Cloud Command Line Reference*.

API

To view security groups

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeSecurityGroups
&GroupName.1=security-group-name
&...auth parameters...
```

  Following is an example response.

```
<DescribeSecurityGroupsResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE>/requestId>
  <securityGroupInfo>
    <item>
```

```
      <ownerId>999988887777</ownerId>
      <groupId>sg-455b6c31</groupId>
      <groupName>WebServers</groupName>
      <groupDescription>Web</groupDescription>
      <vpcId/>
      <ipPermissions>
        <item>
          <ipProtocol>tcp</ipProtocol>
          <fromPort>80</fromPort>
          <toPort>80</toPort>
          <groups/>
          <ipRanges>
            <item>
              <cidrIp>0.0.0.0/0</cidrIp>
            </item>
          </ipRanges>
        </item>
      </ipPermissions>
      <ipPermissionsEgress/>
      <tagSet/>
    </item>
  </securityGroupInfo>
</DescribeSecurityGroupsResponse>
```

Tip

You can filter this list to return only certain security groups of interest to you. For more information about how to filter the results, go to DescribeSecurityGroups in the *Amazon Elastic Compute Cloud API Reference*.

Adding a Security Group Rule

This section describes how to add a rule to a security group.

When you add a rule to a security group, the new rule is automatically applied to any instances in the group.

Note

You can create up to 500 EC2 security groups, with up to 100 rules per group.

Caution

Because an Amazon EC2 instance can belong to multiple security groups, more than 100 rules can apply to an instance. Associating hundreds of rules with an instance might cause problems when you access the instance. We recommend you condense your rules as much as possible.

Note

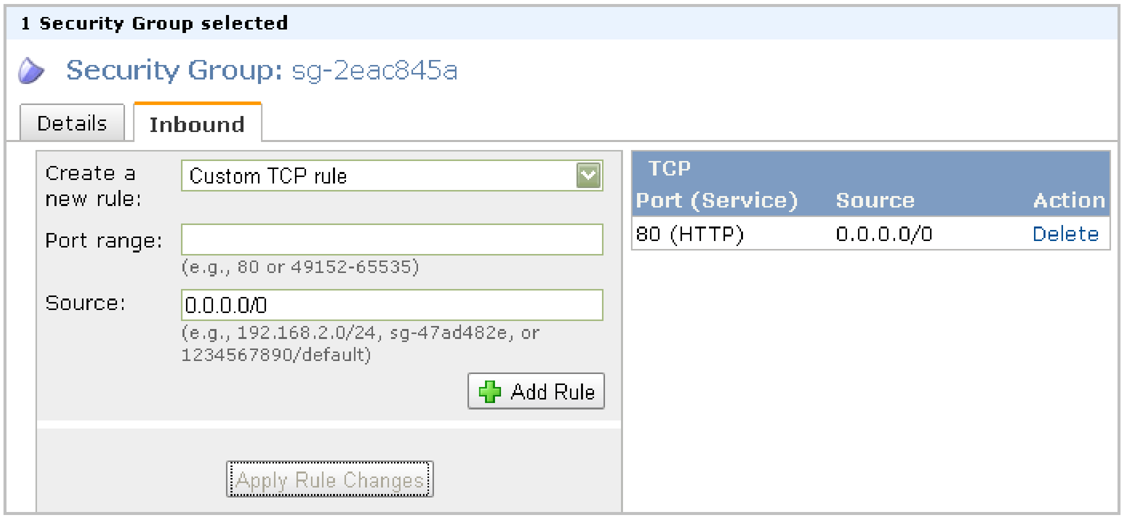
After an EC2 instance is running, you can't change which EC2 security groups it belongs to.

Exception: If you're using Amazon VPC, you can change which VPC security groups a VPC instance is in after launch. For more information, go to Security Groups in the *Amazon Virtual Private Cloud User Guide*.

AWS Management Console

To add a rule to a security group

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. Select an EC2 security group.
   Its rules appear on the **Inbound** tab in the lower pane.



1 Security Group selected

Security Group: sg-2eac845a

Create a new rule: Custom TCP rule

Port range:
(e.g., 80 or 49152-65535)

Source: 0.0.0.0/0
(e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

TCP		
Port (Service)	Source	Action
80 (HTTP)	0.0.0.0/0	Delete

4. To add a rule:

   a. From the **Create a new rule:** drop-down list, select the option you want.
   b. Specify a port or port range (if you've chosen a custom protocol rule).
   c. In the **Source** field, specify one of the following:

      - Name or ID of a security group (to allow access from that group). If the group isn't in your AWS account, prefix the group name with the AWS account ID and a forward slash (e.g., 111122223333/OtherSecurityGroup).
      - IP address range in CIDR notation (to allow access from that IP address range). For example, enter 0.0.0.0/0 to allow all IP addresses to access the specified port range. Enter an IP address or range of addresses to limit access to one computer or a network, for example 203.0.113.5/32.

5. Click **Add Rule**.
   An asterisk appears on the **Inbound** tab.
6. Click **Apply Rule Changes**.
   The new rule is created and applied to all instances that belong to the security group.

Command Line Tools

To add a rule to a security group

- Enter the following command:

```
PROMPT> ec2-authorize group [-P protocol] (-p port_range | -t icmp_type_code)
 [-u source_group_user ...] [-o source_group ...] [-s source_subnet ...]
```

For example, to modify the default security group to allow port 80 access from all IP addresses:

```
PROMPT>  ec2-authorize default -p 80
```

Amazon EC2 returns output similar to the following example.

```
GROUP   sg-2eac845a
PERMISSION                     ALLOWS  tcp     80      80      FROM    CIDR
    0.0.0.0/0       ingress
```

For example, to modify the default security group to allow SSH access from your own system's IP address:

```
PROMPT>  ec2-authorize -p 22 -s <your_ip_address>/32
```

Amazon EC2 returns output similar to the following example.

```
GROUP   sg-2eac845a
PERMISSION                     ALLOWS  tcp     22      22      FROM    CIDR
    <your_ip_address>/32 ingress
```

API

To add a rule to a security group

- Construct a Query request similar to the following.

```
https://ec2.amazonaws.com/
?Action=AuthorizeSecurityGroupIngress
&IpPermissions.1.IpProtocol=tcp
&IpPermissions.1.FromPort=80
&IpPermissions.1.ToPort=80
&IpPermissions.1.IpRanges.1.CidrIp=0.0.0.0/0
&...auth parameters...
```

Following is an example response.

```
<AuthorizeSecurityGroupIngressResponse xmlns="http://ec2.amazon
aws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <return>true</return>
</AuthorizeSecurityGroupIngressResponse>
```

Deleting a Security Group Rule

This section describes how to delete a security group rule.

When you delete a rule from a security group, the change is automatically applied to any instances in the group.

AWS Management Console

To delete a security group rule

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. Select a security group.
   Its rules appear on the **Inbound** tab in the lower pane.
4. To delete a rule, click **Delete** next to the rule.
   An asterisk appears on the **Inbound** tab.
5. Click **Apply Rule Changes**.
   Amazon EC2 deletes the security group rule.

Command Line Tools

The syntax of the `ec2-revoke` command is essentially identical to the syntax of the `ec2-authorize` command. To delete a rule, you specify the parts of the rule (e.g., protocol, port, CIDR range, etc.).

To delete a security group rule

- Enter the following command:

```
PROMPT>  ec2-revoke
group
[-P protocol]
(-p port_range | -t icmp_type_code)
[-u source_group_user ...]
[-o source_group ...]
[-s source_subnet ...]
```

  For example, to modify the default security group to remove port 80 access from all IP addresses:

```
PROMPT>  ec2-revoke default -p 80
```

  Amazon EC2 returns output similar to the following example.

```
GROUP   sg-2eac845a
PERMISSION                     ALLOWS  tcp     80      80      FROM    CIDR
    0.0.0.0/0       ingress
```

You can confirm the rule has been deleted by describing the security group with `ec2-describe-group`.

API

To delete a security group rule

- Construct a Query request similar to the following.

```
https://ec2.amazonaws.com/
?Action=RevokeSecurityGroupIngress
&IpPermissions.1.IpProtocol=tcp
&IpPermissions.1.FromPort=80
&IpPermissions.1.ToPort=80
&IpPermissions.1.IpRanges.1CidrIp=0.0.0.0/0
&...auth parameters...
```

Following is an example response.

```
<RevokeSecurityGroupIngressResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <return>true</return>
</RevokeSecurityGroupIngressResponse>
```

Deleting a Security Group

This section describes how to delete a security group.

The group must not have any instances in it. You can't delete the default security group.

AWS Management Console

To delete a security group

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Security Groups** in the **Navigation** pane.
   The console displays a list of security groups that belong to the account.
3. Select a security group and click **Delete**.
   A confirmation dialog box appears.
4. Click **Yes, Delete**.
   Amazon EC2 deletes the security group.

Command Line Tools

To delete a security group

- Enter the following command:

```
PROMPT>  ec2-delete-group group
```

Amazon EC2 returns output similar to the following:

```
RETURN true
```

API

To delete a security group

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DeleteSecurityGroup
&GroupName=security-group-name
&...auth parameters...
```

Following is an example response.

```
<DeleteSecurityGroupResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-
01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <return>true</return>
</DeleteSecurityGroupResponse>
```

Examples

This section provides examples of configuring EC2 security groups using the command line tools.

Note

In addition to these examples, you can maintain your own firewall on any of your instances. This can be useful if you have specific requirements not met by the Amazon EC2 distributed firewall.

Modifying the Default Group

This example shows Albert modifying the default group to meet his security needs.

Albert Modifies the Default Group

1

Albert launches a copy of his favorite public AMI.

```
PROMPT> ec2-run-instances ami-eca54085
RESERVATION r-a034c7c9 999988887777 default
INSTANCE i-cfd732a6 ami-eca54085 pending 0 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs
```

2

Albert, who is a cautious type, checks the access rules of the default group.

```
PROMPT> ec2-describe-group default
GROUP   sg-2eac845a     999988887777    default default group
PERMISSION      999988887777    default ALLOWS  icmp    -1      -1
    FROM    USER    999988887777    NAME default    ID sg-2eac845a
ingress
PERMISSION      999988887777    default ALLOWS  tcp     0       65535
   FROM    USER    999988887777    NAME default    ID sg-2eac845a
ingress
PERMISSION      999988887777    default ALLOWS  udp     0       65535
   FROM    USER    999988887777    NAME default    ID sg-2eac845a
ingress
```

Albert notices that it only accepts ingress network connections from other members of the default group for all available protocols (TCP, UDP, ICMP) and ports.

3

Albert, being cautious, uses the Linux/UNIX nmap command to port scan his instance.

```
$ nmap -P0 -p1-100 ec2-203-0-113-5.compute-1.amazonaws.com
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2010-03-30
15:06 SAST
All 100 scanned ports on ec2-203-0-113-5.compute-1.amazonaws.com
(203.0.113.5) are: filtered

Nmap finished: 1 IP address (1 host up) scanned in 31.008 seconds
```

4

Albert decides he should be able to SSH into his instance, but only from his own machine.

```
PROMPT> ec2-authorize default -P tcp -p 22 -s 203.0.113.7/32
GROUP   sg-2eac845a
PERMISSION   ALLOWS  tcp  22  22  FROM  CIDR  203.0.113.7/32  ingress
```

5 Albert repeats the Linux/UNIX `nmap` port scan.

```
$ nmap -P0 -p1-100 ec2-203-0-113-5.compute-1.amazonaws.com
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2010-03-30
15:07 SAST
Interesting ports on ec2-203-0-113-5.compute-1.amazonaws.com
(203.0.113.5):
(The 99 ports scanned but not shown are in state: filtered)
PORT   STATE SERVICE
22/tcp open  ssh

Nmap finished: 1 IP address (1 host up) scanned in 32.705 seconds
```

Albert is happy.

Creating a Three-Tier Web Service

Mary wants to deploy her public, failure resilient, three-tier web service (web, application, and database servers) in Amazon EC2. Her grand plan is to have her web tier start off executing in seven instances of ami-fba54092, her application tier executing in twenty instances of ami-e3a5408a,and her multi-master database in two instances of ami-f1a54098. She's concerned about the security of her subscriber database, so she wants to restrict network access to her middle and back tier machines. When the traffic to her site increases over the holiday shopping period, she adds additional instances to her web and application tiers to handle the extra load.

Launch Process

1 First, Mary creates a group for her Apache web server instances and allows HTTP access to the world.

```
PROMPT> ec2-create-group apache -d "Mary's Apache group"
GROUP   sg-ed5b6c99   apache  Mary's Apache group

PROMPT> ec2-describe-group apache
GROUP   sg-ed5b6c99   999988887777    apache  Mary's Apache group

PROMPT> ec2-authorize apache -P tcp -p 80 -s 0.0.0.0/0
GROUP   sg-ed5b6c99
PERMISSION  999988887777 apache  ALLOWS  tcp   80    80    FROM
CIDR    0.0.0.0/0   ingress

PROMPT> ec2-describe-group apache
GROUP   sg-ed5b6c99  999988887777    apache  Mary's Apache group
PERMISSION   999988887777   apache  ALLOWS  tcp   80   80   FROM
CIDR   0.0.0.0/0  ingress
```

2

Mary launches seven instances of her web server AMI as members of the apache group.

```
PROMPT> ec2-run-instances ami-fba54092 -n 7 -g apache
RESERVATION r-0592776c 999988887777 apache
INSTANCE i-cfd732a6 ami-fba54092 pending 0 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732a7 ami-fba54092 pending 1 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732a8 ami-fba54092 pending 2 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732a9 ami-fba54092 pending 3 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732aa ami-fba54092 pending 4 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732ab ami-fba54092 pending 5 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99
INSTANCE i-cfd732ac ami-fba54092 pending 6 m1.small 2010-03-
19T13:59:03+0000 us-east-1a aki-94c527fd ari-96c527ff monitoring-
disabled  ebs   paravirtual   xen   sg-ed5b6c99


PROMPT> ec2-describe-instances i-cfd732a6
RESERVATION     r-0592776c      999988887777  apache
INSTANCE        i-cfd732a6      ami-fba54092       ec2-203-0-113-
12.compute-1.amazonaws.com       running 0
m1.small   2010-03-30T08:43:48+0000       us-east-1a     aki-94c527fd
    ari-96c527ff    monitoring-disabled      203.0.113.12
10.254.137.191                  ebs   paravirtual   xen   sg-ed5b6c99
BLOCKDEVICE     /dev/sda1       vol-cf13b3a6    2010-03-
30T08:01:44.000Z
```

3

Being as cautious as Albert, Mary uses the Linux/UNIX nmap command to confirm the permissions she just configured.

```
$ nmap -P0 -p1-100 ec2-203-0-113-12.compute-1.amazonaws.com
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2010-03-30
15:10 SAST
Interesting ports on ec2-203-0-113-12.compute-1.amazonaws.com
(203.0.113.12):
(The 99 ports scanned but not shown are in state: filtered)
PORT   STATE SERVICE
80/tcp open  http

Nmap finished: 1 IP address (1 host up) scanned in 33.409 seconds
```

4 Mary verifies her web server can be reached.

```
$ telnet ec2-203-0-113-12.compute-1.amazonaws.com  80
Trying 203.0.113.12...
Connected to ec2-203-0-113-12.compute-1.amazonaws.com  (203.0.113.12).
Escape character is '^]'.
```

Mary can reach her web server.

5 Mary creates a separate group for her application server.

```
PROMPT> ec2-create-group appserver -d "Mary's app server"
GROUP   sg-e95b6c9d     appserver       Mary's app server
```

6 Mary starts twenty instances as members of `appserver` group.

```
PROMPT> ec2-run ami-e3a5408a -n 20 -g appserver
```

7 Mary grants network access between her web server group and the application server group.

```
PROMPT> ec2-authorize appserver -o apache -u 999988887777
GROUP                    appserver
PERMISSION               appserver       ALLOWS  tcp     0       65535
   FROM    USER    999988887777    GRPNAME apache            ingress
PERMISSION               appserver       ALLOWS  udp     0       65535
   FROM    USER    999988887777    GRPNAME apache            ingress
PERMISSION                appserver       ALLOWS  icmp    -1       -1
    FROM    USER    999988887777    GRPNAME apache           ingress
```

8 Mary verifies access to her app server is restricted by port scanning one of the application servers using the Linux and UNIX `nmap` command.

```
$ nmap -P0 -p1-100 ec2-203-0-113-9.compute-1.amazonaws.com
Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2010-03-30
15:11 SAST
All 100 scanned ports on ec2-203-0-113-9.compute-1.amazonaws.com
(203.0.113.9) are: filtered

Nmap finished: 1 IP address (1 host up) scanned in 31.008 seconds
```

<table>
<tr>
<td>9</td>
<td>
<p>Mary confirms that her web servers have access to her application servers.</p>
<p>A. She (temporarily) grants SSH access from her workstation to the web server group:</p>
<pre>
PROMPT> <b>ec2-authorize apache -P tcp -p 22 -s 203.0.113.19/32</b>
</pre>
<p>B. She logs in to one of her web servers and connects to an application server on TCP port 8080.</p>
<pre>
$ telnet ec2-203-0-113-9.compute-1.amazonaws.com 8080
Trying 203.0.113.9...
Connected to ec2-203-0-113-9.compute-1.amazonaws.com (203.0.113.9).
Escape character is '^]'
</pre>
<p>C. Satisfied with the setup, she revokes SSH access to the web server group.</p>
<pre>
PROMPT> <b>ec2-revoke apache -P tcp -p 22 -s 203.0.113.19/32</b>
</pre>
</td>
</tr>
<tr>
<td>10</td>
<td>Mary repeats these steps to create the database server group and to grant access between the application server and database server groups.</td>
</tr>
</table>

Using Regions and Availability Zones

Topics

This section describes how to work with Regions and Availability Zones.

Note

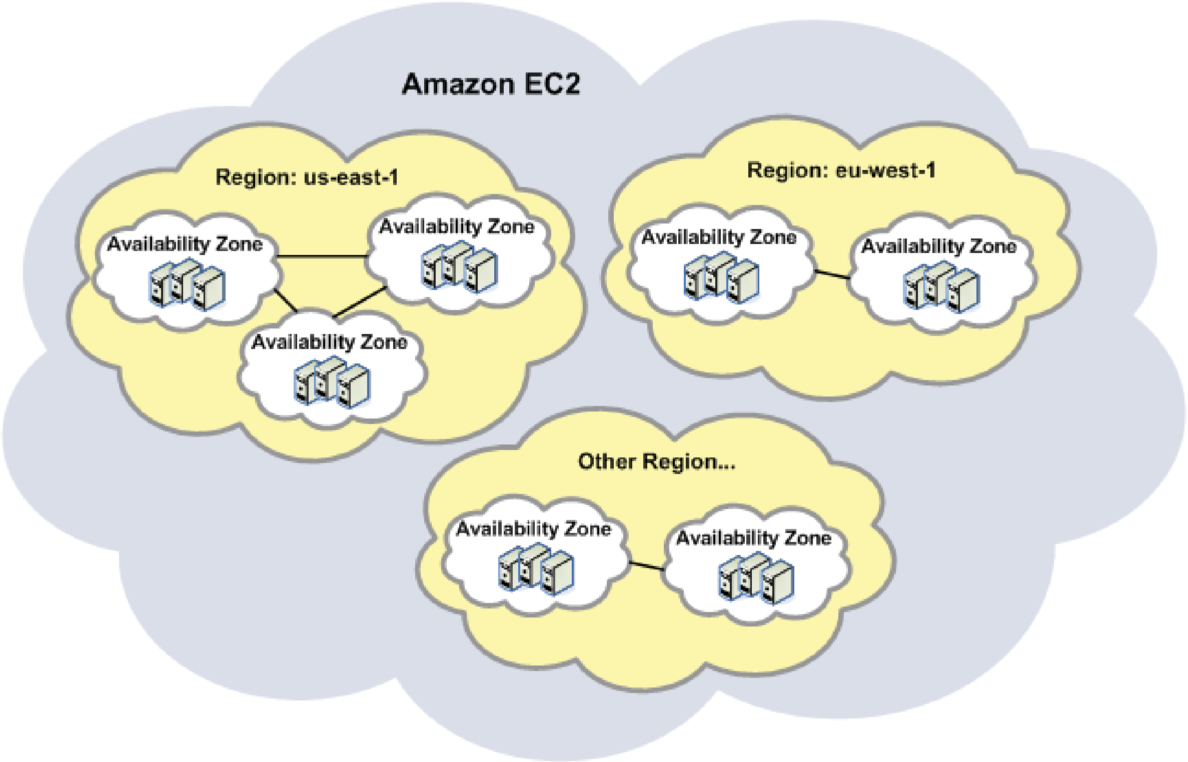
Data transfer between Regions is charged at the Internet data transfer rate for both the sending and the receiving Region. For detailed information on Amazon EC2 charges, go to the Amazon EC2 Product Page.

Region and Availability Zone Concepts

Amazon EC2 provides the ability to place instances in multiple locations. Amazon EC2 locations are composed of Availability Zones and Regions. Regions are dispersed and located in separate geographic areas (US, EU, etc.). Availability Zones are distinct locations within a Region that are engineered to be isolated from failures in other Availability Zones and provide inexpensive, low latency network connectivity to other Availability Zones in the same Region.

By launching instances in separate Regions, you can design your application to be closer to specific customers or to meet legal or other requirements. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.

The following graphic shows a representation of Amazon EC2. Each Region is completely independent. Each Availability Zone is isolated, but connected through low-latency links.



Regions

Amazon EC2 provides multiple Regions so you can launch Amazon EC2 instances in locations that meet your requirements. For example, you might want to launch instances in Europe to be closer to your European customers or to meet legal requirements.

Each Amazon EC2 Region is designed to be completely isolated from the other Amazon EC2 Regions. This achieves the greatest possible failure independence and stability, and it makes the locality of each EC2 resource unambiguous.

To launch or work with instances, you must specify the correct Region URL endpoint. For example, to access the US-East Region (default), you make service calls to the ec2.us-east-1.amazonaws.com service endpoint.

For information about this product's regions and endpoints, go to Regions and Endpoints in the Amazon Web Services General Reference.

Availability Zones

Amazon operates state-of-the-art, highly available data center facilities. However, failures can occur that affect the availability of instances that are in the same location. Although this is rare, if you host all your Amazon EC2 instances in a single location that is affected by such a failure, your instances will be unavailable.

For example, if you have instances distributed across three Availability Zones and one of the instances fails, you can design your application so the instances in the remaining Availability Zones handle any requests.

Note

You can use Availability Zones in conjunction with elastic IP addresses to remap IP addresses across Availability Zones. For information on elastic IP addresses, see Elastic IP Addresses Concepts (p. 293).

API and Command Overview

The following table summarizes the available Region and Availability Zone commands and corresponding API actions. For more information about the commands, go to the Amazon Elastic Compute Cloud Command Line Reference. For more information about the API actions, go to the Amazon Elastic Compute Cloud API Reference.

Command and API Action	Description
`ec2-describe-availability-zones` `DescribeAvailabilityZones`	Lists Availability Zones available to your account
`ec2-describe-regions` `DescribeRegions`	Lists Regions available to your account

Describing Regions and Availability Zones

This section describes how to determine which Regions and Availability Zones are available.

AWS Management Console

To find Regions and Availability Zones

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. To determine available Regions, select the **Region** list box in the **Navigation** pane.

   After selecting a Region, you can view Availability Zones within that Region when launching an instance or creating a new Amazon EBS volume.
3. To view Availability Zones within a Region, click **Volumes** in the **Navigation** pane.

   The **Create Volume** dialog box appears.
4. To view Availability Zones within the Region, select the **Availability Zones** list box.
5. When you are finished, click **Cancel**.

Command Line Tools

To find Regions

- Enter the following command to describe Regions:

```
PROMPT> ec2-describe-regions
REGION  ap-northeast-1       ec2.ap-northeast-1.amazonaws.com
REGION  ap-southeast-1       ec2.ap-southeast-1.amazonaws.com
..
```

To find Availability Zones

- Enter the following command to describe Availability Zones within the `us-east-1` Region:

```
PROMPT> ec2-describe-availability-zones --region us-east-1
```

Amazon EC2 returns output similar to the following:

```
AVAILABILITYZONE        us-east-1a      available       us-east-1
AVAILABILITYZONE        us-east-1b      available       us-east-1
AVAILABILITYZONE        us-east-1c      available       us-east-1
AVAILABILITYZONE        us-east-1d      available       us-east-1
```

API

To find Regions

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeRegions
&...auth parameters...
```

Following is an example response.

```
<DescribeRegionsResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>59dbff89-35bd-4eac-99ed-be587EXAMPLE</requestId>
  <regionInfo>
    <item>
      <regionName>us-east-1</regionName>
      <regionEndpoint>ec2.us-east-1.amazonaws.com</regionEndpoint>
    </item>
    <item>
      <regionName>eu-west-1</regionName>
      <regionEndpoint>ec2.eu-west-1.amazonaws.com</regionEndpoint>
    </item>
  </regionInfo>
</DescribeRegionsResponse>
```

To find Availability Zones

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=DescribeAvailabilityZones
&...auth parameters...
```

Following is an example response.

```
<DescribeAvailabilityZonesResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <requestId>35a9e5f3-4ed2-4cc2-a616-482b71c82c76</requestId>
      <availabilityZoneInfo>
        <item>
          <zoneName>us-east-1a</zoneName>
          <zoneState>available</zoneState>
          <regionName>us-east-1</regionName>
          <messageSet/>
        </item>
        <item>
          <zoneName>us-east-1b</zoneName>
          <zoneState>available</zoneState>
          <regionName>us-east-1</regionName>
          <messageSet/>
        </item>
        <item>
          <zoneName>us-east-1c</zoneName>
          <zoneState>available</zoneState>
          <regionName>us-east-1</regionName>
          <messageSet/>
        </item>
        <item>
          <zoneName>us-east-1d</zoneName>
          <zoneState>available</zoneState>
          <regionName>us-east-1</regionName>
          <messageSet/>
        </item>
      </availabilityZoneInfo>
</DescribeAvailabilityZonesResponse>
```

Specifying the Region to Use

Every command or call you make to EC2 must target a specific Region. This section explains how to specify the Region you want to use.

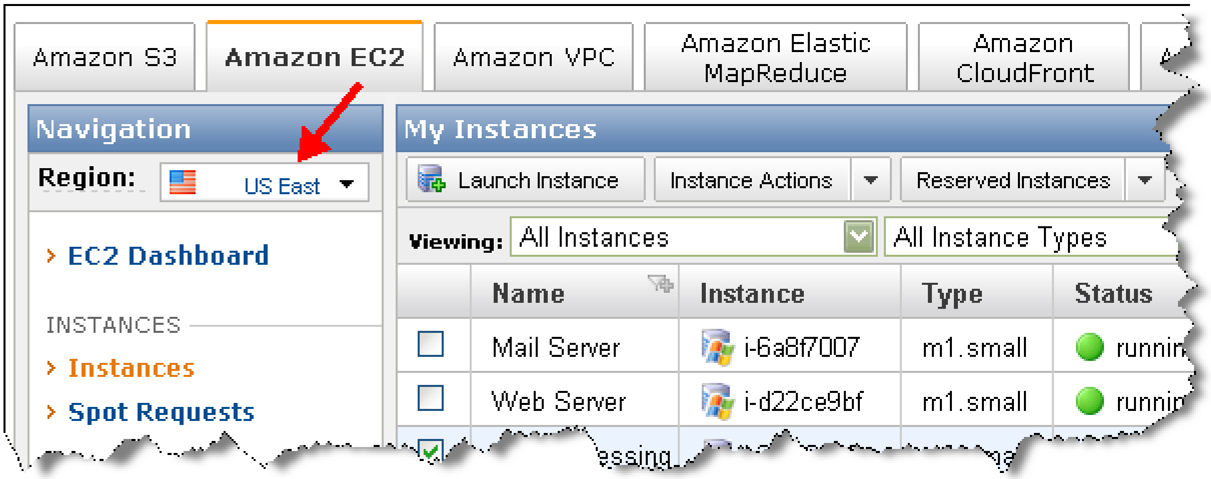
Note

The `ec2.us-west-1.amazonaws.com` Region is the original Amazon EC2 Region and is selected by default.

AWS Management Console

To specify the Region

- Use the Region selector in the top left corner of the page.



	Name	Instance	Type	Status
☐	Mail Server	i-6a8f7007	m1.small	● runnin
☐	Web Server	i-d22ce9bf	m1.small	● runnin

Command Line Tools

To specify the Region

- Change the `EC2_URL` environment variable's value to the Region's endpoint (e.g., https://ec2.us-west-1.amazonaws.com).

  Tip

  You can also use the `--region` command line option (e.g., `--region us-west-1`), or override the URL endpoint using the `-U` flag (e.g., `-U https://ec2.us-west-1.amazonaws.com`).

API

To specify the Region

- Configure your application to use the Region's endpoint (e.g., https://ec2.us-west-1.amazonaws.com).

Launching Instances in a Specific Availability Zone

When you launch an instance, you can optionally specify an Availability Zone. If you do not specify an Availability Zone, Amazon EC2 selects one for you in the Region that you are using. When launching your initial instances, we recommend accepting the default Availability Zone, which allows Amazon EC2 to select the best Availability Zone for you based on system health and available capacity. Even if you have other instances running, you might consider not specifying an Availability Zone if your new instances do not need to be close to, or separated from, your existing instances.

Note

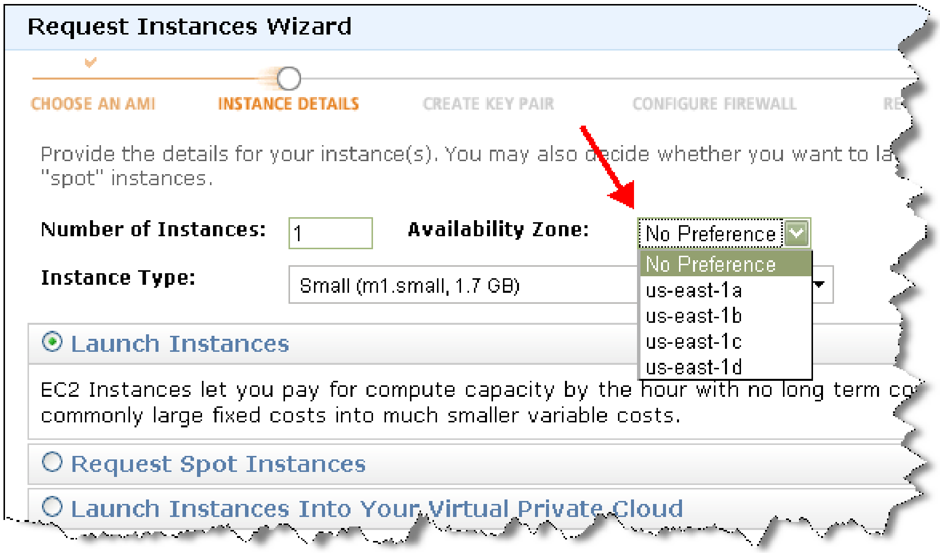
Availability Zones are not the same across accounts. The Availability Zone us-east-1a for account A is not necessarily the same as us-east-1a for account B. Zone assignments are mapped independently for each account.

You are charged a small bandwidth charge for data that crosses Availability Zones. For more information, go to the Amazon EC2 product page.

AWS Management Console

To launch an instance in a specific Availability Zone

- On the **Instance Details** page of the instance launch wizard, you can specify the Availability Zone of your choice, or specify no preference.



Request Instances Wizard

Provide the details for your instance(s). You may also decide whether you want to la
"spot" instances.

Number of Instances: 1

Availability Zone: No Preference

Instance Type: Small (m1.small, 1.7 GB)

● Launch Instances

EC2 Instances let you pay for compute capacity by the hour with no long term co
commonly large fixed costs into much smaller variable costs.

○ Request Spot Instances

○ Launch Instances Into Your Virtual Private Cloud

Command Line Tools

To launch an instance in a specific Availability Zone

- Specify the Availability Zone you want to launch the instances into using the `--availability-zone` option.

```
PROMPT> ec2-run-instances ami_id -n count --availability-zone zone
```

Amazon EC2 returns output similar to the following:

```
RESERVATION r-0ea54067 999988887777 default
INSTANCE i-cfd732a6 ami-6ba54002 pending 0 m1.small 2010-03-19T13:59:03+0000
 us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-cfd732a7 ami-6ba54002 pending 1 m1.small 2010-03-19T13:59:03+0000
 us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-cfd732a8 ami-6ba54002 pending 2 m1.small 2010-03-19T13:59:03+0000
 us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
INSTANCE i-cfd732a9 ami-6ba54002 pending 3 m1.small 2010-03-19T13:59:03+0000
 us-east-1a aki-94c527fd ari-96c527ff monitoring-disabled  ebs
```

API

To launch an instance in a specific Availability Zone

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=RunInstances
&ImageId=ami-id
&MaxCount=1
&MinCount=1
&KeyName=keypair-name
&Placement.AvailabilityZone=zone
&...auth parameters...
```

Related Topics

- Region and Availability Zone FAQ (p. 412)

Micro Instances

Topics

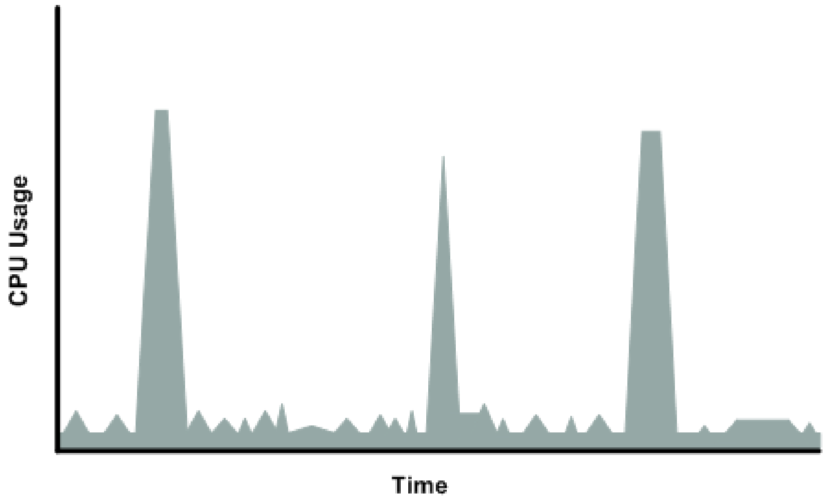
Micro instances (t1.micro) provide a small amount of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically.

This section describes how micro instances generally work so you can understand how to apply them. Our intent is not to specify exact behavior, but to give you visibility into the instance's behavior so you can understand its performance (to a greater degree than you typically get with, for example, a multi-tenant shared web hosting system).

The micro instance is available on both 32-bit and 64-bit platforms, but it's available as an Amazon EBS-backed instance only. For basic specifications for the micro instance type, see Available Instance Types (p. 70).

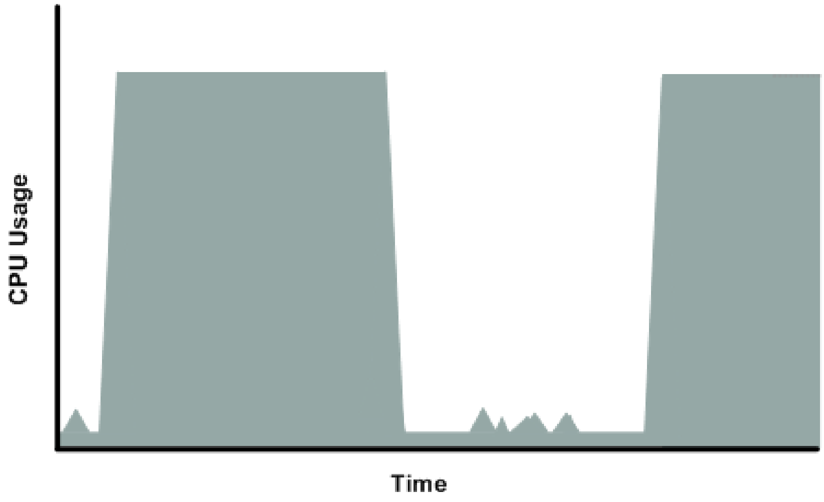
Optimal Application of Micro Instances

Micro instances provide spiky CPU resources for workloads that have a CPU usage profile similar to what is shown in the following figure.

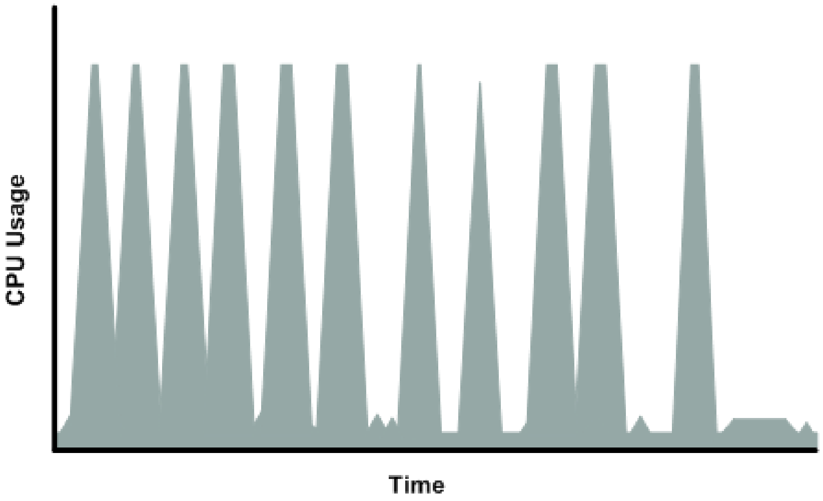


The instance is designed to operate with its CPU usage at essentially only two levels: the normal low background level, and then at brief spiked levels much higher than the background level. We allow the instance to operate at up to 2 EC2 compute units (ECUs) (for more information, see Compute Resources Measurement (p. 72)). The ratio between the maximum level and the background level is designed to be large. Micro instances are designed to support tens of requests per minute on your application. However, actual performance can vary significantly depending on the amount of CPU resources required for each request on your application.

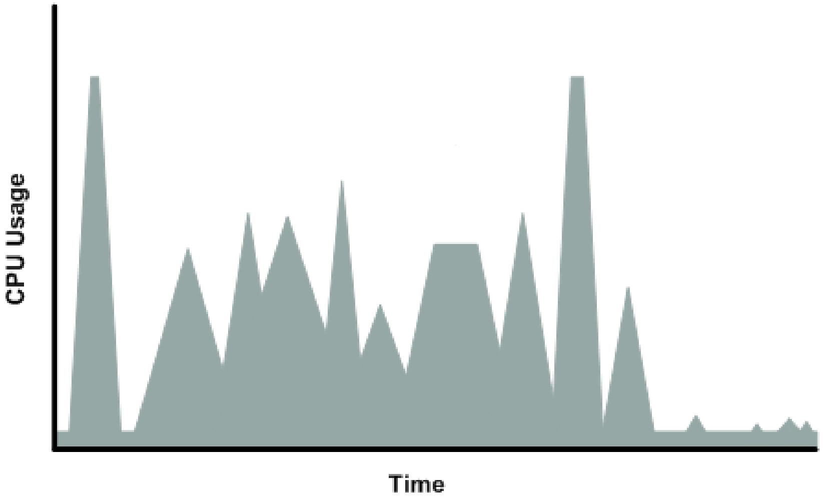
Your application might have a different CPU usage profile than that described in the preceding section. The next figure shows the profile for an application that isn't appropriate for a micro instance. The application requires continuous data-crunching CPU resources for each request, resulting in plateaus of CPU usage that the micro instance isn't designed to handle.



The next figure shows another profile that isn't appropriate for a micro instance. Here the spikes in CPU use are brief, but they occur too frequently to be serviced by a micro instance.



The next figure shows another profile that isn't appropriate for a micro instance. Here the spikes aren't too frequent, but the background level between spikes is too high to be serviced by a micro instance.



In each of the preceding cases of workloads not appropriate for a micro instance, we recommend you consider using a different instance type (for more information about the instance types, see Instance Families and Types (p. 69)).

Available CPU Resources During Spikes

When your instance *bursts* to accommodate a spike in demand for compute resources, it uses unused resources on the host. The amount available depends on how much contention there is when the spike occurs. The instance is never left with no CPU resources, whether other instances on the host are spiking or not.

When the Instance Uses Its Allotted Resources

We expect your application to consume only a certain amount of CPU resources in a period of time. If the application consumes more than your instance's allotted CPU resources, we temporarily limit the instance so it operates at a low CPU level. If your instance continues to use all of its allotted resources, its performance will degrade. We will increase the time we limit its CPU level, thus increasing the time before the instance is allowed to burst again.

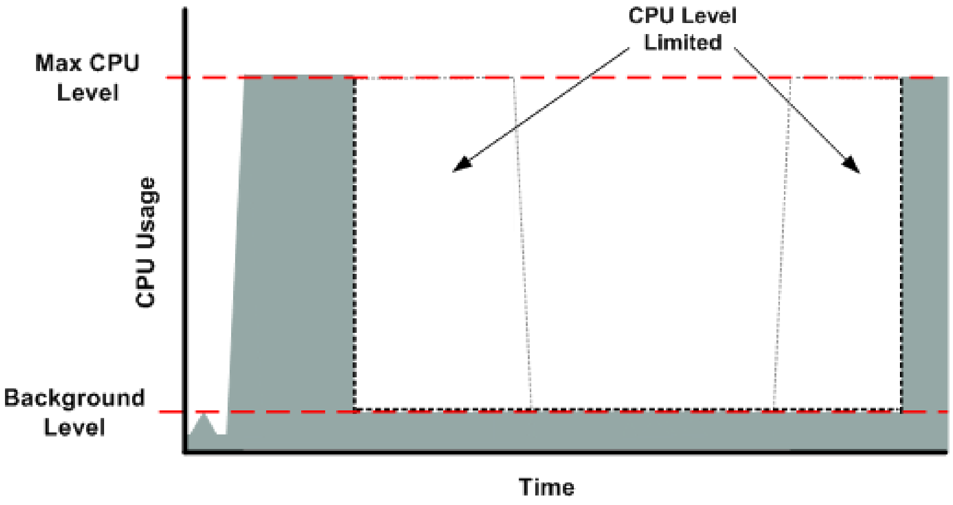
If you enable Amazon CloudWatch monitoring for your micro instance, you can use the "Avg CPU Utilization" graph in the AWS Management Console to determine whether your instance is regularly using all its allotted CPU resources. We recommend that you look at the maximum value reached during each given period. If the maximum value is 100%, we recommend that you use Auto Scaling to scale out (with additional micro instances and a load balancer), or move to a larger instance type. For more information about Auto Scaling, go to the Auto Scaling Developer Guide.

Note

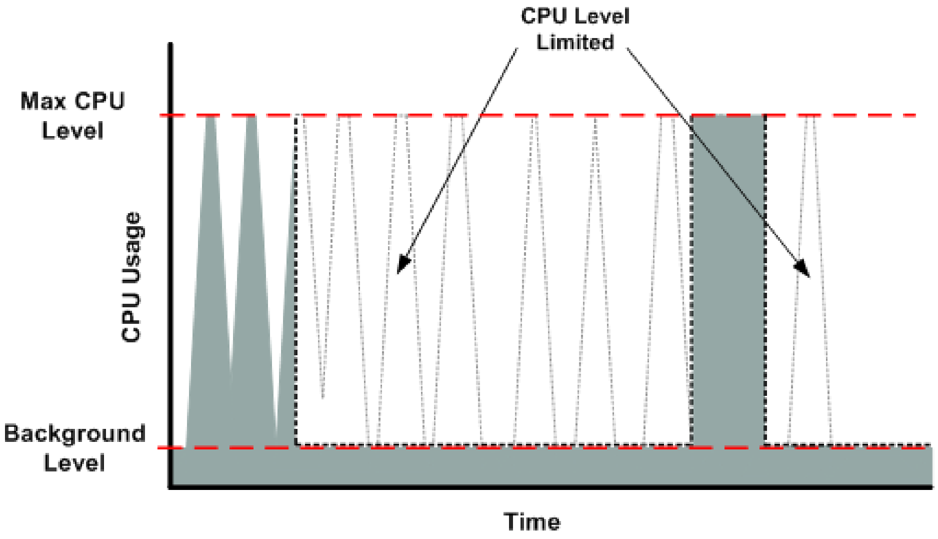
Micro instances are available as either 32-bit or 64-bit. If you need to move to a larger instance, make sure to move to a compatible instance type.

The following figures show the three suboptimal profiles from the preceding section and what it might look like when the instance consumes its allotted resources and we have to limit its CPU level. If the instance consumes its allotted resources, we restrict it to the low background level.

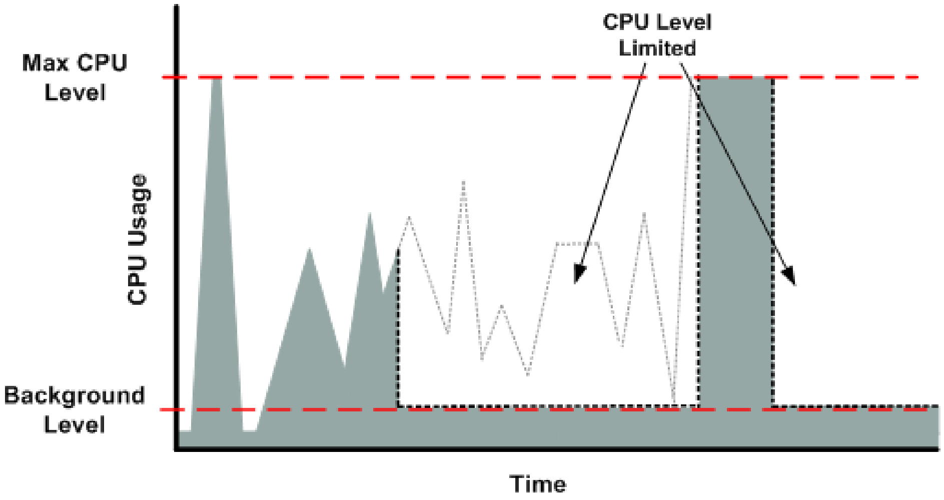
The next figure shows the situation with the long plateaus of data-crunching CPU usage. The CPU hits the maximum allowed level and stays there until the instance's allotted resources are consumed for the period. At that point, we limit the instance to operate at the low background level, and it operates there until we allow it to burst above that level again. The instance again stays there until the allotted resources are consumed and we limit it again (not seen on the graph).



The next figure shows the situation where the requests are too frequent. The instance uses its allotted resources after only a few requests and so we limit it. After we lift the restriction, the instance maxes out its CPU usage trying to keep up with the requests, and we limit it again.

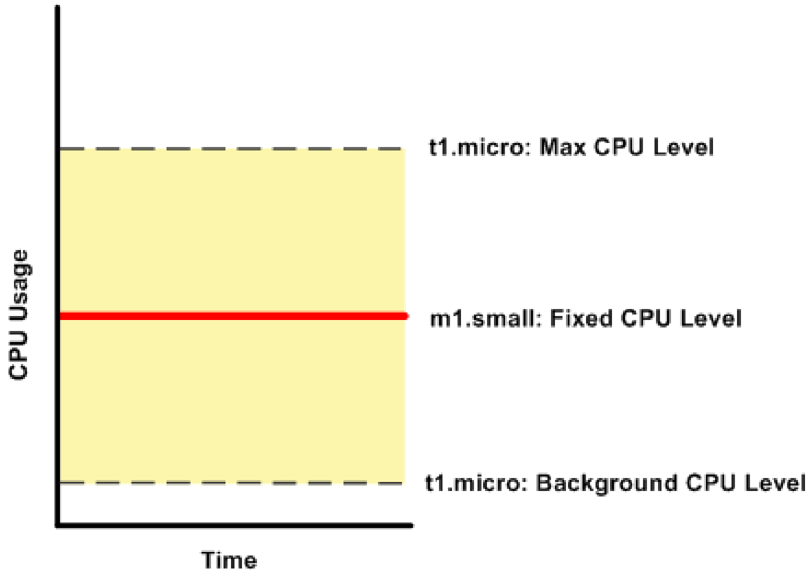


The next figure shows the situation where the background level is too high. Notice that the instance doesn't have to be operating at the maximum CPU level for us to limit it. We limit the instance when it's operating above the normal background level and has consumed its allotted resources for the given period. In this case (as in the preceding one), the instance can't keep up with the work, and we limit it again.



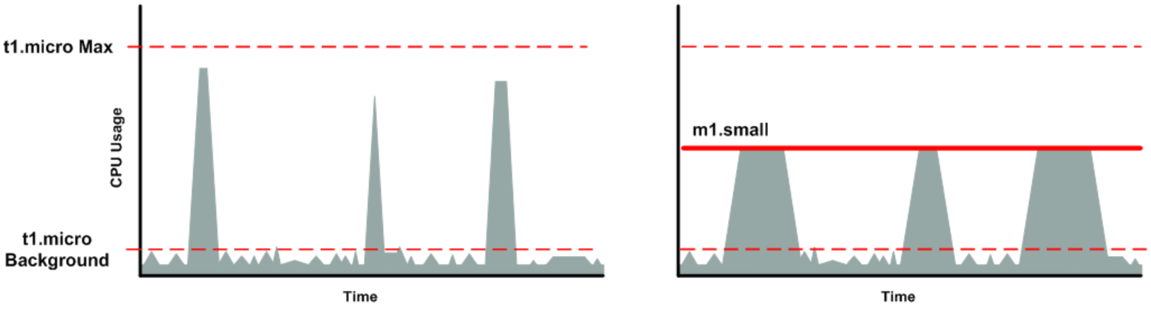
Comparison with the m1.small Instance Type

The micro instance provides different levels of CPU resources at different times (up to 2 ECUs). By comparison, the m1.small instance type provides 1 ECU at all times. The following figure illustrates the difference.

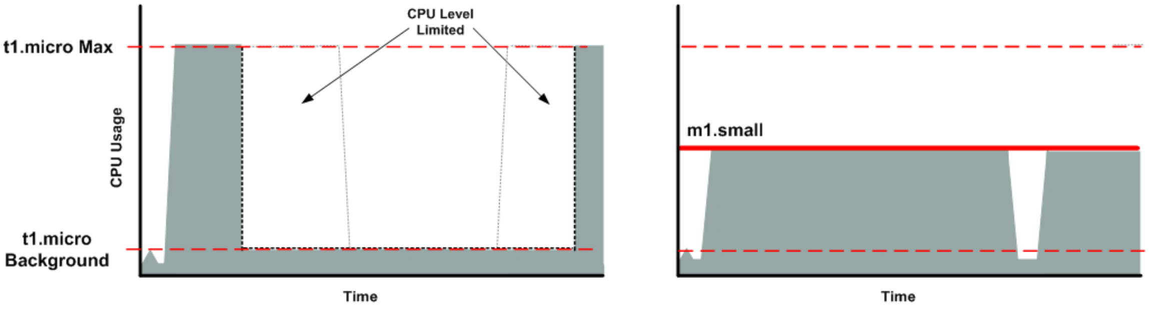


The following figures compare the CPU usage of a micro instance with an m1.small for the various scenarios we've discussed in the preceding sections.

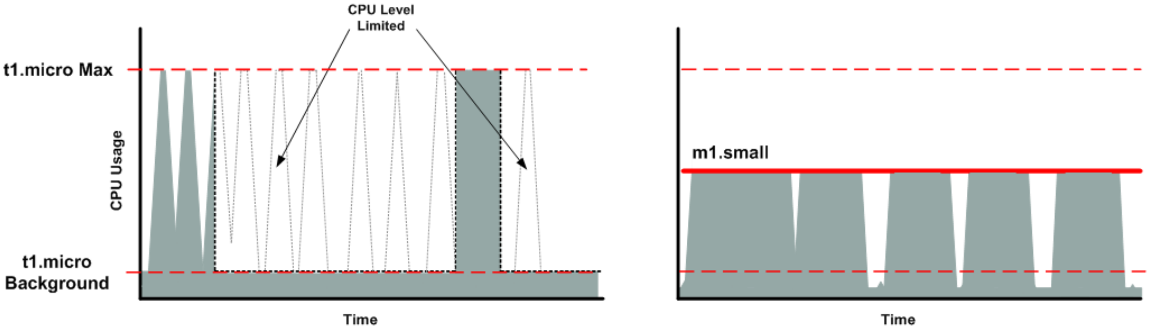
The first figure that follows shows an optimal scenario for a micro instance (the left graph) and how it might look for an m1.small (the right graph). In this case, we don't need to limit the micro instance. The processing time on the m1.small would be longer for each spike in CPU demand compared to the micro instance.



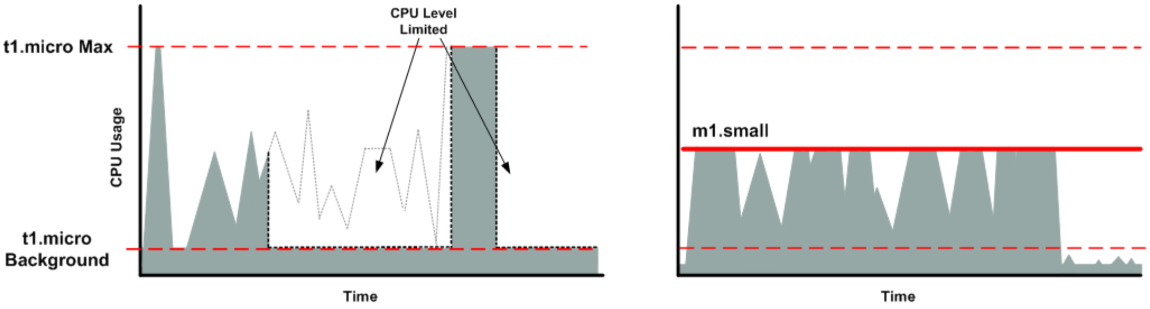
The next figure shows the scenario with the data-crunching requests that used up the allotted resources on the micro instance, and how they might look with the m1.small instance.



The next figure shows the frequent requests that used up the allotted resources on the micro instance, and how they might look on the m1.small instance.



The next figure shows the situation where the background level used up the allotted resources on the micro instance, and how it might look on the m1.small instance.



AMI Optimization for Micro Instances

We recommend you follow these best practices when optimizing an AMI for the micro instance type:

- Design the AMI to run on 600 MB of RAM
- Limit the number of recurring processes that use CPU time (e.g., cron jobs, daemons)

In Linux, you can optimize performance using swap space and virtual memory (e.g., by setting up swap space in a separate partition from the root file system).

In Windows, when you perform significant AMI or instance configuration changes (e.g., enable server roles or install large applications), you might see limited instance performance, because these changes can be memory intensive and require long-running CPU resources. We recommend you first use a larger instance type when performing these changes to the AMI, and then run the AMI on a micro instance for normal operations.

Related Topics

- Using Amazon EBS-Backed AMIs and Instances (p. 177)
- Monitoring Your Instances and Volumes with CloudWatch (p. 348)

Using Cluster Instances

Topics

This section describes cluster instances and how to create HVM AMIs. You'll also learn how to create placement groups and then launch cluster instances into them.

Cluster Instance Concepts

Topics

Amazon EC2 offers two *cluster instance types*: cluster compute instances and cluster GPU instances. These instance types provide a very large amount of CPU coupled with increased networking performance, making them well suited for High Performance Compute (HPC) applications and other demanding network-bound applications. Example categories of such applications include computational biology and chemistry, industrial design, and financial analytics. You can group cluster instances into clusters (known as *cluster placement groups*), allowing applications to get the low-latency network performance required for tightly coupled, node-to-node communication typical of many HPC applications.

Cluster GPU instances have the same hardware specifications as cluster compute instances. However, the cluster GPU instances also include two NVIDIA Tesla M2050 "Fermi" GPUs, making them well suited for HPC applications as well as rendering and media processing applications.

Cluster instances interoperate with other Amazon EC2 features and AWS products just as do other Amazon EC2 instance types. They also function similarly to other Amazon EC2 instance types, but with the important differences described in the following table and sections.

Feature	Description
Cluster Placement Groups	You can group cluster instances in a *cluster placement group* to provide lower latency and high-bandwidth connectivity between the instances. You just create a cluster placement group and then launch the instances into it. For information about creating cluster placement groups, see Creating a Cluster Placement Group (p. 339).
HVM-Based Virtualization	Cluster instances run as Hardware Virtual Machine (HVM)-based instances. HVM is a type of virtualization that uses hardware-assist technology provided by the AWS platform. HVM virtualization lets the guest VM run as though it is on a native hardware platform with the exception that it still uses paravirtual (PV) network and storage drivers for improved performance. For more information, see HVM-Based AMIs (p. 336).
Amazon EBS-backed, HVM-based AMIs	You must launch cluster instances from Amazon EBS-backed, HVM-based AMIs. Launching an instance from Amazon EC2 instance store-backed AMIs is not supported. AWS labels all AMIs as either `hvm` or `paravirtual` so you can determine the virtualization type. For information about using Amazon EBS-backed AMIs, see Using Amazon EBS-Backed AMIs and Instances (p. 177).

Current Limitations

Cluster instances currently have the following limitation:

- They are available only in the US-East (Northern Virginia) Region
- They are not available to use with Amazon Virtual Private Cloud
- They are not available to use with Amazon DevPay

Cluster placement groups currently have the following limitation:

- Reserved Instances are not guaranteed within specific requested cluster placement group.

HVM-Based AMIs

Cluster instances require the use of HVM-based, Amazon EBS-backed AMIs. Paravirtual (PV) AMIs employed by Linux instances and other EC2 instance types that use paravirtual machine (PVM)-based virtualization are not supported. Cluster instances do not require the use of Amazon Kernel Images (AKIs) or Amazon RAM disk Images (ARIs) that are part of PVM-based AMIs used with other EC2 instance types.

Reference AMI

To help you get started with cluster instances, AWS provides a reference Amazon Linux AMI. The AMI includes the NVIDIA driver and CUDA toolkit, which enable the NVIDIA GPUs. This Amazon Linux AMI has been built to be binary compatible with the CentOS 5.x series of releases and is very similar to CentOS in file system layout and package management. Applications built for CentOS 5 and later should run without recompilation of the Amazon Linux AMI. For information about this reference AMI, go to the Amazon Linux AMI details at http://-aws.amazon.com/-amazon-linux-ami/. For information on using the reference AMI, see Amazon Linux AMI Basics (p. 55). You can customize an instance and create your own AMI from it (for more information, see Creating an HVM AMI from a Running Instance (p. 338)).

AWS CloudFormation Templates for Cluster Instances

To make it easy for you to launch a cluster, you can use AWS CloudFormation templates that provision a two-node cluster with NFS storage, monitoring and a mounted 200 GB data set.

Each template performs the following tasks:

1. Creates a new controller instance.
2. Bootstraps StarCluster.
3. Generates a new cluster configuration.
4. Creates a new data volume.
5. Creates a new cluster master, worker instance, and storage.
6. Configures the resources to process compute jobs using Sun Grid Engine.

You can use these templates to create a stack, log into the controller image, and use the cluster master to submit jobs or configure your cluster. Download the templates from AWS:

- Download the template for a cc2.8xlarge cluster from https://s3.amazonaws.com/cloudformation-templates-us-east-1/cc2-cluster.json
- Download the template for a cc1.4xlarge cluster from https://s3.amazonaws.com/cloudformation-templates-us-east-1/cc1-cluster.json

For more information about using AWS CloudFormation, see the AWS CloudFormation User Guide.

Cluster Instance Hardware Specifications

Hardware specifications for the cc2.8xlarge cluster compute instance type are described in the following table.

Processor	88 EC2 compute units (2 x Intel Xeon CPU with 8 cores Compute Units)
Memory	60.5 GB
Platform	64-bit
Network	10 Gbps Ethernet
Local Storage	3370 GB instance 64-bit storage (4 x 840 GB plus 10 GB root partition)

Hardware specifications for the cc1.4xlarge cluster compute instance type and cg1.4xlarge cluster GPU instance type are described in the following table.

Processor	33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core "Nehalem" architecture)
Memory	23 GB Note: The cluster GPU instances can use up to 22 GB, with 1 GB reserved for GPU operation. The 22 GB doesn't include the GPU on-board memory, which is 3 GB per GPU for the NVIDIA Tesla M2050 GPU.
Platform	64-bit

Network	10 Gbps Ethernet
Local Storage	1690 GB instance 64-bit storage (2 x 840 GB, plus 10 GB root partition)
GPUs (cluster GPU instances only)	Two NVIDIA Tesla M2050 "Fermi" GPUs Note: The instance must have the corresponding NVIDIA driver installed; the reference AMI we provide includes the driver. If you need to download and install the driver, see Manual Installation of the NVIDIA Driver (p. 344).

When determining the number of cluster instances you can launch, be aware that you're limited to a maximum of eight cluster GPU instances (there's no limit on cluster compute instances) and 20 total EC2 instances, by default. If you need more instances, go to the "Request to Increase Amazon EC2 Instance Limit" form at aws.amazon.com/contact-us/ec2-request.

When planning the number of instances to use in a cluster based on the number of processor cores needed for your application, note that cluster instances use multi-core processors with hyper-threading enabled. This means they have either eight physical cores with 16 threads (cc1.4xlarge), or 16 physical cores with 32 threads (cc2.8xlarge) available to the instance.

Important

We recommend you launch all the instances you'll need in the cluster placement group at once in a single launch request. If you launch only a few and then try adding more to the placement group later, you increase your chances of getting an "Insufficient Capacity" error.

If you want to launch both cluster compute and cluster GPU instances in a placement group, launch the cluster GPU instances first.

Creating an HVM AMI from a Running Instance

To launch cluster instances in Amazon EC2, you must use a Linux/UNIX Amazon EBS-backed HVM AMI. These AMIs have been designed specifically for cluster computing.

You can create an HVM AMI by customizing a running instance of another HVM AMI, and then creating an image from that instance. We have provided the Amazon Linux HVM-based AMI `ami-321eed5b` for you to start your customization.

AWS Management Console

To create an HVM AMI from a running instance

1. For instructions that use the AWS Management Console, see Creating an Image from a Running Instance (p. 185).
2. Use the reference Amazon Linux AMI ami-321eed5b.

Command Line Tools

To create an HVM AMI from a running instance

1. Use `ec2-create-image` to create your own image from your customized instance.

```
PROMPT>  ec2-create-image -n your_image_name  instance_id
```

For example:

```
PROMPT>  ec2-create-image -n "My_Custom_HVM_AMI" i-1a2b3c4d
```

Amazon EC2 begins creating the image and returns the new image's ID. For example:

```
IMAGE ami-5f4d3a2b
```

2. To check whether the AMI is ready, use the following command.

```
PROMPT>  ec2-describe-images -o self
```

Amazon EC2 returns information about the AMI.

API

To create an HVM AMI from a running instance

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateImage
&InstanceId=i-1a2b3c4d
&Name=My_Custom_HVM_AMI
&AUTHPARAMS...
```

Sample response:

```
<CreateImageResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <imageId>ami-5f4d3a2b</imageId>
</CreateImageResponse>
```

Creating a Cluster Placement Group

A *placement group* is a logical grouping of instances. You first create a cluster placement group and then launch multiple cluster instances into the group. Currently cluster compute instances are available only in the US-East (Northern Virginia) Region. You must give each placement group a name that is unique within your account. For more information about cluster placement groups, see Cluster Instance Concepts (p. 335).

Note

You can't merge cluster placement groups. Instead you must terminate the instances in one of the groups, and then relaunch the instances into the other group.

AWS Management Console

To create a cluster placement group

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Placement Groups** in the **Navigation** pane.

   The console displays the list of placement groups in your account.
3. Click **Create Placement Group**.

   The **Placement Group** dialog box appears.
4. Provide a unique name for the placement group and click **Yes, Create**.

   Amazon EC2 creates the placement group.

Command Line Tools

To create a cluster placement group

1. Use the `ec2-create-placement-group` command. You must name the cluster placement group and set the `-s` option to `cluster`.

   ```
   PROMPT> ec2-create-placement-group XYZ-cluster -s cluster
   ```

   Sample response:

   ```
   PLACEMENTGROUP  XYZ_cluster  cluster  available
   ```

2. Record the name of the placement group.
   You launch your cluster instances into this group.
3. Use the `ec2-describe-placement-groups` command to get details about your cluster placement groups.

   If you do not specify the name of the cluster placement group, details about all the placement groups in your account are returned.

   ```
   PROMPT> ec2-describe-placement-groups
   ```

   Sample response:

   ```
   PLACEMENTGROUP  XYZ-cluster  cluster  available
   PLACEMENTGROUP  ABC-cluster  cluster  available
   ```

API

To create a cluster placement group

1. Construct the following Query request. You must name the cluster placement group and specify the strategy as `cluster`.

```
Action=CreatePlacementGroup
&GroupName=XYZ-cluster
&Strategy=cluster
&AUTHPARAMS...
```

Sample response:

```
<CreatePlacementGroupResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-
01/">
   <requestID>d4904fd9-82c2-4ea5-adfe-a9cc3EXAMPLE</requestId>
   <return>true</return>
</CreatePlacementGroupResponse>
```

2. To get information about your cluster placement groups, construct a Query request similar to the following.

```
Action=DescribePlacementGroups
&GroupName.1=XYZ-cluster
&AUTHPARAMS...
```

Sample response:

```
<DescribePlacementGroupsResponse xmlns="http://ec2.amazonaws.com/doc/2011-
12-01/">
  <requestId>d4904fd9-82c2-4ea5-adfe-a9cc3EXAMPLE</requestId>
     <placementGroupSet>
        <item>
           <groupName>XYZ-cluster</groupName>
           <strategy>cluster</strategy>
           <state>available</state>
        </item>
     </placementGroupSet>
</DescribePlacementGroupsResponse>
```

Launching an Instance in a Cluster Placement Group

Before you launch an instance in a cluster placement group, ensure that you have the following:

- A valid HVM AMI
- An existing placement group
- An estimate of the total number of instances you want to launch in the placement group

Important

We recommend you launch all the instances you'll need in the placement group at once in a single launch request. If you launch only a few and then try adding to the placement group later, you increase your chances of getting an "Insufficient Capacity" error.

If you want to launch both cluster compute and cluster GPU instances in a placement group, launch the cluster GPU instances first.

For instructions to create an HVM AMI, see Cluster Instance Concepts (p. 335). For instructions to create a cluster placement group, see Creating a Cluster Placement Group (p. 339).

For information about cluster placement groups, see Cluster Instance Concepts (p. 335).

AWS Management Console

Launching a cluster instance with the AWS Management Console is just like launching any other type of instance. The only difference is that the launch wizard prompts you on whether you want to launch your instances into a placement group.

Command Line Tools

To launch instances in a cluster placement group

- Use the `ec2-run-instances` command with the name of the placement group.

  The following example launches 10 cc1.4xlarge instances into the XYZ-cluster placement group.

  ```
  PROMPT> ec2-run-instances ami-7ea24a17 --type cc1.4xlarge -z us-east-1a -n
   10 --placement-group XYZ-cluster
  ```

  Sample response:

  ```
  RESERVATION     r-064bcd6d      999988887777    default
  INSTANCE i-89fea78c ... cc1.4xlarge ... us-east-1a ... XYZ-cluster   hvm
  ...
  ```

API

To launch instances in a placement group

- Construct a Query request similar to the following. This example launches 10 cc1.4xlarge instances in the XYZ-cluster placement group:

  ```
  Action=RunInstances
  &InstanceType=cc1.4xlarge
  &Placement.GroupName=XYZ-cluster
  &MinCount=10
  &MaxCount=10
  &AUTHPARAMS...
  ```

  Sample response:

  ```
  <RunInstancesResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
     ...
     <placement>
        <availabiltyZone>us-east-1a</availabilityZone>
        <groupName>XYZ-cluster</groupName>
     </placement>
     ...
  ```

```
   <virtualizationType>hvm</virtualizationType>
</RunInstancesResponse>
```

Deleting a Cluster Placement Group

You might want to delete a cluster placement group if your cluster instance requirements change or if you no longer need the placement group.

For information about cluster placement groups, see Cluster Instance Concepts (p. 335).

AWS Management Console

To delete a cluster placement group

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Terminate all running instances in the cluster placement group.
3. Click **Placement Groups** in the **Navigation** pane.
   The console displays a list of cluster placement groups that belong to the account.
4. Select the cluster placement group and click **Delete**.
   A confirmation dialog box appears.
5. Click **Yes, Delete**.
   Amazon EC2 deletes the cluster placement group.

Command Line Tools

To delete a cluster placement group

1. Terminate all running instances in the cluster placement group.
2. Use the `ec2-delete-placement-group` command.

   ```
   PROMPT> ec2-delete-placement-group XYZ-cluster
   ```

   Sample response:

   ```
   PLACEMENTGROUP  XYZ-cluster  deleted
   ```

API

To delete a cluster placement group

1. Terminate all running instances in the cluster placement group.
2. Construct a Query request similar to the following.

```
Action=DeletePlacementGroup
&GroupName=XYZ-cluster
&AUTHPARAMS...
```

Sample response:

```
<DeletePlacementGroupResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-
01/">
   <requestID>d4904fd9-82c2-4ea5-adfe-a9cc3EXAMPLE</requestId>
   <return>true</return>
</DeletePlacementGroupResponse>
```

Using the NVIDIA Driver on a Cluster GPU Instance

The reference AMI we provide (ami-321eed5b) already has the driver installed. Amazon provides updated and compatible builds of the NVIDIA kernel drivers for each new official kernel upgrade. If you decide to use a different NVIDIA driver version than the one Amazon provides or, if you want to use a different kernel than an official Amazon Linux AMI build, then you will first need to uninstall the Amazon-provided NVIDIA packages from your system to avoid conflicts with the versions of the drivers you are trying to install.

Use this command to uninstall Amazon-provided NVIDIA packages:

```
[root@ip-xxx ~]# sudo yum erase nvidia cudatoolkit
```

Important

The Amazon-provided CUDA toolkit package has dependencies on the NVIDIA drivers. Uninstalling the NVIDIA packages erases the CUDA toolkit. You will need to reinstall the CUDA toolkit after installing the NVIDIA driver.

Manual Installation of the NVIDIA Driver

A cluster GPU instance must have the NVIDIA Tesla M2050 GPU driver. The NVIDIA driver you install needs to be compiled against the kernel you intend to run on your instance.

Tip

You can download NVIDIA drivers at http://www.nvidia.com/Download/Find.aspx. Look for the Tesla M-Class M2050 driver for Linux 64-bit systems.

To install the driver

1. Make sure the kernel-devel package is installed and matches with the version of the kernel you are currently running.

   ```
   [root@ip-xxx ~]# yum  install kernel-devel- uname –r
   ```

2. Install the NVIDIA driver.

```
[root@ip-xxx ~]# /root/NVIDIA-Linux-x86_64_260.19.12.run
```

3. Reboot the instance.
4. Confirm that the driver is functional.

   ```
   [root@ip-xxx ~]# /usr/bin/nvidia-smi -q -a
   ```

   The response lists the installed NVIDIA driver version and details about the GPUs.
5. Confirm that the GPUs are exposed.

   ```
   [root@ip-xxx ~]# ls /dev/*nv*
   /dev/nvidia0 /dev/nvidia1 /dev/nvidiactl /dev/nvram
   ```

If a newer version of the NVIDIA driver is available, you might want to update your instance to use it. To update your instance, follow the steps in the preceding section for installing the driver and run the self-install script from the newly downloaded driver version.

To install the CUDA toolkit package, download the package from the NVIDIA website and run the self-install script as outlined in step 2 of the preceding section for installing the driver.

Auto-Scaling and Load Balancing Your Instances

If your expect your application to have significant variability in usage, you might want to use Auto Scaling and Elastic Load Balancing, two features of Amazon EC2 that help manage the variability.

Auto Scaling

Auto Scaling enables you to scale up or down the number of instances you are using based on parameters that you specify, such as traffic or CPU load.

Auto Scaling also monitors the health of each Amazon EC2 instance that it launches. If any instance terminates unexpectedly, Auto Scaling detects the termination and launches a replacement instance.

For a high degree of flexibility, you can organize Amazon EC2 instances into AutoScalingGroups, which enable you to scale different server classes (e.g., web servers, back end servers) at different rates. For each group, you specify the minimum number of instances, the maximum number of instances, and the parameters to increase and decrease the number of running instances.

For information on setting up Auto Scaling, go to the Amazon Auto Scaling Developer Guide.

Load Balancing

Elastic Load Balancing lets you automatically distribute the incoming traffic (or load) among all the instances you are running. The service also makes it easy to add new instances when you need to increase the capacity of your web site application.

Customers reach your web site via your web URL, such as www.mywebsite.com. This single address might actually represent several instances of your running web application. To always have an available web site, you need to run multiple instances. Otherwise, your customers might see delays when accessing your site, or worse, might not be able to access your site at all.

Elastic Load Balancing manages the incoming requests by optimally routing traffic so that no one instance is overwhelmed. You can quickly add more instances to applications that are experiencing an upsurge in traffic or remove capacity when traffic is slow.

For information on setting up Elastic Load Balancing, go to the Elastic Load Balancing Developer Guide.

Monitoring Your Instances and Volumes

Topics

Amazon Web Services (AWS) automatically provides data, such as Amazon CloudWatch metrics and instance status, that you can use to monitor your Amazon EC2 instances and Amazon Elastic Block Store (Amazon EBS) volumes:

- CloudWatch metrics are statistical data you can use to view, analyze, and set alarms on the operational behavior of your instances and volumes. These metrics include CPU utilization, network traffic, I/O, and latency.
- Instance status provides information about specific events that are scheduled for your instances, including activities that AWS may perform such as rebooting and retirement. You can view scheduled events for any of your instances in the AWS Management Console or retrieve them using the command line or API.

Monitoring Your Instances and Volumes with CloudWatch

Topics

Amazon CloudWatch is a service that collects raw data from partnered AWS products such as Amazon EC2 and then processes the information into readable, near real-time metrics. These statistics are recorded for a period of two weeks, allowing you access to historical information and providing you with a better perspective on how your web application or service is performing. For detailed information about Amazon CloudWatch, go to the Amazon CloudWatch Developer Guide.

The following table describes the types of Amazon EC2 monitoring data available to you.

Resource	Type	Description
Instances	Basic	Data is available automatically in 5-minute periods at no charge.
	Detailed	Data is available in 1-minute periods at an additional cost. To get this level of data, you must specifically enable it for the instance. For the instances where you've enabled detailed monitoring, you can also get aggregated data across groups of similar instances. For information about pricing, go to the Amazon CloudWatch product page.
Volumes	Basic	Data is available automatically in 5-minute periods at no charge. No detailed data is available for Amazon EBS volumes.

Monitoring Instances

You can get monitoring data for your Amazon EC2 instances using either the Amazon CloudWatch API or the AWS Management Console. The console displays a series of graphs based on the raw data from the Amazon CloudWatch API. Depending on your needs, you might prefer to use either the data from the API or the graphs in the console.

Data from the Amazon Cloudwatch API

You can use the Amazon CloudWatch `GetMetricStatistics` API to get any of the instance metrics listed in the following table. The *period* refers to how often the system reports a data point for each metric for an instance. If you've enabled detailed monitoring, each data point covers the instance's previous 1 minute of activity. Otherwise, each data point covers the instance's previous 5 minutes of activity.

Metric	Description
`CPUUtilization`	The percentage of allocated EC2 compute units that are currently in use on the instance. This metric identifies the processing power required to run an application upon a selected instance. Units: *Percent*

Metric	Description
`DiskReadOps`	Completed read operations from all ephemeral disks available to the instance (if your instance uses Amazon EBS, see Amazon EBS Metrics (p. 354).) This metric identifies the rate at which an application reads a disk. This can be used to determine the speed in which an application reads data from a hard disk. Units: *Count*
`DiskWriteOps`	Completed write operations to all ephemeral disks available to the instance (if your instance uses Amazon EBS, see Amazon EBS Metrics (p. 354).) This metric identifies the rate at which an application writes to a hard disk. This can be used to determine the speed in which an application saves data to a hard disk. Units: *Count*
`DiskReadBytes`	Bytes read from all ephemeral disks available to the instance (if your instance uses Amazon EBS, see Amazon EBS Metrics (p. 354).) This metric is used to determine the volume of the data the application reads from the hard disk of the instance. This can be used to determine the speed of the application. Units: *Bytes*
`DiskWriteBytes`	Bytes written to all ephemeral disks available to the instance (if your instance uses Amazon EBS, see Amazon EBS Metrics (p. 354).) This metric is used to determine the volume of the data the application writes onto the hard disk of the instance. This can be used to determine the speed of the application. Units: *Bytes*
`NetworkIn`	The number of bytes received on all network interfaces by the instance. This metric identifies the volume of incoming network traffic to an application on a single instance. Units: *Bytes*
`NetworkOut`	The number of bytes sent out on all network interfaces by the instance. This metric identifies the volume of outgoing network traffic to an application on a single instance. Units: *Bytes*

Note

When you get data from the Amazon CloudWatch system, you can specify a *`Period`* request parameter (i.e., how granular you want the returned data to be). This is different than the period we use when we collect the data (either 1-minute periods for detailed monitoring, or 5-minute periods for basic monitoring). We recommend you specify a period in your request that is equal to or larger than the collection period to ensure the returned data is valid.

You can use the dimensions in the following table to refine the metrics returned for your instances.

Dimension	Description
`AutoScalingGroupName`	This dimension filters the data you request for all instances in a specified capacity group. An *AutoScalingGroup* is a collection of instances you define if you're using the Auto Scaling service. This dimension is available only for EC2 metrics when the instances are in such an AutoScalingGroup. Available for instances with Detailed or Basic Monitoring enabled.
`ImageId`	This dimension filters the data you request for all instances running this EC2 Amazon Machine Image (AMI). Available for instances with Detailed Monitoring enabled.
`InstanceId`	This dimension filters the data you request for the identified instance only. This helps you pinpoint an exact instance from which to monitor data. Available for instances with Detailed Monitoring enabled.
`InstanceType`	This dimension filters the data you request for all instances running with this specified instance type. This helps you categorize your data by the type of instance running. For example, you might compare data from an m1.small instance and an m1.large instance to determine which has the better business value for your application. Available for instances with Detailed Monitoring enabled.

For more information about using the `GetMetricStatistics` action, go to GetMetricStatistics in the *Amazon CloudWatch API Reference*.

Graphs in the AWS Management Console

After you launch an instance, you can go to the AWS Management Console and view the instance's monitoring graphs. They're displayed when you select the instance on the **Instances** page in the EC2 Dashboard. A **Monitoring** tab is displayed next to the instance's **Description** tab. The following graphs are available:

- Average CPU Utilization (Percent)
- Average Disk Reads (Bytes)
- Average Disk Writes (Bytes)
- Maximum Network In (Bytes)
- Maximum Network Out (Bytes)

The AWS Management Console contains a console for Amazon CloudWatch. In the Amazon CloudWatch console you can search and browse all your AWS resource metrics, view graphs to troubleshoot issues and discover trends, create and edit alarms to be notified of problems, and see at-a-glance overviews of your alarms and AWS resources. For more information, go to AWS Management Console in the *Amazon CloudWatch Developer Guide*.

Enabling Detailed Monitoring on an Amazon EC2 Instance

This section describes how to enable detailed monitoring on either a new instance (as you launch it) or on a running or stopped instance. After you enable detailed monitoring, the Amazon EC2 console in the AWS Management Console displays monitoring graphs with a 1-minute period for the instance.

AWS Management Console

To enable detailed monitoring when launching an instance

- When launching an instance with the AWS Management Console, select the check box for **Enable detailed CloudWatch Monitoring for this instance** on the **Advanced Instance Options** section of the launch wizard.

  After the instance is launched, you can select the instance in the console and view its monitoring graphs on the instance's **Monitoring** tab in the lower pane.

For information about launching instances, see Launching and Using Instances (p. 75).

To enable detailed monitoring on an existing instance

- In your list of instances in the AWS Management Console, right-click the instance and click **Enable Detailed Monitoring**. The instance can be either running or stopped.

  Detailed data (collected with a 1-minute period) is then available for the instance in the AWS Management Console graphs or through the API.

Command Line Tools

To enable detailed monitoring when launching an instance

- Use the `ec2-run-instances` command with the `--monitor` flag.

  ```
  PROMPT>  ec2-run-instances ami-2bb65342 -k gsg-keypair --monitor
  ```

  Amazon EC2 returns output similar to the following example. The status of monitoring is listed as *monitoring-enabled*.

  ```
  RESERVATION     r-7430c31d      999988887777    default
  INSTANCE        i-ae0bf0c7      ami-2bb65342    pending gsg-keypair   0
   m1.small   2008-03-21T16:19:25+0000       us-east-1a   monitoring-enabled
  ```

  After the instance is running, detailed data (collected with a 1-minute period) is then available for the instance in the AWS Management Console graphs or through the API.

To enable detailed monitoring on an existing instance

- Use the `ec2-monitor-instances` command with one or more instance IDs.

  ```
  PROMPT>  ec2-monitor-instances i-1a2b3c4d
  i-1a2b3c4d    monitoring-pending
  ```

  Detailed data (collected with a 1-minute period) is then available for the instance in the AWS Management Console graphs or through the API.

Create a CloudWatch Alarm

You can create an Amazon CloudWatch alarm that monitors any one of your Amazon EC2 instance’s CloudWatch metrics. CloudWatch will automatically send you a notification when the metric reaches a threshold you specify. You can create a CloudWatch alarm on the Amazon EC2 tab of the AWS Management Console or you can use the CloudWatch tab and configure more advanced options.

To create a CloudWatch alarm for high CPU

1. In the AWS Management Console, on the Amazon EC2 tab, select the instance for which you want to create an alarm.
2. On the instance’s **Monitoring** tab in the lower pane, click **Create Alarm**.
3. In the **Create Alarm** dialog box, set the criteria for your alarm. In this example, we’ll set an alarm if the instance’s average CPU utilization is above 70 percent.
4. The check box next to Send a notification is selected by default. Click **Create topic**, and either use the default name, or select an existing name. (Notifications use Amazon Simple Notification Service (Amazon SNS)).
5. In the **With these recipients** box, enter the email addresses of the recipients you want to notify. You can enter up to 10 email addresses, each separated by a comma.
6. Configure the threshold for your alarm.

   a. In the **Whenever** boxes, select **Average** and **CPU Utilization**.

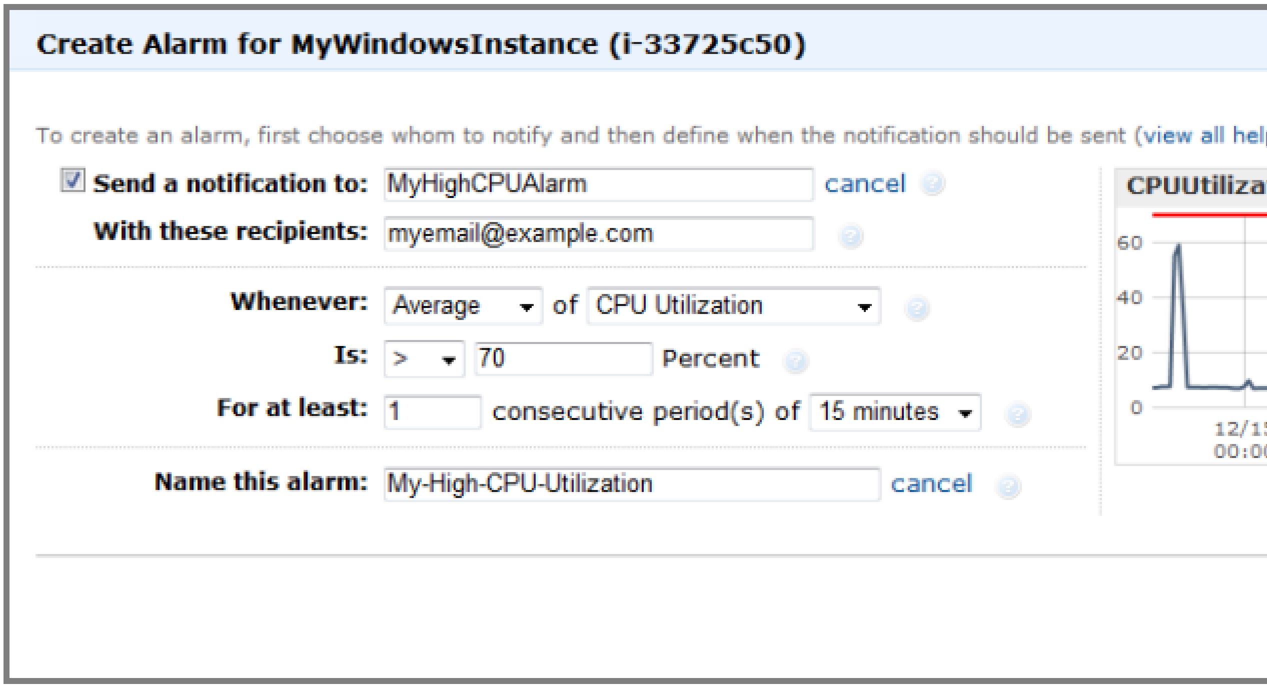
   b. In the **Is** boxes, define the threshold for the alarm by selecting **>** and entering **70**.

   c. In the **For at least** boxes, specify the sampling period and number of samples evaluated by the alarm. You can leave the defaults or define your own. For our example, we’ll monitor for 1 period of 15 minutes.

   > **Note**
   >
   > A shorter period creates a more sensitive alarm. A longer period can mitigate brief spikes in a metric.

   d. In **Alarm name**, a name is automatically generated for you. Click **Edit** if you want to change the name.

   > **Important**
   >
   > You cannot modify the name after you create the alarm.



Create Alarm for MyWindowsInstance (i-33725c50)

To create an alarm, first choose whom to notify and then define when the notification should be sent (view all hel

☑ Send a notification to: MyHighCPUAlarm cancel

With these recipients: myemail@example.com

Whenever: Average of CPU Utilization

Is: > 70 Percent

For at least: 1 consecutive period(s) of 15 minutes

Name this alarm: My-High-CPU-Utilization cancel

7. Click **Create Alarm**.

After you create the alarm, you can use the **Monitoring** tab in the Amazon EC2 console to view a summary of alarms that have been set for that instance. From there, you can also edit the alarm.

Note

If you created a new Amazon SNS topic for this alarm or added new email addresses to an existing topic, each email address added will receive a subscription confirmation email from Amazon SNS. The person who receives the email must confirm it by clicking the included link in order to receive notifications.

Monitoring Amazon EBS Volumes

The following table describes the types of monitoring data available for your Amazon EBS volumes.

Type	Description
Basic	Data is available automatically in 5-minute periods at no charge. This includes data for the root device volumes for Amazon EBS-backed instances. No detailed data is available for Amazon EBS volumes.

Note

When you get data from Amazon CloudWatch, you can specify a `Period` request parameter (i.e., how granular you want the returned data to be). This is different than the period we use when we collect the data (5-minute periods). We recommend you specify a period in your request that is equal to or larger than the collection period to ensure the returned data is valid.

You can get the data using either the Amazon CloudWatch API or the AWS Management Console. The console takes the raw data from the Amazon CloudWatch API and displays a series of graphs based on the data. Depending on your needs, you might prefer to use either the data from the API or the graphs in the console.

Amazon EBS Metrics

You can use the Amazon CloudWatch `GetMetricStatistics` API to get any of the Amazon EBS volume metrics listed in the following table. The period for all the metrics is 5 minutes, which means the system reports one data point every 5 minutes for each metric for each volume, and that data point covers the volume's previous 5 minutes of activity.

The following table groups metrics that are similar. The metrics in the first two rows are also available for the local stores on Amazon EC2 instances.

Metric	Description
`VolumeReadBytes` `VolumeWriteBytes`	The total number of bytes transferred in the period. Units: Bytes
`VolumeReadOps` `VolumeWriteOps`	The total number of operations in the period. Units: Count
`VolumeTotalReadTime` `VolumeTotalWriteTime`	The total number of seconds spent by all operations that completed in the period. If multiple requests are submitted at the same time, this total could be greater than the length of the period. For example, say the period is 5 minutes (300 seconds); if 700 operations completed during that period, and each operation took 1 second, the value would be 700 seconds. Units: Seconds
`VolumeIdleTime`	The total number of seconds in the period when no read or write operations were submitted. Units: Seconds
`VolumeQueueLength`	The number of read and write operation requests waiting to be completed in the period. Units: Count

Graphs in the AWS Management Console

Once you create a volume, you can go to the AWS Management Console and view the volume's monitoring graphs. They're displayed when you select the volume on the **Volumes** page in the EC2 Dashboard. A **Monitoring** tab is displayed next to the volume's **Description** tab. The following table lists the graphs that are displayed. The column on the right describes how the raw data metrics from the Amazon CloudWatch API are used to produce each graph. The period for all the graphs is 5 minutes.

Graph Name	Description Using Raw Metrics
Read Bandwidth (KiB/s)	Sum(VolumeReadBytes) / Period / 1024
Write Bandwidth (KiB/s)	Sum(VolumeWriteBytes) / Period / 1024
Read Throughput (Ops/s)	Sum(VolumeReadOps) / Period

Graph Name	Description Using Raw Metrics
Write Throughput (Ops/s)	Sum(VolumeWriteOps) / Period
Avg Queue Length (ops)	Avg(VolumeQueueLength)
% Time Spent Idle	Sum(VolumeIdleTime) / Period * 100
Avg Read Size (KiB/op)	Avg(VolumeReadBytes) / 1024
Avg Write Size (KiB/op)	Avg(VolumeWriteBytes) / 1024
Avg Read Latency (ms/op)	Avg(VolumeTotalReadTime) * 1000
Avg Write Latency (ms/op)	Avg(VolumeTotalWriteTime) * 1000

For the average latency graphs and average size graphs, the average is calculated over the total number of operations (read or write, whichever is applicable to the graph) that completed during the period.

The AWS Management Console contains a console for Amazon CloudWatch. In the Amazon CloudWatch console you can search and browse all your AWS resource metrics, view graphs to troubleshoot issues and discover trends, create and edit alarms to be notified of problems, and see at-a-glance overviews of your alarms and AWS resources. For more information, go to AWS Management Console in the *Amazon CloudWatch Developer Guide*.

Monitoring the Status of Your Instances

Instance status describes specific events that may be scheduled for your instances. Scheduled events provide information about upcoming activities such as rebooting or retirement that are planned for your instances, along with the scheduled start and end time of each event. These scheduled events are not frequent.

There are three types of scheduled events:

- **Instance reboot**: AWS may schedule an instance for a reboot for necessary maintenance such as to apply patch upgrades to underlying host that *do not* require the host to be rebooted.
- **System reboot**: AWS may schedule an instance for a reboot for necessary maintenance such as to apply patch upgrades that *do* require the host to be rebooted.
- **Instance retirement**: AWS may schedule instances for retirement in cases where there is an unrecoverable issue with the hardware on an underlying host.

⚠ **Important**

No action is required on your part if one of your instances is scheduled for reboot. We recommend that you wait for the reboot to occur within its scheduled maintenance window.

For instances scheduled for retirement, we recommend that you take the actions detailed later in this section.

Monitoring Scheduled Events with Instance Status

You can view scheduled events for your instances using the AWS Management Console, the API, or the command line interface.

AWS Management Console

To view scheduled events for your instances

1. Log in to the AWS Management Console.
2. Click the **Amazon EC2** tab.
3. In the **Navigation** pane, click **Scheduled Events**. You see a list of instances that have upcoming events scheduled.
4. Alternatively, you can do the following to view upcoming scheduled events:

   a. In the **Navigation** pane, click the **EC2 Dashboard**.

   b. Under Scheduled Events, you can see the number of upcoming scheduled events for your Amazon EC2 instances, as well as a countdown to the next scheduled event.

Command Line

To view scheduled events for your instances

- Enter the following command:

```
PROMPT> ec2-describe-instance-status
```

Amazon EC2 returns output similar to the following:

```
Type    InstanceId      AvailabilityZone        InstanceStateName       In
stanceStateCode
INSTANCE        i-6d9eaa0c      us-east-1d      running 16
Type    EventCode       NotBefore       NotAfter        EventDescription
EVENT   instance-reboot 2011-12-05T13:00:00+0000        2011-12-
06T13:00:00+0000        The instance is scheduled for a reboot
INSTANCE        i-ee7e508e      us-east-1c      running 16
Type    EventCode       NotBefore       NotAfter        EventDescription
EVENT   instance-reboot 2011-12-06T13:00:00+0000        2011-12-
07T13:00:00+0000        The instance is scheduled for a reboot
INSTANCE        i-1c654d7c      us-east-1a      running 16
Type    EventCode       NotBefore       NotAfter        EventDescription
EVENT   instance-reboot 2011-12-07T13:00:00+0000        2011-12-
08T13:00:00+0000        The instance is scheduled for a reboot
INSTANCE        i-893eb0ea      us-east-1d      running 16
Type    EventCode       NotBefore       NotAfter        EventDescription
EVENT   instance-reboot 2011-12-07T13:00:00+0000        2011-12-
08T13:00:00+0000        The instance is scheduled for a reboot
```

For more information about using the `ec2-describe-instance-status` command, see ec2-describe-instance-status in the *Amazon Elastic Compute Cloud Command Reference User's Guide.*

API

You can use the `DescribeInstanceStatus` action to retrieve the status of your instances. For more information, see DescribeInstanceStatus in the *Amazon Elastic Compute Cloud API Reference Guide*.

Working with an Instance that Has a Scheduled Event

This section discusses tasks you can perform if your instance has one of the following scheduled events.

- Reboot: an instance or a system reboot
- Instance retirement

Recommended Tasks for Instances Scheduled for a Reboot

AWS may schedule instances for a reboot to perform maitenance tasks such as patch upgrades to the software on an underlying host. These scheduled reboots are not frequent. There are two types of reboot events: system reboot and instance reboot. In either case, your instance will be rebooted. During a system reboot, the hardware supporting your instance will also be rebooted. During an instance reboot, your instance will be rebooted but the hardware supporting the instance will not be rebooted.

Important

No action is required on your part if one of your instances is scheduled for reboot. We recommend that you wait for the reboot to occur automatically within its scheduled maintenance window.

Scheduled reboot events start within their scheduled maintenance window. Once initiated, both system and instance reboots typically complete in a matter of minutes. After a reboot completes, your instance is available to use; it is not necessary to wait until the scheduled end time.

To verify that the reboot has occurred, check your scheduled events and verify that the instance no longer shows a scheduled event. We recommend that you check your instance after it is rebooted to ensure that your application is functioning as you expected.

Optional Alternative Tasks for Instances Scheduled for Reboot

We recommend that you wait for the reboot to occur automatically within its scheduled window. If you choose, you may perform the instance reboot yourself to control the timing of the event.

Instance Reboot

For instance reboot events, you can perform the reboot yourself (be careful to not shutdown or terminate your instance) at any time before the scheduled reboot event begins. The reboot can be initiated using the AWS Management Console, a `RebootInstances` API call, or from within the instance (e.g., at the command prompt). After you reboot, any pending maintenance to the underlying host is performed automatically, and you can begin using your instance again after the instance has fully booted.

Note

After you perform the reboot, the scheduled event for the instance reboot is canceled immediately. The event's description is updated in the AWS console to reflect this.

System Reboot

If you choose to perform a system reboot, your course of action will differ depending on whether your instance’s root device is an EBS volume or the instance store. You can determine the root device type for an instance using either the DescribeInstances API or the AWS Management Console. In the console, you select an instance and view the root device type listed in the **Description** tab.

Instances Using an EBS Volume as the Root Device Type

If your instance's root device is an EBS volume, you can stop and re-start it (be careful not to shutdown or terminate your instance). If you choose to do so, some configuration settings will changes.

If you stop and restart your instance, the following changes occur. If you wait for Amazon EC2 to perform the scheduled reboot automatically, these configuration settings remain the same.

- Data in the instance store (e.g. data not stored on any attached EBS volume including the root device) will no longer be available. Before you stop the instance, backup any data you may need.
- The public DNS name and the private IP address of the instance will change.
- If you associated an Elastic IP address with this instance, stopping this instance also disassociates the Elastic IP address from it. Once you start the instance, re-associate it with the Elastic IP address if the address is still required.

  Note

  When the Elastic IP address is not in use, you will incur charges.

- For instances in Amazon Virtual Private Cloud (Amazon VPC), the Elastic IP address and the private IP address remain unchanged.

Before you stop and restart an instance, perform the following tasks:

1. Retrieve any data from the instance store (e.g., data that is not stored on any attached EBS volumes including the root device) that you will need later. This data will not be available after you stop and restart an instance.
2. Take a snapshot of your existing volume (storage charges will apply).
3. Note the necessary configuration data in case you will need it later, including the DNS public name and the private IP address.
4. Stop and restart your instance. For more information, see Stopping and Starting Instances (p. 89).
5. Re-associate an Elastic IP address if this address is necessary.
6. If other applications or instances rely on the public DNS name or the private IP address of this instance, update the information with the new configuration data.

Note

After you stop and re-start the instance, the scheduled event for the system reboot is canceled immediately. The event's description is updated in the AWS console to reflect this.

Instances Using the Instance Store as the Root Device Type

If your instance's root device is the instance store and you cannot wait for AWS to reboot your instance in the scheduled maintenance window, you can opt to launch a replacement instance. After you have launched the replacement instance, you can terminate the original instance. If you choose to take this action, be aware that:

- The public DNS name and the private IP address of your replacement instance will differ from your original instance.
- You must first ensure that you have an AMI with your desired configuration that you can use when launching the new instance. For more information, see:

- Bundling Amazon EC2 instance store-backed Windows AMIs (p. 21)
- Bundling Amazon EC2 instance store-backed Linux/UNIX AMIs (p. 25)

Instances Scheduled for Retirement

AWS may schedule instances for retirement in cases where there is an unrecoverable issue with the hardware on an underlying host. If you have an instance scheduled for retirement, we recommend that you perform the following tasks based on your instance's root device type, before its scheduled retirement time.

Instances Using an EBS Volume as the Root Device Type

If your instance’s root device is an EBS volume, you can stop and re-start it (be careful not to shutdown or terminate the instance). If you choose to do so, some configuration settings will change. The following changes occur:

- Data in the instance store (e.g. data not stored on any attached EBS volume including the root device) will no longer be available. Before you stop the instance, backup any data you may need.
- The public DNS name and the private IP address of the instance will change.
- If you associated an Elastic IP address with this instance, stopping this instance also disassociates the Elastic IP address from it. Once you start the instance, re-associate it with the Elastic IP address if the address is still required.

  Note

  When the Elastic IP address is not in use, you will incur charges.

- For instances in Amazon Virtual Private Cloud (Amazon VPC), the Elastic IP address and the private IP address remain unchanged.

Before you stop and restart an instance, perform the following tasks:

1. Retrieve any data from the instance store (e.g., data that is not stored on any attached EBS volumes including the root device) that you will need later. This data will not be available after you stop and restart an instance.
2. Take a snapshot of your existing volume (storage charges will apply).
3. Note the necessary configuration data in case you will need it later, including the DNS public name and the private IP address.
4. Stop and restart your instance. For more information, see Stopping and Starting Instances (p. 89).
5. Re-associate an Elastic IP address if this address is necessary.
6. If other applications or instances rely on the public DNS name or the private IP address of this instance, update the information with the new configuration data.

Note

After you stop and re-start the instance, the scheduled event is canceled immediately. The event's description is updated in the AWS console to reflect this.

Instances Using the Instance Store as the Root Device Type

If your instance's root device is the instance store and your instance is scheduled for retirement, you can opt to launch a replacement instance. After you have launched a replacement instance, you can terminate the original instance. If you choose to take this action, be aware that:

- The public DNS name and the private IP address of your replacement instance will differ from your original instance.
- You must first ensure that you have an AMI with your desired configuration to use when launching the new instance. For more information, see:
  - Bundling Amazon EC2 instance store-backed Windows AMIs (p. 21)
  - Bundling Amazon EC2 instance store-backed Linux/UNIX AMIs (p. 25)

Using Public Data Sets

Topics

This section describes how to use Amazon EC2 public data sets.

Public Data Set Concepts

Amazon EC2 provides a repository of public data sets that can be seamlessly integrated into AWS cloud-based applications. Amazon stores the data sets at no charge to the community and, like with all AWS services, you pay only for the compute and storage you use for your own applications.

Previously, large data sets such as the mapping of the Human Genome and the US Census data required hours or days to locate, download, customize, and analyze. Now, anyone can access these data sets from their Amazon EC2 instances and start computing on the data within minutes. You can also leverage the entire AWS ecosystem and easily collaborate with other AWS users. For example, you can produce or use prebuilt server images with tools and applications to analyze the data sets. By hosting this important and useful data with cost-efficient services such as Amazon EC2, AWS hopes to provide researchers across a variety of disciplines and industries with tools to enable more innovation, more quickly.

For more information, go to the Public Data Sets Page.

Available Public Data Sets

Public data sets are currently available in the following categories:

- **Biology—**Includes Human Genome Project, GenBank, and other content.
- **Chemistry—**Includes multiple versions of PubChem and other content.
- **Economics—**Includes census data, labor statistics, transportation statistics, and other content.
- **Encyclopedic—**Includes Wikipedia content from multiple sources and other content.

Finding Public Data Sets

Before you launch a public data set, you must locate the set to launch.

To find a public data set

1. Go to the Public Data Sets Page.
2. Locate a public data set and write down its snapshot ID for your operating platform (e.g., Windows, Linux/UNIX).

Launching an Instance

Launch an instance as you normally do. For more information, see Launching and Using Instances (p. 75).

Launching a Public Data Set Volume

To use a public data set, you launch an Amazon EBS volume, specifying its snapshot ID.

AWS Management Console

To create an Amazon EBS volume

1. Log in to the AWS Management Console and click the **Amazon EC2** tab.
2. Click **Volumes** in the **Navigation** pane.
   The console displays a list of current volumes.
3. Click **Create Volume**.
   The **Create Volume** dialog box appears.
4. Configure the following settings and click **Create**.

   - Size of the volume (in GiB) (optional)
   - Availability Zone in which to launch the instance
   - The ID of the public data set snapshot

   Amazon EC2 begins creating the volume.

Command Line Tools

To create an Amazon EBS volume

1. Enter the following command.

   ```
   PROMPT>  ec2-create-volume --snapshot public-data-set-snapshot-id --zone
   availability-zone
   ```

   Amazon EBS returns information about the volume similar to the following example.

   ```
   VOLUME vol-4d826724 85 us-east-1a creating 2008-02-14T00:00:00+0000
   ```

2. To check whether the volume is ready, use the following command.

   ```
   PROMPT>  ec2-describe-volumes vol-4d826724
   ```

   Amazon EBS returns information about the volume similar to the following example.

   ```
   VOLUME  vol-4d826724 85 us-east-1a  available 2008-07-29T08:49:25+0000
   ```

API

To create an Amazon EBS volume

- Construct the following Query request.

```
https://ec2.amazonaws.com/
?Action=CreateVolume
&AvailabilityZone=zone
&SnapshotId=public-data-set-snapshot-id
&...auth parameters...
```

Following is an example response.

```
<CreateVolumeResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <volumeId>vol-4d826724</volumeId>
  <size>85</size>
  <status>creating</status>
  <createTime>2008-05-07T11:51:50.000Z</createTime>
  <availabilityZone>us-east-1a</availabilityZone>
  <snapshotId>snap-59d33330</snapshotId>
</CreateVolumeResponse>
```

Mounting the Public Data Set Volume

Mount the volume as you normally do. For more information, see Making an Amazon EBS Volume Available for Use (p. 138).

Amazon Virtual Private Cloud

Amazon Virtual Private Cloud enables you to create a virtual network in the AWS cloud. With a Virtual Private Cloud (VPC), you can define a virtual network that closely resembles a traditional data center. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of routing and access control lists. This section gives a brief introduction to Amazon VPC.

Amazon Virtual Private Cloud Concepts

If you're familiar with Amazon EC2, you know that each instance you launch is randomly assigned a public IP address in the Amazon EC2 address space. Amazon VPC enables you to create an isolated portion of the AWS cloud (a *VPC*) and launch Amazon EC2 instances that have private (RFC 1918) addresses in the range of your choice (e.g., 10.0.0.0/16). You can define *subnets* within your VPC, which enable you to group similar kinds of instances based on IP address range.

By using Amazon VPC with Amazon EC2 (instead of Amazon EC2 alone), you gain the ability to:

- Logically group your Amazon EC2 instances, and assign them private IP addresses
- Control the egress traffic from your Amazon EC2 instances (in addition to controlling the ingress traffic to them)
- Add an additional layer of security to your Amazon EC2 instances in the form of network Access Control Lists (ACLs)
- Connect your VPC to your corporate data center with a VPN connection, so you can use the AWS cloud as an extension of your corporate data center network

Levels of Privacy

When you create a VPC, you can configure it based on the level of privacy you want. In the most private scenario, you can attach only a *virtual private gateway*, and create an IPsec tunnel between your VPC and home network. In this scenario, your EC2 instances have no direct exposure to the Internet.

In the most public scenario, you can attach only an *Internet gateway* to the VPC and enable traffic to flow between the Internet and all the instances in your VPC.

You can configure your VPC to be somewhere in between, with both a virtual private gateway and an Internet gateway. Here, some instances could receive Internet traffic (e.g., web servers), whereas others could remain unexposed (e.g., database servers). This is a common scenario for running a multi-tier web application in the AWS cloud.

These different scenarios are discussed in more detail in the Amazon VPC documentation.

Routing and Security

You can configure routing in your VPC to control where traffic flows (e.g., to the Internet gateway, virtual private gateway, etc). With an Internet gateway, your VPC has direct access to other AWS products such as Amazon Simple Storage Service (Amazon S3). If you choose to have only a virtual private gateway with a connection to your home network, you can route your Internet-bound traffic over the VPN and control its egress with your security policies and corporate firewall. In the latter case, you incur additional bandwidth charges when accessing AWS products over the Internet.

You can use *security groups* and *network ACLs* to help secure the instances in your VPC. Security groups might be familiar if you're an Amazon EC2 user, and network ACLs might be familiar if you're a network administrator. Security groups act like a firewall at the instance level, whereas network ACLs are an additional layer of security that act at the subnet level.

By default, the instances you launch in your VPC have only private IP addresses. If you want an instance to have a public IP address, you can assign it an *Elastic IP address*, which is a static, public address you can assign to any instance in your VPC. For an instance in your VPC to be addressable from the Internet, it must have an Elastic IP address.

You can use Network Address Translation (NAT) to enable instances that don't have Elastic IP addresses to reach the Internet. You can set up the VPC's routing so that traffic from private instances goes through a special NAT instance that has an Elastic IP address. We provide a NAT Amazon Machine Image (AMI) that you can use for this purpose.

Dedicated Instances

Amazon EC2 instances launched into a VPC have a tenancy attribute. Setting the instance's tenancy attribute to `dedicated` specifies that your instance will run on single-tenant hardware. Amazon VPCs have a related attribute called *instance tenancy*. Setting this instance tenancy attribute to `dedicated` specifies that only Dedicated Instances can be launched into the VPC.

For more information, go to Using EC2 Dedicated Instances Within Your VPC in the *Amazon Virtual Private Cloud User Guide*.

Amazon VPC Documentation

Amazon VPC has its own set of documentation to describe how to create and use your VPC. The following table gives links to the Amazon VPC guides.

Description	Documentation
How to get started using Amazon VPC	Amazon Virtual Private Cloud Getting Started Guide
How to use Amazon VPC through the AWS Management Console	Amazon Virtual Private Cloud User Guide
Complete descriptions of all the Amazon VPC commands	Amazon Elastic Compute Cloud Command Line Reference (the Amazon VPC commands are part of the Amazon EC2 reference)
Complete descriptions of the Amazon VPC API actions, data types, and errors	Amazon Elastic Compute Cloud API Reference (the Amazon VPC API actions are part of the Amazon EC2 reference)
Information for the network administrator who needs to configure the gateway at your end of an optional IPsec VPN connection	Amazon Virtual Private Cloud Network Administrator Guide

Elastic Network Interfaces

Topics

Each EC2 instance has a default network interface that is assigned a private IP address on your Amazon VPC network. You can create and attach an additional network interface, known as an elastic network interface (ENI), to any Amazon EC2 instance in your VPC for a total of two network interfaces per instance.

ENIs have several attributes including, a private IP address, an elastic IP address (optional), a MAC address, membership in specified security groups, a description, and a source/destination check flag. You can create an elastic network interface, attach it to an instance, detach it from an instance, and attach it to another instance. An ENI's attributes, including the private IP address, elastic IP addresses, and MAC address, will follow the ENI as it is attached or detached from an instance and reattached to another instance.

You can attach an ENI to an instance during the launch process (cold attach), when an instance is stopped (warm attach), and when an instance is running (hot attach).

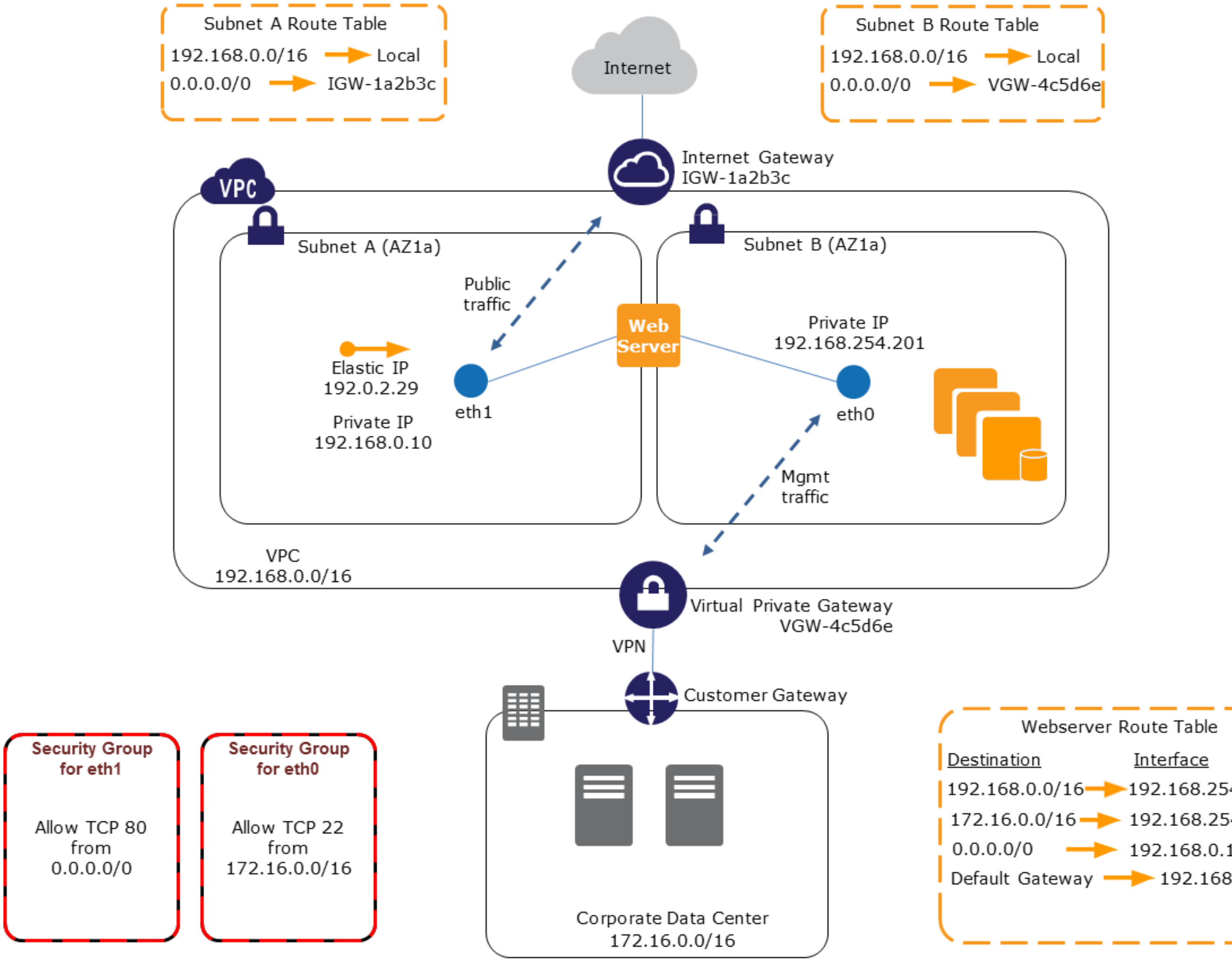
Attaching more than one network interface to an instance is useful when you want to:

- Create a management network.
- Use network and security appliances in your VPC.
- Create dual-homed instances with workloads/roles on distinct subnets.
- Create a low budget high availability solution.

Creating a Management Network

You can create a management network by utilizing ENIs. In a management network scenario the secondary network interface on the instance handles public-facing traffic and the primary network interface handles back-end management traffic and is connected to a separate subnet in your VPC that has more restrictive access controls. The public facing interface, which may or may not be behind a load balancer, will have an associated security group allowing access to the server from the Internet (e.g.; allow TCP port 80 and

443 from 0.0.0.0/0, or from the loadbalancer) while the private facing interface will have an associated security group allowing SSH access only from an allowed range of IP addresses either within the VPC or from the Internet, a private subnet within the VPC or a virtual private gateway.



Use Network and Security Appliances in Your VPC

Some third party network and security appliances such as load balancers, network address translation (NAT) servers, and proxy servers prefer to be configured with a second network interface. You can create and attach secondary network interfaces to instances in a VPC that are running these types of applications and configure the second interface with its own public and private IP addresses, security groups, and source/destination checking.

Creating Dual-homed Instances with Workloads/Roles on Distinct Subnets

You can place an ENI on each of your web servers that connects to a mid-tier network where an application server resides. The application server can also be dual-homed to a backend network (subnet) where the database server resides. Instead of routing network packets through the dual-homed instances, each dual-homed instance receives and processes requests on the front end, initiates a connection to the backend, and then sends requests to the servers on the backend network.

Create a Low Budget High Availability Solution

If one of your instances serving a particular function fails, its network interface can be attached to a replacement or hot standby instance pre-configured for the same role in order to rapidly recover the service. For example, you can use an ENI as your primary or secondary network interface to a critical service such as a database instance or a NAT instance. If the instance fails, you (or more likely, the code running on your behalf) can attach the ENI to a hot standby instance. Because the interface maintains its private IP address, elastic IP address, and MAC address, network traffic will begin flowing to the standby instance as soon as you attach the ENI to the replacement instance. Users will experience a brief loss of connectivity between the time the instance fails and the time that the ENI is attached to the standby instance, but no changes to the VPC route table or your DNS server are required.

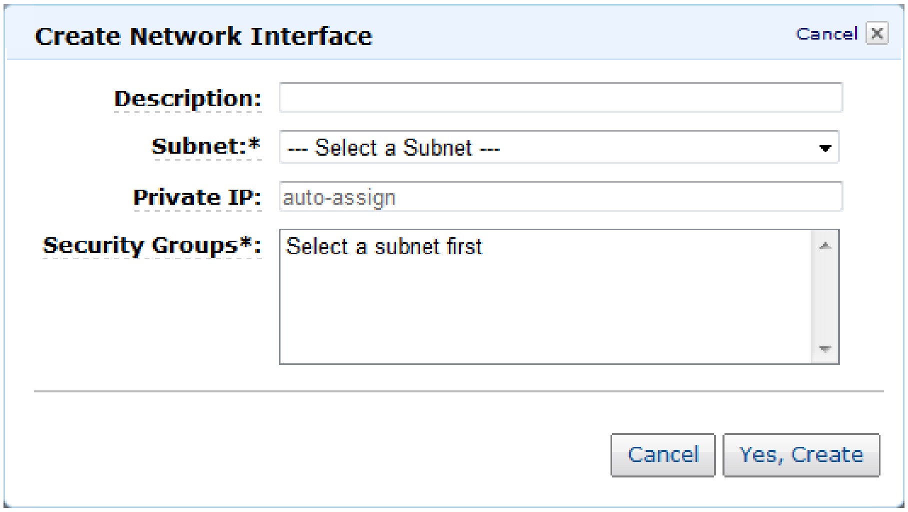
Best Practices for Configuring Network Interfaces

- You can attach an ENI to an instance when it's running (hot attach), when it's stopped (warm attach), or when the instance is being launched (cold attach).
- You can detach secondary (eth1) network interfaces when the instance is running or stopped. However, you cannot detach the primary (eth0) interface.
- You can attach an ENI in one subnet to an instance in another subnet in the same VPC, however, both the network interface and the instance must reside in the same Availability Zone.
- When launching an instance from the CLI or API you can specify the ENIs to attach to the instance for both the primary (eth0) and additional network interfaces (you cannot specify eth0 when launching an instance from the AWS Management Console.) Warm attach and hot attach via API, command line tools, and the AWS Management Console enable you to only attach the second network interface (eth1).
- Launching an EC2 instance with multiple network interfaces will automatically configure interfaces, private IP addresses and route tables on the operating system of the instance. Warm or hot attaching an additional network interface (when the instance is stopped or running) may require you to manually bring up the second interface, configure the private IP address, and modify the route table accordingly. Microsoft Windows Server 2003 and 2008 instances will automatically recognize the warm or hot attach and will automatically configure themselves. Instances that are not running a Microsoft Windows Server operating system might require the manual interaction to configure the interface on the operating system.)
- Attaching multiple network interfaces to an instance is intended to assist with the above use cases. Attaching multiple network interfaces to a single instance is not recommended as a method to provide multiple public IP addresses on a single instance. A separate feature to support multiple IP addresses per instance is in development and will be available shortly.
- Attaching a second network interface to an instance is not a method to increase or double the network bandwidth to or from the dual-homed instance.

Create an Elastic Network Interface

To create an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. At the top of the **Network Interfaces** pane, click **Create Network Interface**.
3. In the **Create Network Interface** dialog box, in the **Description** box, type a descriptive name for the network interface.



Description:

Subnet:* --- Select a Subnet ---

Private IP: auto-assign

Security Groups*: Select a subnet first

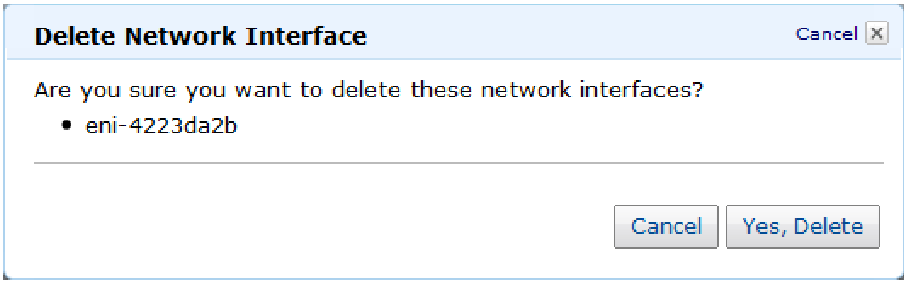
4. In the **Subnet** drop-down list box, select the subnet in which you want to create the network interface. The network interface cannot be moved to another subnet after it is created.
5. In the **Private IP** box, enter the private IP address for the network interface. If you don't specify an IP address, AWS will automatically assign one to the interface from an available IP address within the selected subnet.
6. In the **Security Groups** drop-down list box, select one or more security groups to use with the network interface.
7. Click **Yes, Create**.

Delete an Elastic Network Interface

You must first detach an ENI from an instance before you can delete it. Deleting an ENI releases all attributes associated with the network interface and releases any private IP addresses or elastic IP addresses for use by another instance.

To delete an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. At the top of the **Network Interfaces** pane, select an ENI, and then click **Delete Network Interface**.
3. In the **Delete Network Interface** dialog box, click **Yes, Delete**.



Are you sure you want to delete these network interfaces?

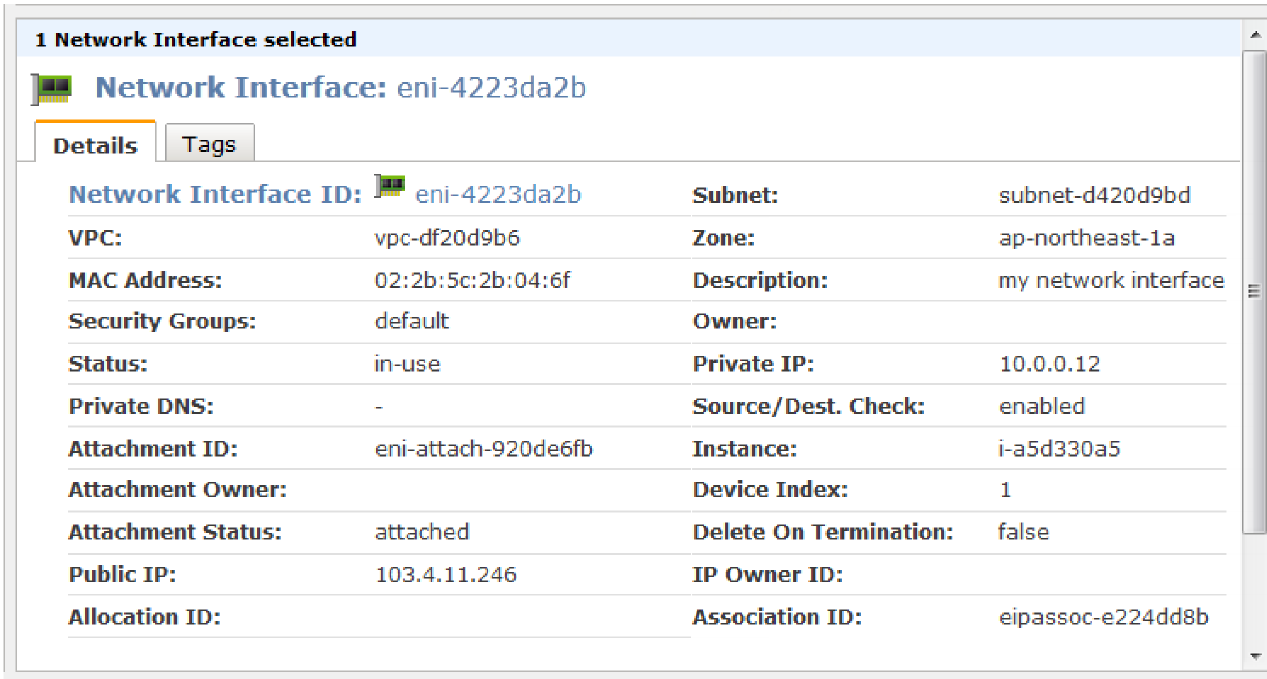
- eni-4223da2b

Viewing Details about an Elastic Network Interface

You can view the attributes of your ENIs.

To view details about an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, select an ENI.
3. In the lower pane, on the **Details** tab, you can view the details about the ENI.



1 Network Interface selected

Network Interface: eni-4223da2b

Network Interface ID:	eni-4223da2b	Subnet:	subnet-d420d9bd
VPC:	vpc-df20d9b6	Zone:	ap-northeast-1a
MAC Address:	02:2b:5c:2b:04:6f	Description:	my network interface
Security Groups:	default	Owner:	
Status:	in-use	Private IP:	10.0.0.12
Private DNS:	-	Source/Dest. Check:	enabled
Attachment ID:	eni-attach-920de6fb	Instance:	i-a5d330a5
Attachment Owner:		Device Index:	1
Attachment Status:	attached	Delete On Termination:	false
Public IP:	103.4.11.246	IP Owner ID:	
Allocation ID:		Association ID:	eipassoc-e224dd8b

Attach an Elastic Network Interface When Launching an Instance

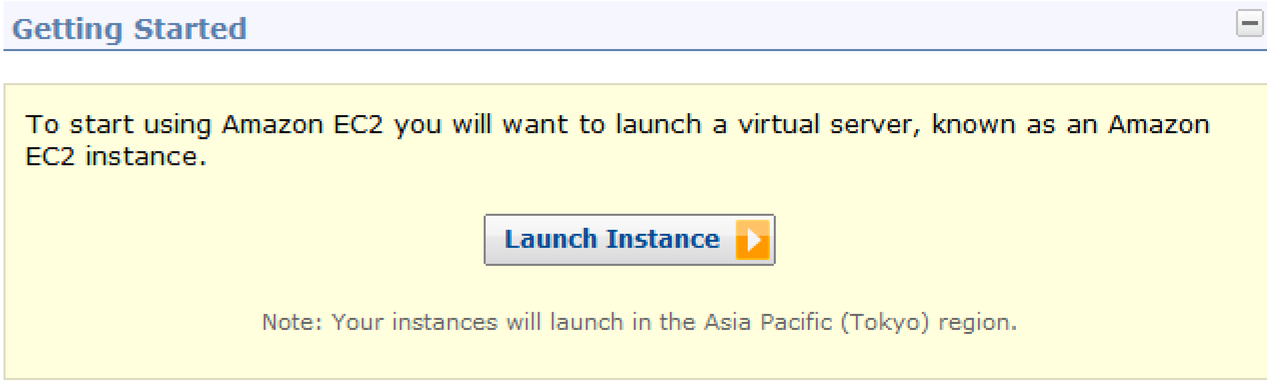
To attach an ENI when launching an instance

You can attach an additional network interface, designated as eth1, to an instance when you launch it into a VPC using the Launch Instances Wizard.

Note

If an error occurs when attaching an ENI to your instance, this will cause the instance launch to fail.

1. In the EC2 console, click **Launch Instance**.

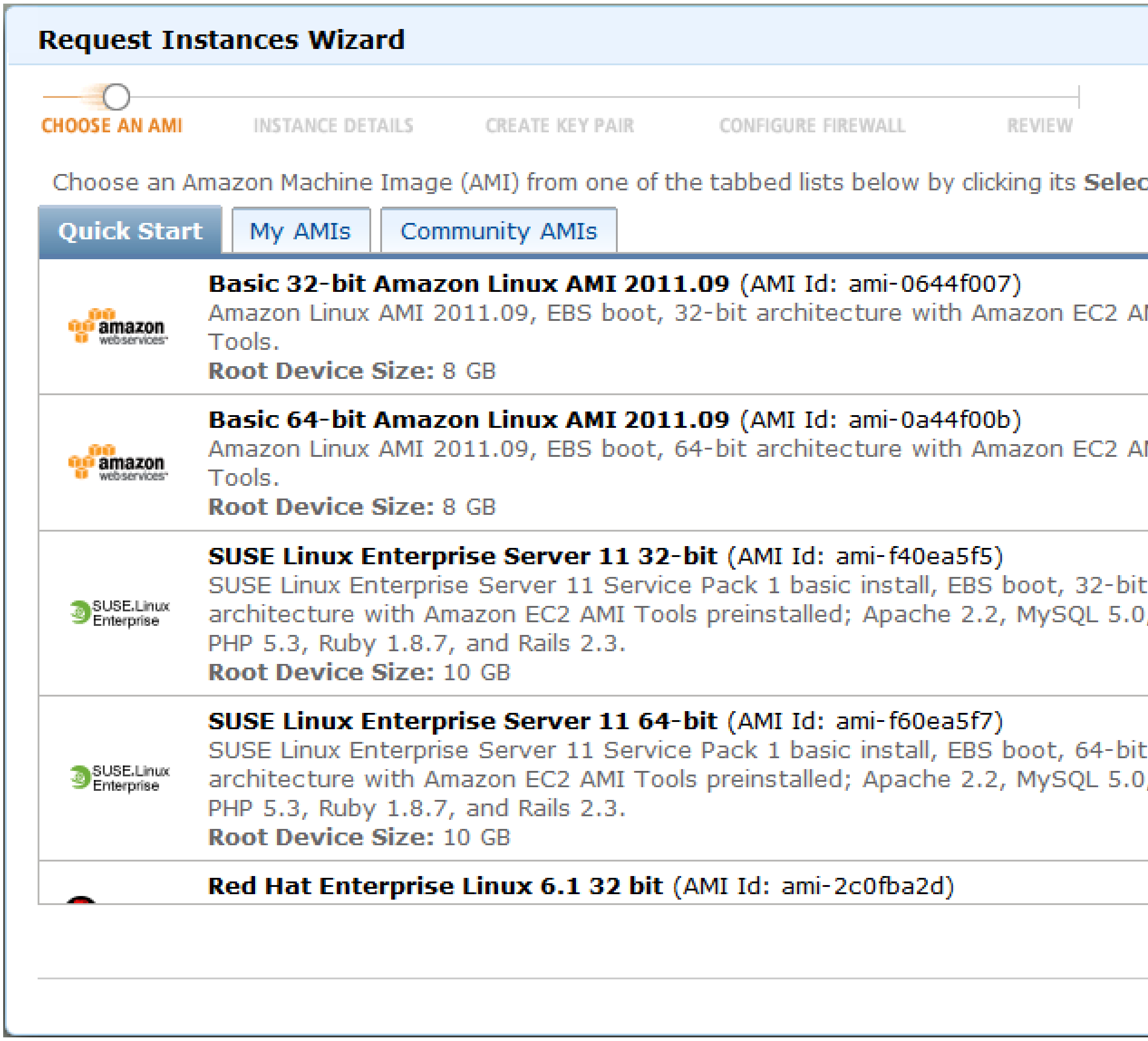


Getting Started

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Note: Your instances will launch in the Asia Pacific (Tokyo) region.

2. In the **Request Instances Wizard**, choose an Amazon Machine Image (AMI).



Request Instances Wizard

Choose an Amazon Machine Image (AMI) from one of the tabbed lists below by clicking its **Selec**

Basic 32-bit Amazon Linux AMI 2011.09 (AMI Id: ami-0644f007)
Amazon Linux AMI 2011.09, EBS boot, 32-bit architecture with Amazon EC2 AI Tools.
Root Device Size: 8 GB

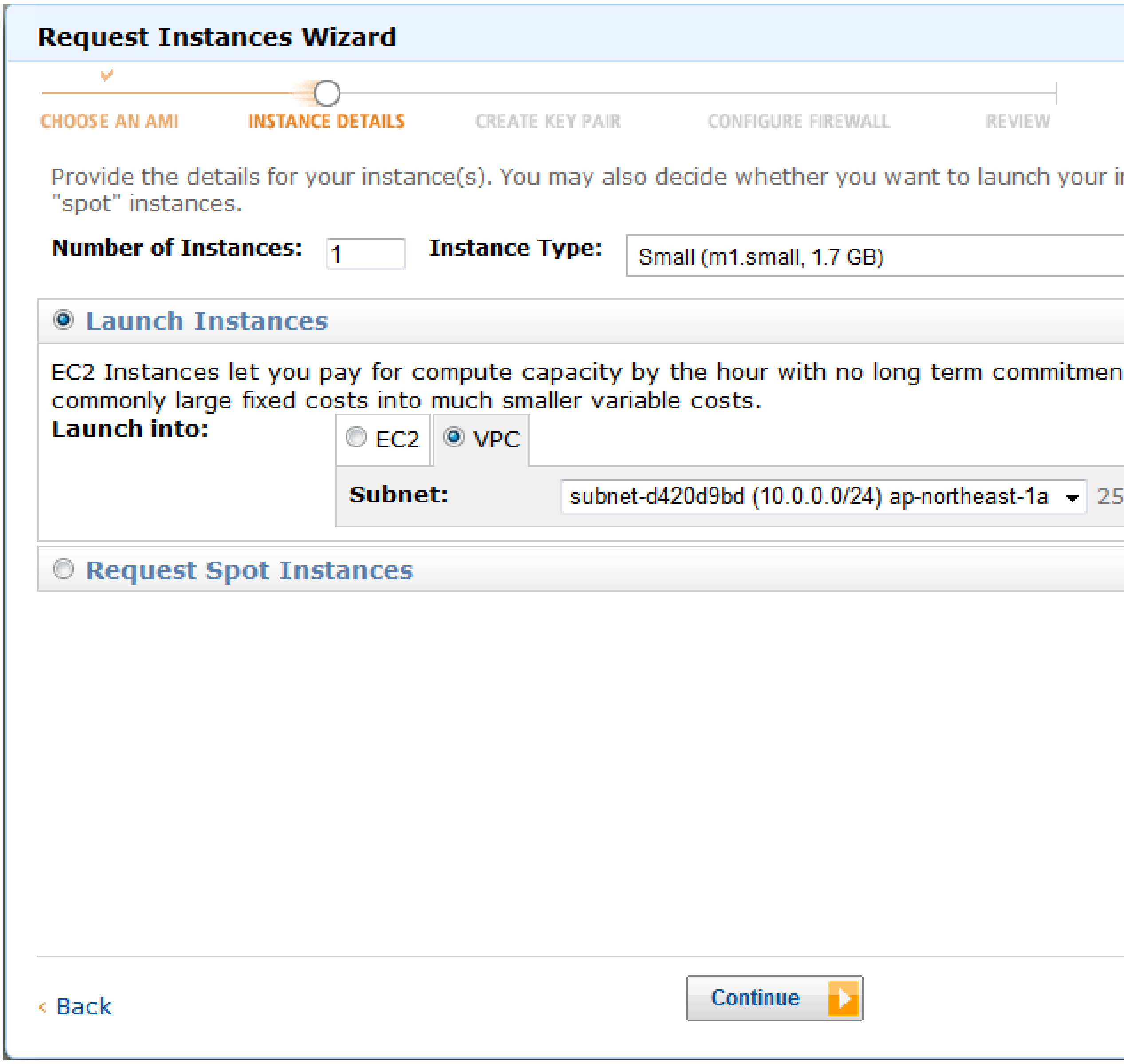
Basic 64-bit Amazon Linux AMI 2011.09 (AMI Id: ami-0a44f00b)
Amazon Linux AMI 2011.09, EBS boot, 64-bit architecture with Amazon EC2 AI Tools.
Root Device Size: 8 GB

SUSE Linux Enterprise Server 11 32-bit (AMI Id: ami-f40ea5f5)
SUSE Linux Enterprise Server 11 Service Pack 1 basic install, EBS boot, 32-bit architecture with Amazon EC2 AMI Tools preinstalled; Apache 2.2, MySQL 5.0, PHP 5.3, Ruby 1.8.7, and Rails 2.3.
Root Device Size: 10 GB

SUSE Linux Enterprise Server 11 64-bit (AMI Id: ami-f60ea5f7)
SUSE Linux Enterprise Server 11 Service Pack 1 basic install, EBS boot, 64-bit architecture with Amazon EC2 AMI Tools preinstalled; Apache 2.2, MySQL 5.0, PHP 5.3, Ruby 1.8.7, and Rails 2.3.
Root Device Size: 10 GB

Red Hat Enterprise Linux 6.1 32 bit (AMI Id: ami-2c0fba2d)

3. On the **INSTANCE DETAILS** page, set the number and size of instance you wan to launch, and then click the **Launch Instances** radio button.



Request Instances Wizard

Provide the details for your instance(s). You may also decide whether you want to launch your i
"spot" instances.

Number of Instances: 1 **Instance Type:** Small (m1.small, 1.7 GB)

● Launch Instances

EC2 Instances let you pay for compute capacity by the hour with no long term commitmen
commonly large fixed costs into much smaller variable costs.

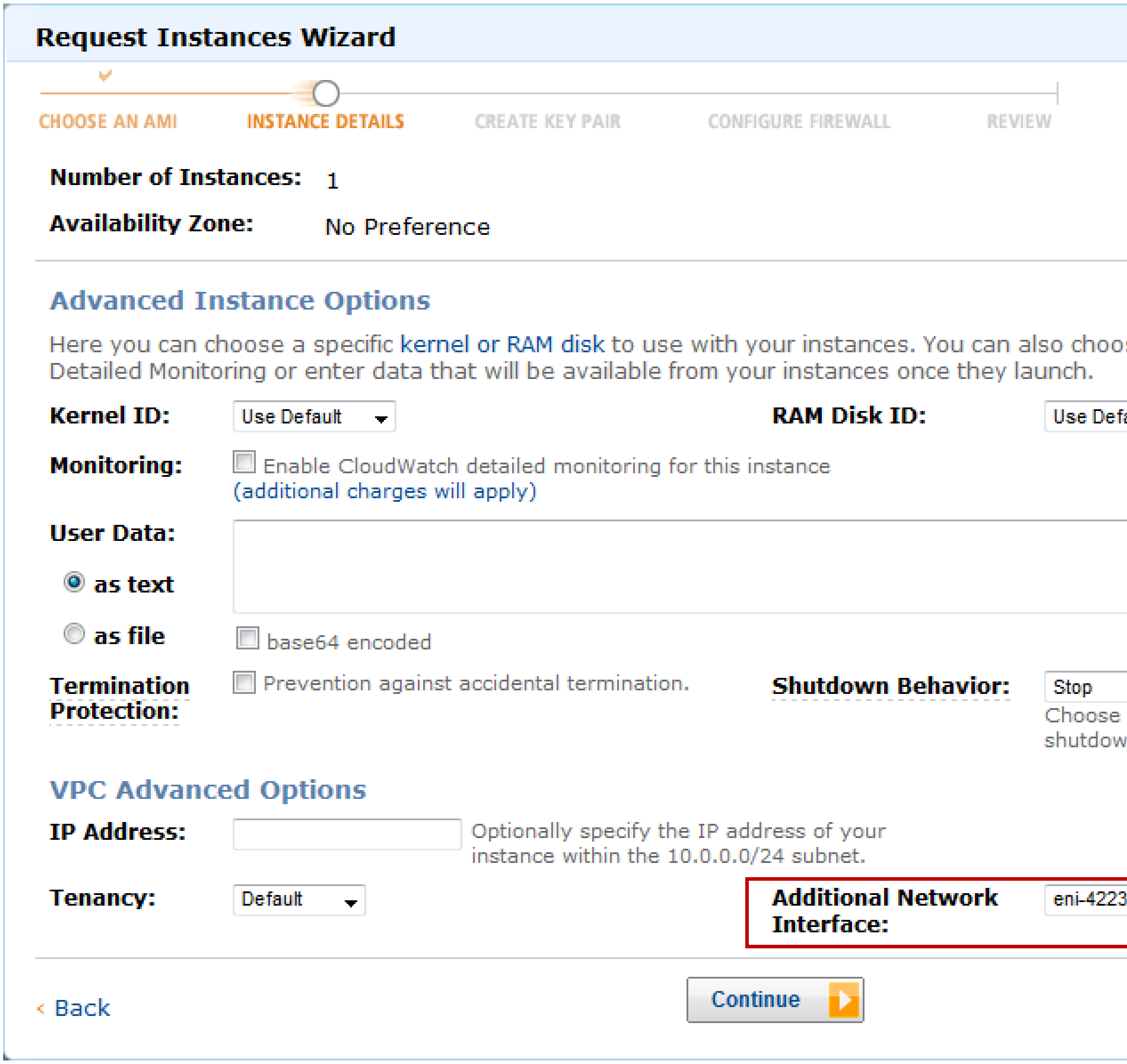
Launch into: ○ EC2 ● VPC

Subnet: subnet-d420d9bd (10.0.0.0/24) ap-northeast-1a 25

○ Request Spot Instances

4. On the **EC2** tab, set the availability zone, and then click the **VPC** tab and confirm that your subnet is selected in the **Subnet** list box.
5. Click **Continue**, and then on the **Advanced Instance Options** page, select an additional network interface from the pop-up menu.

   You can enter an IP address for the primary network interface (eth0) on this page. The private IP address of the additional network interface (eth1) was previously selected when the ENI was created.



Request Instances Wizard

Number of Instances: 1

Availability Zone: No Preference

Advanced Instance Options

Here you can choose a specific kernel or RAM disk to use with your instances. You can also choo
Detailed Monitoring or enter data that will be available from your instances once they launch.

Kernel ID: Use Default

RAM Disk ID: Use Defa

Monitoring: □ Enable CloudWatch detailed monitoring for this instance
(additional charges will apply)

User Data:

● as text

○ as file

□ base64 encoded

Termination Protection: □ Prevention against accidental termination.

Shutdown Behavior: Stop
Choose
shutdow

VPC Advanced Options

IP Address: Optionally specify the IP address of your instance within the 10.0.0.0/24 subnet.

Tenancy: Default

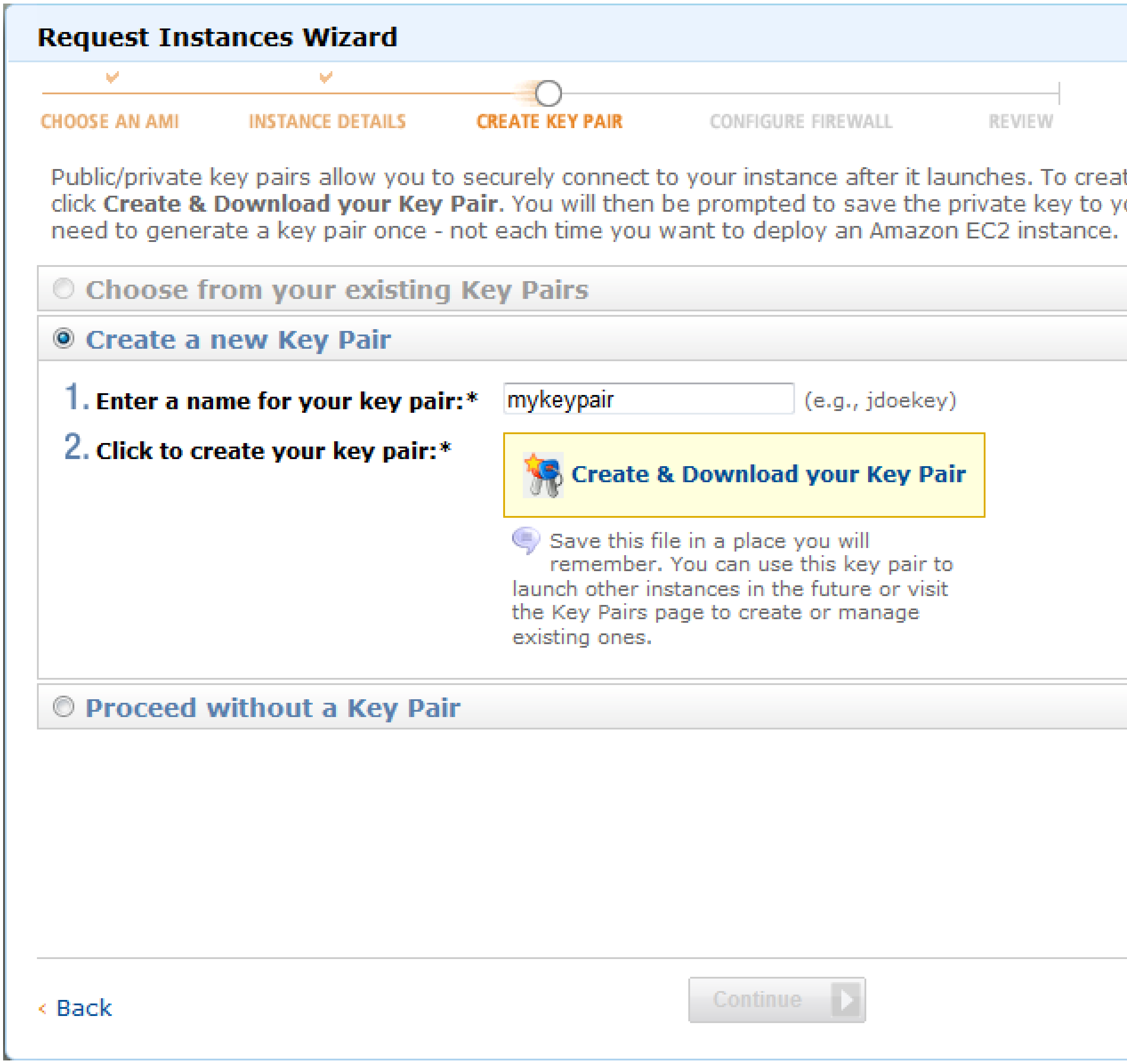
Additional Network Interface: eni-4223

6. Click **Continue**, and then on the **CREATE KEY PAIR** page, select an existing key pair or create a new one. If you create a new key pair, you must download it before you can click **Continue**.

Note

A *key pair* is a security credential similar to a password, which you use to securely connect to your instance once it's running. If you're new to Amazon EC2 and haven't created any key pairs, when the wizard displays the **Create Key Pair** page, the **Create a new Key Pair** button is selected by default. You'll need to create a key pair. For more information, see Getting an SSH Key Pair (p. 77).

When you create a new key pair, EC2 uses the name you specify (e.g., `mykeypair`) to name the private key file (with a `.pem` extension) associated with the key pair.



Request Instances Wizard

Public/private key pairs allow you to securely connect to your instance after it launches. To creat
click **Create & Download your Key Pair**. You will then be prompted to save the private key to y
need to generate a key pair once - not each time you want to deploy an Amazon EC2 instance.

○ Choose from your existing Key Pairs

● Create a new Key Pair

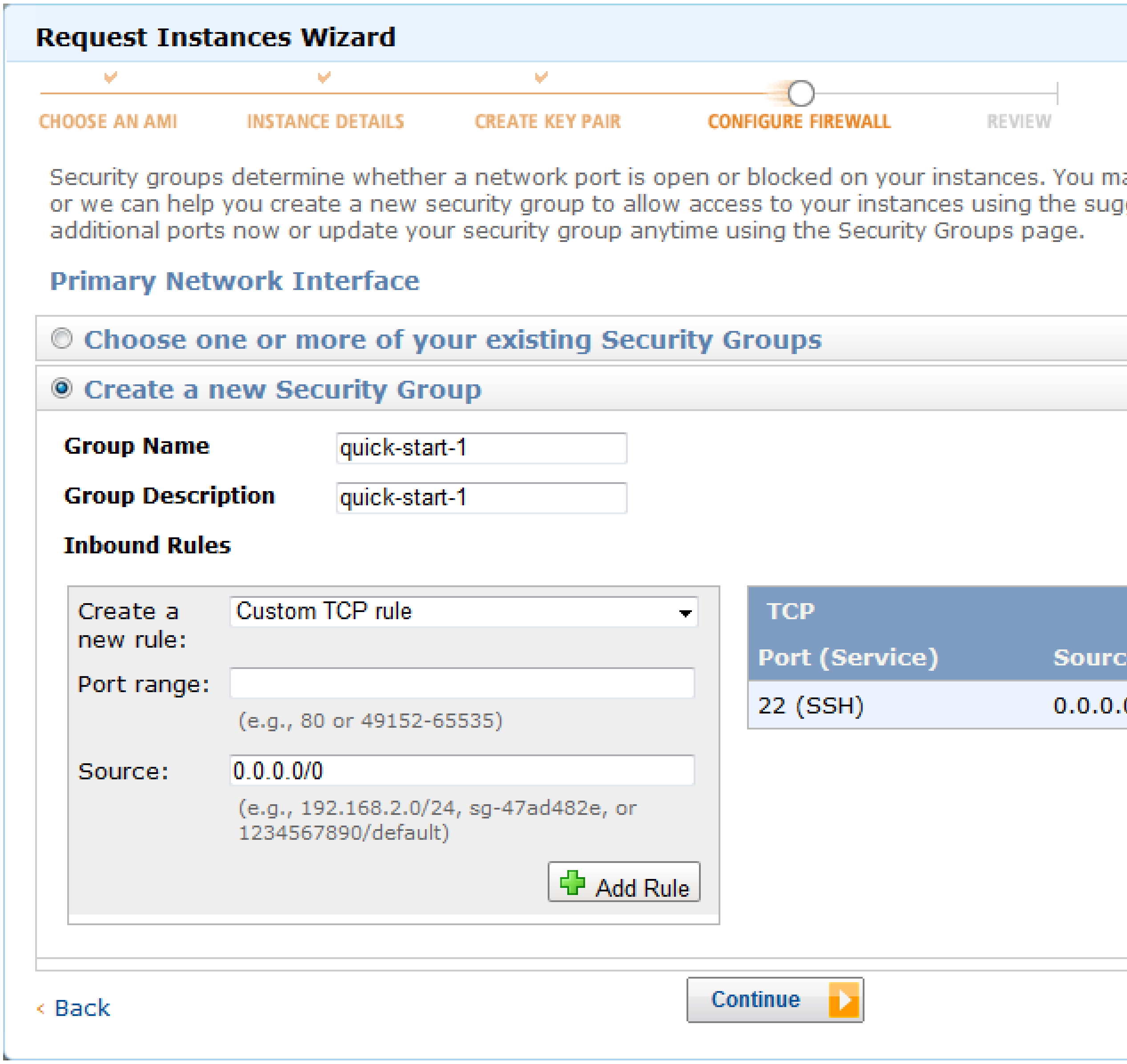
1. Enter a name for your key pair:* mykeypair (e.g., jdoekey)
2. Click to create your key pair:*

Save this file in a place you will remember. You can use this key pair to launch other instances in the future or visit the Key Pairs page to create or manage existing ones.

○ Proceed without a Key Pair

7. On the **CONFIGURE FIREWALL** page, select an existing security group for the primary network interface or create a new one, and then click **Continue**.

   The security group for the additional network interface was previously selected when the ENI was created.



Security groups determine whether a network port is open or blocked on your instances. You ma
or we can help you create a new security group to allow access to your instances using the sug
additional ports now or update your security group anytime using the Security Groups page.

Primary Network Interface

○ Choose one or more of your existing Security Groups

● Create a new Security Group

Group Name: quick-start-1

Group Description: quick-start-1

Inbound Rules

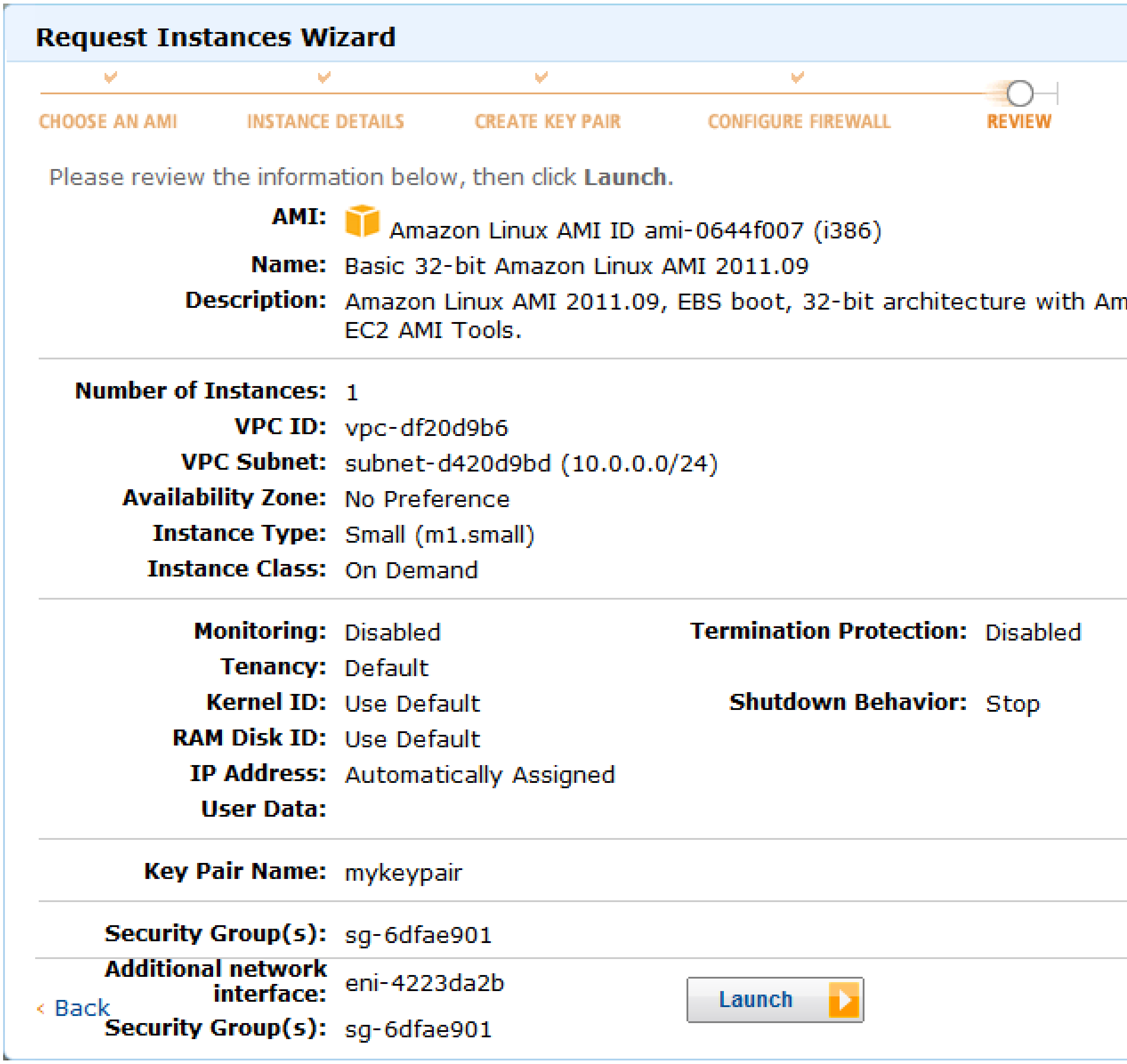
Create a new rule: Custom TCP rule

Port range:
(e.g., 80 or 49152-65535)

Source: 0.0.0.0/0
(e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

TCP	
Port (Service)	Sourc
22 (SSH)	0.0.0.(

8. On the **REVIEW** page, details about the primary and additional network interface are displayed. Review the settings, and then click **Launch**.



Request Instances Wizard

Please review the information below, then click **Launch**.

AMI: Amazon Linux AMI ID ami-0644f007 (i386)
Name: Basic 32-bit Amazon Linux AMI 2011.09
Description: Amazon Linux AMI 2011.09, EBS boot, 32-bit architecture with Am
EC2 AMI Tools.

Number of Instances: 1
VPC ID: vpc-df20d9b6
VPC Subnet: subnet-d420d9bd (10.0.0.0/24)
Availability Zone: No Preference
Instance Type: Small (m1.small)
Instance Class: On Demand

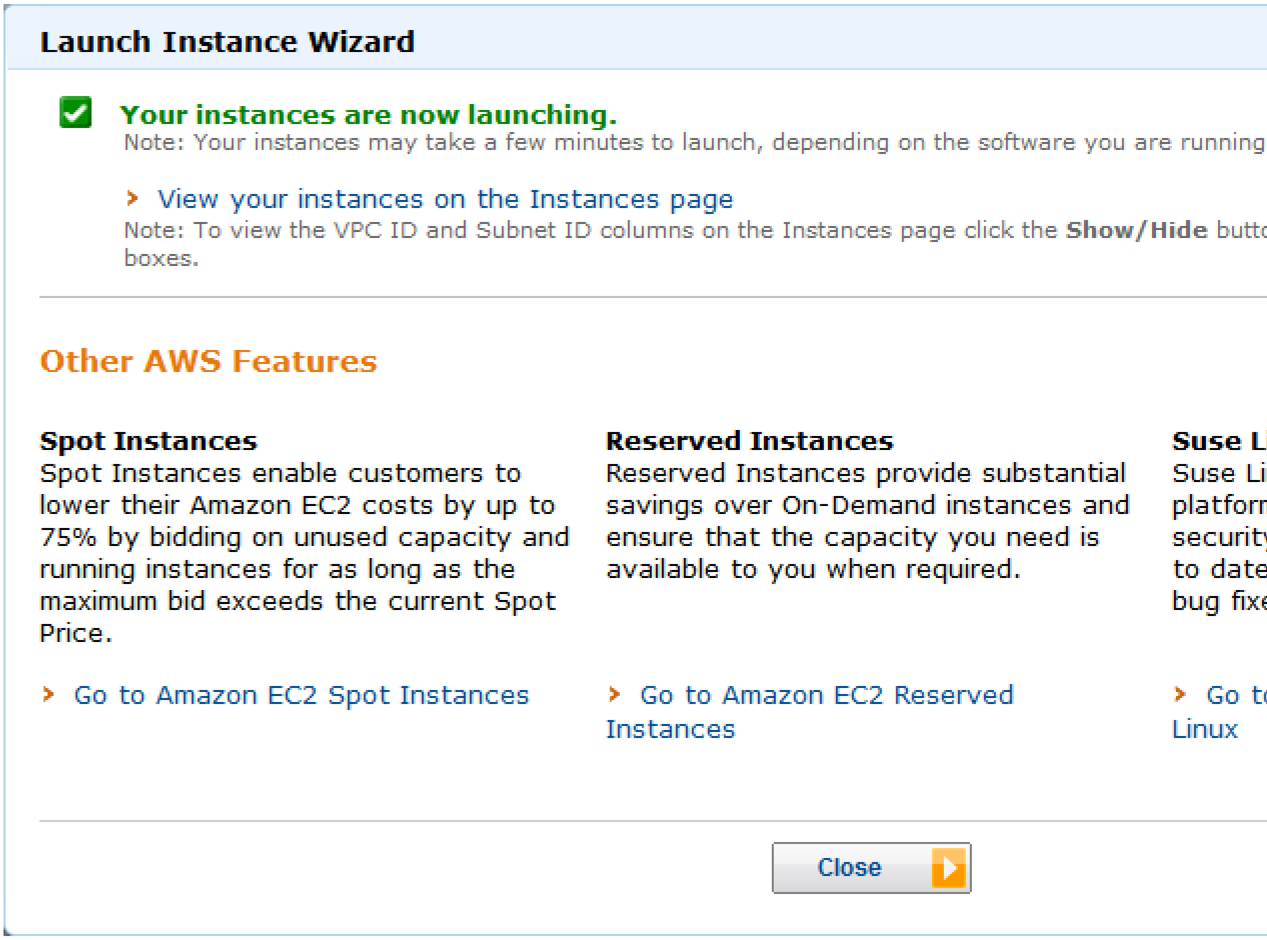
Monitoring: Disabled
Termination Protection: Disabled
Tenancy: Default
Kernel ID: Use Default
Shutdown Behavior: Stop
RAM Disk ID: Use Default
IP Address: Automatically Assigned
User Data:

Key Pair Name: mykeypair

Security Group(s): sg-6dfae901

Additional network interface: eni-4223da2b
Security Group(s): sg-6dfae901

9. On the confirmation page, click **Close**, and then in the **Navigation** pane, click **Instances** to view your instance's status. It takes a short time for an instance to launch. The instance's status is *pending* while it's launching. After a short period, your instance's status switches to *running*. You can click **Refresh** to refresh the display.



Launch Instance Wizard

✓ Your instances are now launching.

Note: Your instances may take a few minutes to launch, depending on the software you are running

› View your instances on the Instances page

Note: To view the VPC ID and Subnet ID columns on the Instances page click the **Show/Hide** butt boxes.

Other AWS Features

Spot Instances
Spot Instances enable customers to lower their Amazon EC2 costs by up to 75% by bidding on unused capacity and running instances for as long as the maximum bid exceeds the current Spot Price.

› Go to Amazon EC2 Spot Instances

Reserved Instances
Reserved Instances provide substantial savings over On-Demand instances and ensure that the capacity you need is available to you when required.

› Go to Amazon EC2 Reserved Instances

Suse L
Suse Li platforn securit to date bug fixe

› Go t Linux

Attach an Elastic Network Interface to a Stopped or Running Instance

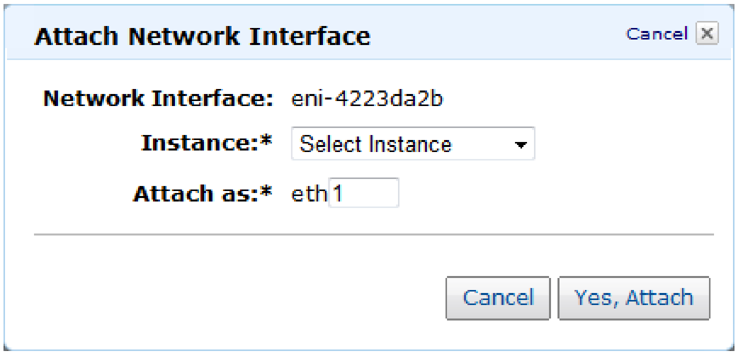
You can attach an ENI to any of your stopped or running instances in your Amazon VPC from either the Instances page or the Network Interfaces page of the EC2 console.

To attach an ENI to a stopped or running instance on the Instances page

1. In the EC2 console, in the **Navigation** pane, click **Instances**.
2. In the **My Instances** pane, right-click an EC2 instance, and then click **Attach Network Interface**.
3. In the **Attach Network Interface** dialog box, select the network interface you want to attach to the instance, and then click **Yes, Attach**.

To attach an ENI to a stopped or running instance on the Network Interfaces page

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Networking Interfaces** pane, select an ENI.
3. At the top of the **Network Interfaces** pane, click **Attach**.
4. In the **Attach Network Interface** dialog box, in the **Instances** pop-up list box, select the instance you want to attach the ENI to, and then click **Yes, Attach**.



Attach Network Interface

Network Interface: eni-4223da2b

Instance:* Select Instance

Attach as:* eth1

Detach an Elastic Network Interface from an Instance

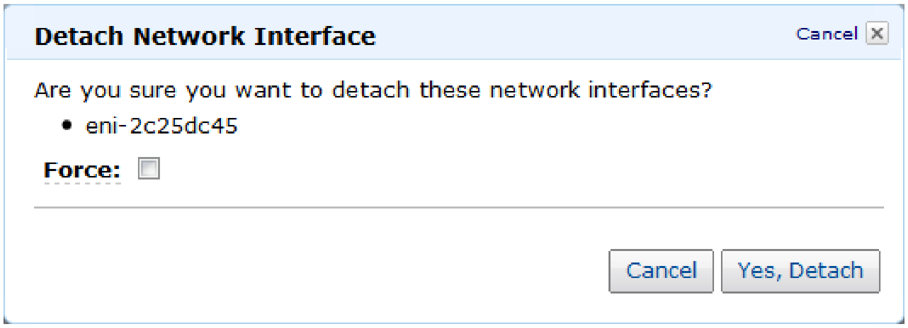
You can detach an eth1 elastic network interface (ENI) any time from either the Instances page or the Network Interfaces page of the EC2 console.

To detach an ENI from an instance on the Instances page

1. In the EC2 console, in the **Navigation** pane, click **Instances**.
2. In the **My Instances** pane, right-click an EC2 instance, and then click **Detach Network Interface**.
3. In the **Detach Network Interface** dialog box, select the network interface you want to detach from the instance, and then click **Yes, Detach**.

To detach an ENI from an instance on the Network Interfaces page

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, select an ENI, and then click **Detach**.
3. In the **Detach Network Interface** dialog box, click **Yes, Detach**.



Detach Network Interface

Are you sure you want to detach these network interfaces?

- eni-2c25dc45

Force: □

Change an Elastic Network Interface’s (ENI’s) Security Group

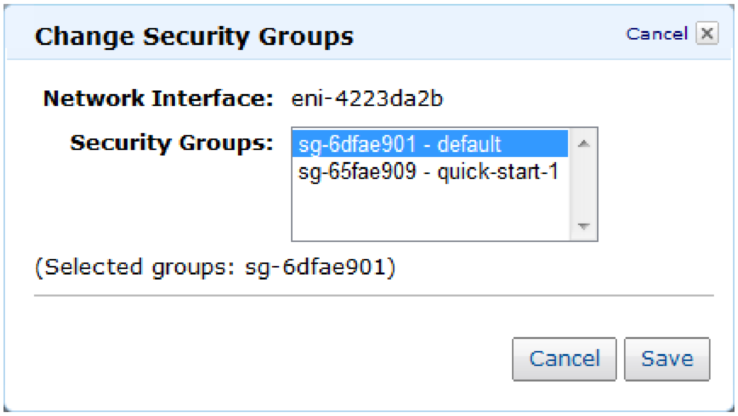
You can change the security groups that are associated with elastic network interfaces (ENIs).

Note

Security group membership for interfaces owned by other Amazon Web Services, such as Elastic Load Balancing, cannot be changed in the **Network Interfaces** page or by using the EC2 API\CLI. To modify the security group membership of an interface owned by one of these services, use the service specific console or API\CLI commands.

To change an ENI's security group

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, select an ENI.
3. At the top of the **Network Interfaces** pane, click **Change Security Groups**.
4. In the **Change Security Groups** dialog box, in the **Security Groups** list box, select the security groups you want to use, and then click **Save**.



Change Security Groups

Network Interface: eni-4223da2b

Security Groups: sg-6dfae901 - default
sg-65fae909 - quick-start-1

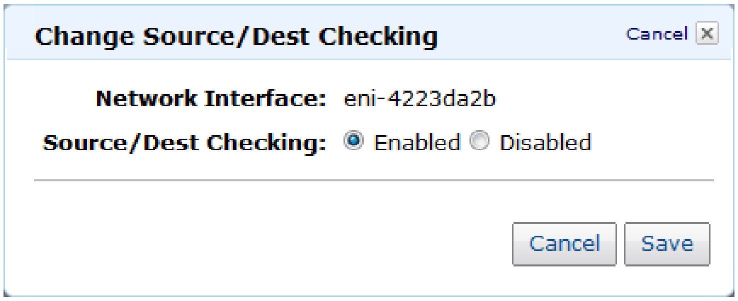
(Selected groups: sg-6dfae901)

Change Source/Destination Checking for an Elastic Network Interface

The Source/Destination Check attribute controls whether source/destination checking is enabled on the instance. Disabling this attribute enables an instance to handle network traffic that isn't specifically destined for the instance. For example, instances running services such as network address translation, routing, or firewalls should set this value to disabled. The default value is enabled.

To change source/destination checking for an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, right-click an ENI, and then click **Change Source/Dest Check**.
3. In the **Change Source/Dest Checking** dialog box, select **Enabled** (if enabling), or **Disabled** (if disabling), and the click **Save**.



Change Source/Dest Checking

Network Interface: eni-4223da2b

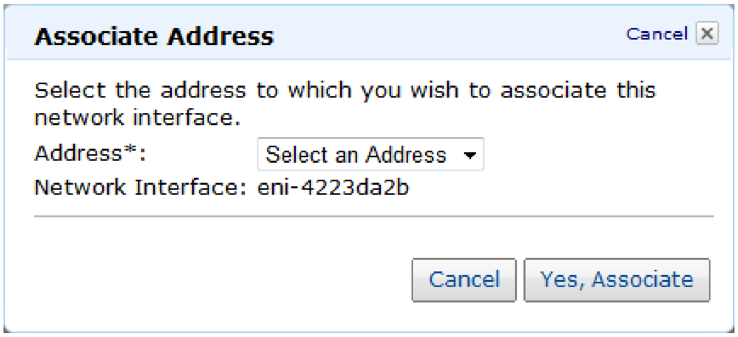
Source/Dest Checking: ● Enabled ○ Disabled

Associate an Elastic IP Address with an Elastic Network Interface

You can associate an elastic IP address with an elastic network interface (ENI).

To associate an elastic IP address with an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, right-click a network interface, and then click **Associate Address**.
3. In the **Associate Address** dialog box, select the elastic IP address you want to associate with your ENI, and then click **Yes, Associate**.



Associate Address

Select the address to which you wish to associate this network interface.

Address*: Select an Address

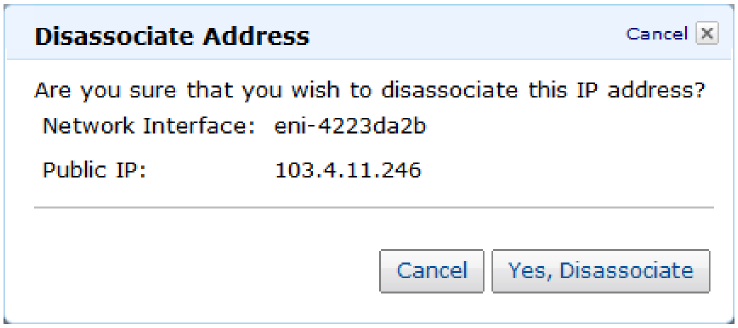
Network Interface: eni-4223da2b

Disassociate an Elastic IP Address from an Elastic Network Interface

If the elastic network interface (ENI) has an elastic IP address associated with it, you can disassociate the address, and then either associate it with another ENI or release it back to the address pool.

To disassociate an elastic IP address from an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, right-click a network interface, and then click **Disassociate Address**.
3. In the **Disassociate Address** dialog box, click **Yes, Disassociate**.



Disassociate Address

Are you sure that you wish to disassociate this IP address?

Network Interface: eni-4223da2b

Public IP: 103.4.11.246

Change Termination Behavior for Elastic Network Interfaces

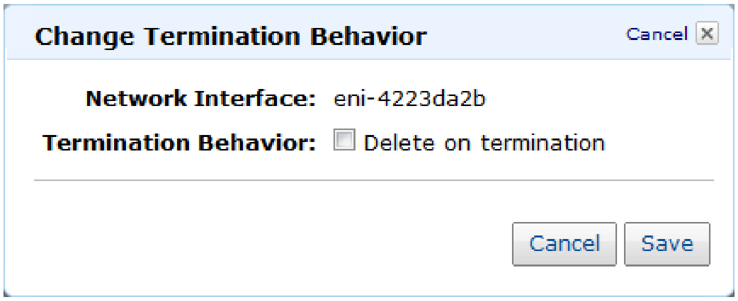
You can set the termination behavior for elastic network interfaces (ENIs) attached to an instance so that they are automatically deleted when you delete the instance they're attached to.

Note

By default, ENIs that are automatically created and attached to instances using the EC2 console (Create Network Interfaces button or the Launch Instances Wizard) are set to terminate when the attached instance terminates. However, ENIs created using the create-network-interface command are not set to be terminated when the instance is terminated.

To change termination behavior for ENIs

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, right-click an ENI, and then click **Change Termination Behavior**.
3. In the **Change Termination Behavior** dialog box, select the **Delete on termination** check box if you want the ENI to be deleted when you terminate an instance.



Change Termination Behavior

Network Interface: eni-4223da2b

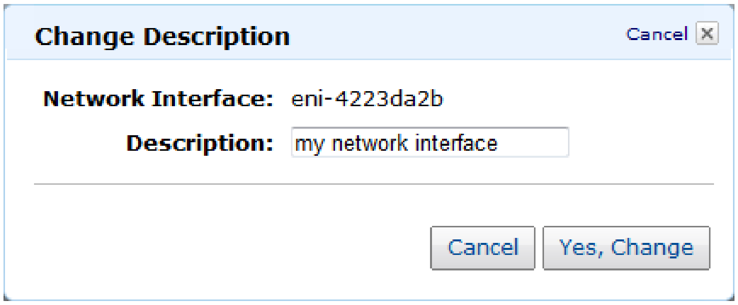
Termination Behavior: □ Delete on termination

Add or Edit a Description for an Elastic Network Interface

To add or edit a description for an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.

2. In the **Network Interfaces** pane, right-click and ENI, and then click **Change Description**.
3. In the **Change Description** dialog box, enter a description for the ENI, and then click **Yes, Change**.



Change Description

Network Interface: eni-4223da2b

Description: my network interface

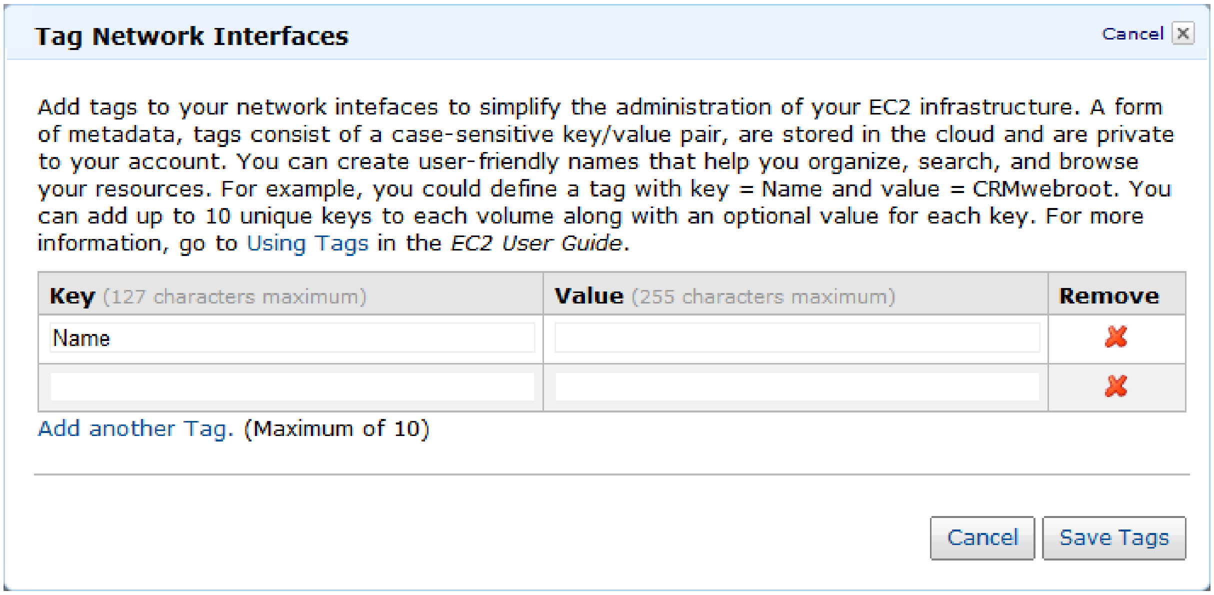
Add or Edit Tags for an Elastic Network Interface

Tags are metadata you can add to an elastic network interface (ENI). Each tag consists of a key and an optional value. For more information about using tags, see Using Tags (p. 275).

Tags are private and are only visible to your account. Tags which have keys that begin with aws: have been created by AWS and cannot be edited or deleted.

To add or edit tags for an ENI

1. In the EC2 console, in the **Navigation** pane, click **Network Interfaces**.
2. In the **Network Interfaces** pane, select an ENI.
3. In the lower pane, click the **Tags** tab, and then click **Add/Edit Tags**.
4. In the **Tag Network Interfaces** dialog box, enter a key and an optional value for each tag that you want to add, and then click **Save Tags**.



Tag Network Interfaces

Add tags to your network intefaces to simplify the administration of your EC2 infrastructure. A form of metadata, tags consist of a case-sensitive key/value pair, are stored in the cloud and are private to your account. You can create user-friendly names that help you organize, search, and browse your resources. For example, you could define a tag with key = Name and value = CRMwebroot. You can add up to 10 unique keys to each volume along with an optional value for each key. For more information, go to Using Tags in the *EC2 User Guide*.

Key (127 characters maximum)	**Value** (255 characters maximum)	**Remove**
Name		✗
		✗

Add another Tag. (Maximum of 10)

Canceling Amazon EC2

If you decide Amazon EC2 isn't what you need, you can cancel your registration at any time.

To cancel any AWS product

1. From http://aws.amazon.com, point to **Your Account** and click **Account Activity**.
2. From the **Account Activity** page, click the **View/Edit Service** link under the service you want to cancel.
3. On the **View/Edit Service** page, click the link to **cancel this service**.

Getting Started with the Command Line Tools

Topics

The following sections describe how to set up your Amazon EC2 environment for use with the Amazon EC2 command line tools (also called the *API tools* or the *CLI tools*).

Note

To start using the AWS Management Console to manage your Amazon EC2 environment, follow the procedures in the Amazon Elastic Compute Cloud Getting Started Guide.

Prerequisites

Topics

This document assumes that the reader is comfortable working in a Linux/UNIX or Windows environment.

An installation of a Java 5 compatible Java Runtime Environment (JRE) is required. Additionally, accessing Linux and UNIX instances requires access to an SSH client and accessing Windows instances requires access to a Remote Desktop client. For more information, refer to the two following sections.

As a convention, all command line text is prefixed with a generic **`PROMPT>`** command line prompt. The actual command line prompt on your machine is likely to be different. We also use **`$`** to indicate a Linux/UNIX specific command and **`C:\>`** for a Windows specific command. Although we don't provide explicit instructions, the tools also work correctly on Mac OS X (which resemble the Linux and UNIX commands). The example output resulting from the command is shown immediately thereafter without any prefix.

Setting the Java Home Variable

The Amazon EC2 command line tools require Java version 5 or later to run. Either a JRE or JDK installation is acceptable. To view and download JREs for a range of platforms, including Linux/UNIX and Windows, go to http://java.sun.com/j2se/1.5.0/.

The command line tools depend on an environment variable (`JAVA_HOME`) to locate the Java runtime. This environment variable should be set to the full path of the directory that contains a subdirectory named `bin` which in turn contains the `java` (on Linux and UNIX) or the `java.exe` (on Windows) executable. You might want to simplify things by adding this directory to your path before other versions of Java. Make sure you don't include the bin directory in the path; that's a common mistake some users make. The command line tools won't work if you do.

Note

If you are using Cygwin, EC2_HOME, EC2_PRIVATE_KEY, and EC2_CERT, you must use Linux/UNIX paths (e.g,, /usr/bin instead of C:\usr\bin). However, JAVA_HOME should have a Windows path. Additionally, the value of EC2_HOME cannot contain any spaces, even if the value is quoted or the spaces are escaped.

On Linux and UNIX, you can set this environment variable as follows.

```
$ export JAVA_HOME=<PATH>
```

Note

The `export` command applies only to the current shell session. To permanently create or update an environment variable, include the command in a start-up script. For example, if you use Bash shell, you can include commands in your `~/.bashrc` or `/etc/profile` file.

An example of the syntax in Windows follows (enclose the path in quotation marks if the path contains spaces).

```
C:\> set JAVA_HOME="<PATH>"
```

Note

The Windows environment variables are reset when you close the command window. You might want to set them permanently with the `setx` command.

You can confirm this by running **`$JAVA_HOME/bin/java -version`** and checking the output.

```
$ $JAVA_HOME/bin/java -version
java version "1.5.0_09"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_09-b03)
Java HotSpot(TM) Client VM (build 1.5.0_09-b03, mixed mode, sharing)
```

The syntax is different on Windows, but the output is similar.

```
C:\> %JAVA_HOME%\bin\java -version
java version "1.5.0_09"
```

```
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_09-b03)
Java HotSpot(TM) Client VM (build 1.5.0_09-b03, mixed mode, sharing)
```

Accessing Linux and UNIX Instances through SSH Clients

For some of the examples illustrated here you'll need access to an SSH client. Most Linux and UNIX installations include an SSH client by default. If yours does not, the OpenSSH project provides a free implementation of the full suite of SSH tools. For more information, go to the http://www.openssh.org.

Windows users can download and install PuTTY, a free SSH client. To download the client and installation instructions, go to the http://www.chiark.greenend.org.uk/~sgtatham/putty/. For information on how to use PuTTY with Amazon EC2, see Connecting to Linux/UNIX Instances from Windows Using PuTTY (p. 117).

Accessing Windows Instances through Remote Desktop Clients

Some of the examples in this guide require a Remote Desktop client. If you do not have one, go to the Microsoft home page.

Setting Up the Tools

Topics

Before you'll be able to use Amazon EC2, you need to download the command line tools and set them up to use your AWS account.

API Tools vs. AMI Tools

The command line tools are also called the *API tools* because they wrap the EC2 API, and to help distinguish them from the *AMI tools*, which are a set of tools for creating (bundling) and migrating certain types of AMIs. In this documentation, the term *command line tools* is equivalent to *API tools*.

The sections that follow describe how to set up the command line tools (API tools). If you find that you need the AMI tools for your particular situation, be aware that many AMIs already have the AMI tools installed. However, if you still need to download them, go to Amazon EC2 AMI Tools.

Where to Get the Command Line Tools (API Tools)

The command line tools (API tools) are available as a ZIP file in the Amazon EC2 API Tools. These tools are written in Java and include shell scripts for both Windows and Linux/UNIX/Mac OSX. The ZIP file is self-contained; no installation is required. Simply download the file and unzip it.

Tell the Tools Where They Live

The command line tools depend on environment variable `EC2_HOME` to locate supporting libraries. Before using the tools, set `EC2_HOME` to the directory path where the command line tools were unzipped.

On Linux and UNIX, you can set this environment variable as follows.

```
$ export EC2_HOME=<path-to-tools>
```

On Windows the syntax is slightly different.

```
C:\> set EC2_HOME=<path-to-tools>
```

In addition, to make your life a little easier, you can add the tools' `bin` directory to your system `PATH`. The rest of this guide assumes you have done so.

On Linux and UNIX, you can update your `PATH` as follows.

```
$ export PATH=$PATH:$EC2_HOME/bin
```

Note

The `export` command applies only to the current shell session. To permanently create or update an environment variable, include the command in a start-up script. For example, if you use Bash shell, you can include commands in your `~/.bashrc` or `/etc/profile` file.

On Windows the syntax is slightly different.

```
C:\> set PATH=%PATH%;%EC2_HOME%\bin
```

Note

The Windows environment variables are reset when you close the command window. You might want to set them permanently with the `setx` command.

Tell the Tools Who You Are

The command line tools need access to the private key and X.509 certificate for your AWS account. For more information, see How to Create an X.509 Certificate and Private Key (p. 267).

Since there's nothing stopping you from having more than one AWS account, you need to identify yourself to the command line API tools so they know which credentials to use for requests. It's possible, but tedious, to provide this information on the command line every time you invoke the tools. It's far simpler to set up some environment variables and be done with it.

There are two environment variables you can set up. They can be set to point at your private key and certificate. After you set these environment variables, the tools use their values to find the relevant credentials. The environment variable `EC2_PRIVATE_KEY` should reference your private key file, and `EC2_CERT` should reference your X.509 certificate.

On Linux and UNIX, you can set these environment variables as follows.

```
$ export EC2_PRIVATE_KEY=~/.ec2/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
$ export EC2_CERT=~/.ec2/cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
```

On Windows the syntax is slightly different.

```
C:\> set EC2_PRIVATE_KEY=c:\ec2\pk-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
C:\> set EC2_CERT=c:\ec2\cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem
```

Set the Service Endpoint URL

By default, the Amazon EC2 tools use the US-East (Northern Virginia) Region (`us-east-1`) with the `ec2.us-east-1.amazonaws.com` service endpoint URL. This section describes how to specify a different Region by setting the service endpoint URL.

Note

Amazon EC2 prices vary by Region. For information about pricing, go to Amazon EC2 Pricing.

To set the service endpoint URL

1. View available Regions with the `ec2-describe-regions` command.

```
PROMPT> ec2-describe-regions
REGION  ap-northeast-1      ec2.ap-northeast-1.amazonaws.com
REGION  ap-southeast-1      ec2.ap-southeast-1.amazonaws.com
..
```

   The result shows the Region names and corresponding service endpoints.

2. Set the service endpoint URL using one of the following commands:

   - For Linux and UNIX:

```
$ export EC2_URL=https://<service_endpoint>
```

   - For Windows:

```
C:\> set EC2_URL=https://<service_endpoint>
```

Where to Go from Here

For an overview of using Amazon EC2 with the command line tools, follow the instructions under Launching and Using Instances (p. 75):

- Finding a Suitable AMI (p. 75)
- Getting an SSH Key Pair (p. 77)
- Adding Rules to the Default Security Group (p. 83)
- Running an Instance (p. 85)
- Stopping and Starting Instances (p. 89)

Using AWS Identity and Access Management

Topics

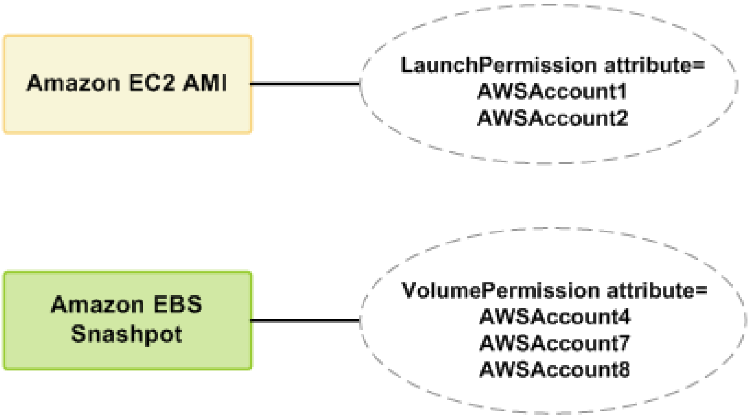
This section provides an introduction to using AWS Identity and Access Management with Amazon EC2.

Identity and Access Management

AWS Identity and Access Management (IAM) is a web service that enables Amazon Web Services (AWS) customers to manage users and user permissions in AWS. The service is targeted at organizations with multiple users or systems that use AWS products such as Amazon EC2 and the AWS Management Console. With IAM, you can centrally manage users, security credentials, such as access key, and permissions that control which AWS resources users can access. For more information on AWS Identity and Access Management, go to Using IAM.

Amazon EC2 Permissions

Amazon EC2 has its own permissions system that covers Amazon Machine Images (AMIs) and Amazon EBS snapshots. There's no ACL system or policy system to give permissions to launch AMIs or create volumes from snapshots. Instead, the Amazon EC2 API lets you modify the attributes on an AMI or snapshot to give another AWS Account those permissions. The following diagram illustrates the concept. Each AMI has a *LaunchPermission* attribute that you can set to one or more AWS Account IDs in order to share the AMI with those AWS Accounts. Each Amazon EBS snapshot has a similar *VolumePermission* attribute.



This sharing works at the AWS Account level only; you can't restrict access only to specific users within the AWS Account you're sharing, or only to specific users in your own AWS Account. All users in the AWS Account you're sharing with can use the AMI or snapshot you've shared.

Note

Don't be confused by the fact that the attribute for specifying the AWS Account to share with is called `UserId`. The value you specify for `UserId` is an AWS Account ID.

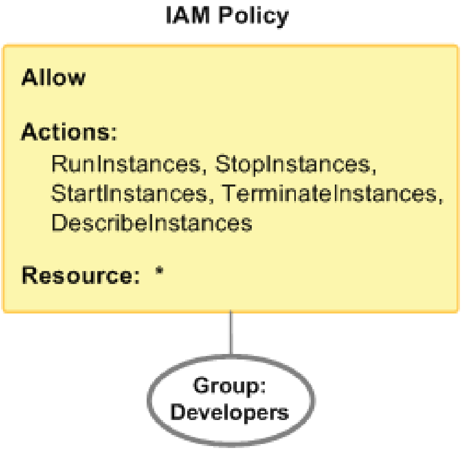
For information on modifying attributes on an AMI, see Sharing AMIs (p. 51). For information on modifying attributes on a snapshot, see Modifying Snapshot Permissions (p. 143)

Amazon EC2 Permissions and AWS Identity and Access Management (IAM)

Using IAM with Amazon EC2 doesn't change how you use the Amazon EC2 API to share AMIs and snapshots with other AWS Accounts. However, you can use IAM policies to specify which Amazon EC2 actions a user in your AWS Account can use with EC2 resources in general. You can't specify a particular Amazon EC2 resource in the policy (e.g,. a specific AMI). Instead, you must specify * as the resource to indicate all resources in the AWS Account.

Example 1: Creating a permission policy

You could create a policy that gives the Developers group permission to use only `RunInstances`, `StopInstances`, `StartInstances`, `TerminateInstances`, and `DescribeInstances`. They could then use those with any AMI that belongs to your AWS Account, any public AMIs, or any AMIs that have been shared with your AWS Account. The following diagram illustrates the concept.

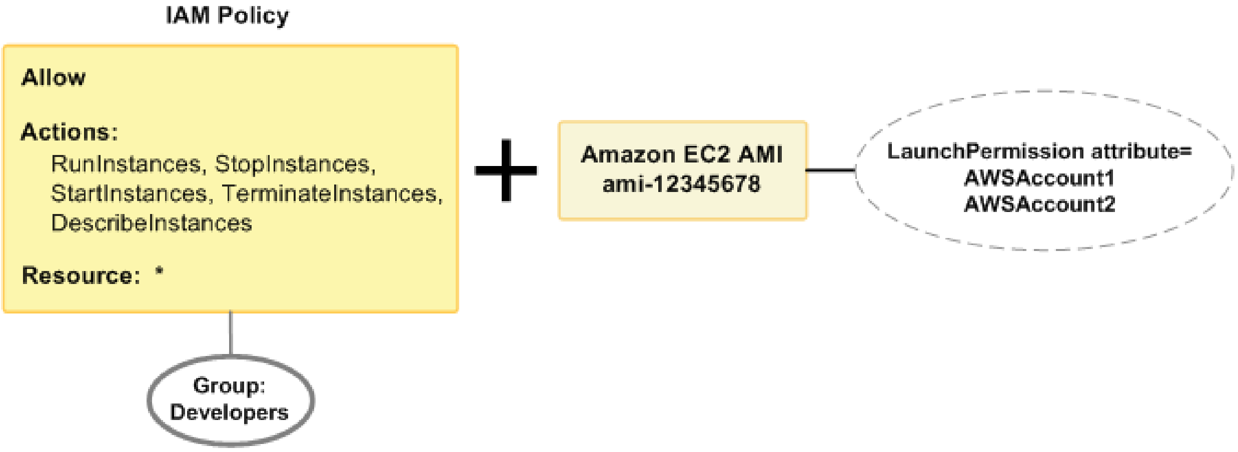


Important

Amazon EC2 uses SSH keys, Windows passwords, and security groups to control who has access to specific Amazon EC2 instances. You can't use the IAM system to allow or deny access to a specific instance.

Example 2: Setting LaunchPermission

This example builds on the previous one. In addition to the IAM policy attached to the Developers group, you set the `LaunchPermission` attribute on the AMI in your AWS Account with ID ami-12345678 so that two other AWS Accounts can launch the AMI. Anyone possessing the keys for either of those AWS Accounts can launch an instance of the AMI. Also, any user in the Developers group can launch an instance of that AMI, because the Developers group has permission to use `RunInstances` with any AMI in the AWS Account.



For examples of IAM policies that cover Amazon EC2 actions, see Example Policies for Amazon EC2 (p. 393). For more information about granting permissions to AMIs and snapshots, refer to the topics about the `ModifyImageAttribute` and `ModifySnapshotAttribute` APIs in the Amazon Elastic Compute Cloud API Reference.

No Amazon Resource Names(ARNs) for Amazon EC2

Amazon EC2 has no Amazon Resource Names (ARNs) because you can t specify a particular Amazon EC2 resource in an IAM policy. When writing a policy to control access to Amazon EC2 actions, you use * as the resource. For more information about ARNs, see Identifiers For IAM Entities.

Amazon EC2 Actions

In an IAM policy, you can specify any and all actions that Amazon EC2 offers. Each action name must be prefixed with the lowercase string `ec2:`. For example: `ec2:RunInstances`, `ec2:CreateImage`, `ec2:*` (for all Amazon EC2 actions). For a list of the actions, refer to the Query API or SOAP API action names in the Amazon Elastic Compute Cloud API Reference.

Amazon EC2 Keys

Amazon EC2 implements the following policy keys:

AWS-Wide Policy Keys

- `aws:CurrentTime` (for date/time conditions)
- `aws:EpochTime` (the date in epoch or UNIX time, for use with date/time conditions)
- `aws:SecureTransport` (Boolean representing whether the request was sent using SSL)
- `aws:SourceIp` (the requester's IP address, for use with IP address conditions)
- `aws:UserAgent` (information about the requester's client application, for use with string conditions)

If you use `aws:SourceIp`, and the request comes from an Amazon EC2 instance, we evaluate the instance's public IP address to determine if access is allowed.

For services that use only SSL, such as Amazon RDS and Amazon Route 53, the `aws:SecureTransport` key has no meaning.

The key names are case insensitive. For example, `aws:CurrentTime` is equivalent to `AWS:currenttime`.

Example Policies for Amazon EC2

This section shows several simple policies for controlling user access to Amazon EC2.

Note

In the future, Amazon EC2 might add new actions that should logically be included in one of the following policies, based on the policy’s stated goals.

Example 1: Allow a group to only be able to describe, run, stop, start, and terminate instances

In this example, we create a policy that gives access to the relevant actions and attach it to the group. The resource is stated as "*", because you can't specify a particular Amazon EC2 resource in an IAM policy.

```
{
   "Statement":[{
      "Effect":"Allow",
      "Action":["ec2:DescribeInstances","ec2:RunInstances",
                "ec2:StopInstances","ec2:StartInstances",
                "ec2:TerminateInstances"],
      "Resource":"*"
      }
   ]
}
```

Example 2: Allow managers to only be able to list the current Amazon EC2 resources in the AWS Account

In this example, we create a policy that lets managers use the Amazon EC2 actions with `Describe` in the name.

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

Example 3: Share an AMI with a partner

You can't use IAM policies to share a particular AMI with a partner; however, you can do that directly through the Amazon EC2 API. The partner needs his or her own AWS Account. You simply use the Amazon EC2 `ModifyImageAttribute` action or `ec2-modify-image-attribute` command to specify the AWS Account ID you want to share the AMI with. For more information about `ModifyImageAttribute`, go to the Amazon Elastic Compute Cloud API Reference. For more information about `ec2-modify-image-attribute`, go to the Amazon Elastic Compute Cloud Command Line Reference.

Using Temporary Security Credentials

In addition to creating IAM users with their own security credentials, IAM also enables you to grant temporary security credentials to any user allowing this user to access your AWS services and resources. You can manage users who have AWS accounts; these users are IAM users. You can also manage users for your system who do not have AWS accounts; these users are called federated users. Additionally, "users" can also be applications that you create to access your AWS resources.

You can use these temporary security credentials in making requests to Amazon EC2. The API libraries compute the necessary signature value using those credentials to authenticate your request. If you send requests using expired credentials Amazon EC2 denies the request.

For more information about IAM support for temporary security credentials, go to Granting Temporary Access to Your AWS Resources in *Using IAM*.

Example Using Temporary Security Credentials to Authenticate an Amazon EC2 Request

The following example demonstrates how to obtain temporary security credentials to authenticate an Amazon EC2 request.

```
https://ec2.amazonaws.com/?Action=DescribeInstances
&InstanceId.0=I-45fa2e72
&Signature=Dqlp3Sd6ljTUA9Uf6SGtEExwUQEXAMPLE
&SignatureVersion=2
&SignatureMethod=HmacSHA256
&Timestamp=2010-03-31T12%3A00%3A00.000Z
&SecurityToken=Security Token Value
&AWSAccessKeyId=Access Key ID provided by AWS Security Token Service
```

Making API Requests

Topics

This section provide an introduction to using the Query and SOAP APIs for EC2.

Required Knowledge

If you plan to access EC2 through its API, we assume you're familiar with the following:

- XML (For an overview, go to the *W3 Schools XML Tutorial*)
- Web services (go to *W3 Schools Web Services Tutorial*)
- HTTP requests
- One or more programming languages

Endpoints

For information about this product's regions and endpoints, go to Regions and Endpoints in the Amazon Web Services General Reference.

If you just specify the general endpoint (ec2.amazonaws.com), the us-east-1 endpoint is used. If you want to use a different region, you need to specify the endpoint associated with it. For example, if you specify ec2.us-west-2.amazonaws.com as the endpoint, Amazon EC2 directs your request to the us-west-2 endpoint which is used for the US West (Oregon) region.

For more information about Regions, see Region and Availability Zone Concepts (p. 321).

Making Query Requests

Topics

Query requests are HTTP or HTTPS requests that use the HTTP verb GET or POST and a Query parameter named `Action`.

Endpoints

For information about this product's regions and endpoints, go to Regions and Endpoints in the Amazon Web Services General Reference.

If you just specify the general endpoint (ec2.amazonaws.com), the us-east-1 endpoint is used.

For more information about Regions, see Region and Availability Zone Concepts (p. 321).

Structure of a GET Request

This guide presents the EC2 GET requests as URLs, which can be used directly in a browser. The URL consists of:

- **Endpoint—**The web service entry point to act on (e.g., ec2.amazonaws.com, which defaults to the us-east-1 Region)
- **Action—**The action you want to perform on the endpoint; for example: running an instance (`RunInstance`)
- **Parameters—**Any request parameters

The following is an example GET request to run instances.

```
https://ec2.amazonaws.com/?Action=RunInstances&ImageId=ami-60a54009&Max
Count=3&MinCount=1&Placement.AvailabilityZone=us-east-1b&Monitoring.En
abled=true&AWSAccessKeyId=0GS7553JW74RRM612K02EXAMPLE&Version=2011-12-01&Ex
pires=2010-10-10T12:00:00Z&Signature=lBP67vCvGlDMBQ1dofZxg8E8SUEXAMPLE&Signa
tureVersion=2&SignatureMethod=HmacSHA256
```

Important

Because the GET requests are URLs, you must URL encode the parameter values. In the EC2 documentation, we leave the example GET requests unencoded to make them easier to read.

To make the GET examples even easier to read, this guide presents them in the following parsed format.

```
https://ec2.amazonaws.com/
?Action=RunInstances
&ImageId=ami-60a54009
&MaxCount=3
```

```
&MinCount=1
&Placement.AvailabilityZone=us-east-1b
&Monitoring.Enabled=true
&AWSAccessKeyId=0GS7553JW74RRM612K02EXAMPLE
&Version=2011-12-01
&Expires=2010-10-10T12:00:00Z
&Signature=lBP67vCvGlDMBQ1dofZxg8E8SUEXAMPLE
&SignatureVersion=2
&SignatureMethod=HmacSHA256
```

The first line represents the *endpoint* of the request. After the endpoint is a question mark (?), which separates the endpoint from the parameters. Each parameter is separated by an ampersand (&).

The *`Action`* parameter indicates the action to perform (for a list of the actions, go to Amazon Elastic Compute Cloud API Reference).

Note

In the example Query requests we present in the EC2 documentation, we omit the parameters related to authentication to make it easier to focus on the ones relevant to the particular operation. We replace them with the following literal string to remind you that a real request includes the parameters: `&AuthParams`.

Query Parameters

Each Query request must include some common parameters to handle authentication and selection of an action. For more information, go to the Amazon Elastic Compute Cloud API Reference.

Some operations take lists of parameters. These lists are specified using the *`param.`n* notation. Values of *n* are integers starting from 1.

Query API Authentication

You can send Query requests over either HTTP or HTTPS. Regardless of which protocol you use, you must include a signature in every Query request. This section describes how to create the signature. The method described in the following procedure is known as *signature version 2*.

Caution

If you are currently using signature version 1: Version 1 is deprecated, and you should move to signature version 2 immediately. For information about the deprecation schedule and the differences between signature version 2 and version 1, go to Making Secure Requests to Amazon Web Services.

To create the signature

1. Create the canonicalized query string that you need later in this procedure:

   a. Sort the UTF-8 query string components by parameter name with natural byte ordering.
   The parameters can come from the GET URI or from the POST body (when `Content-Type` is `application/x-www-form-urlencoded`).

   b. URL encode the parameter name and values according to the following rules:

      - Do not URL encode any of the unreserved characters that RFC 3986 defines.

These unreserved characters are A-Z, a-z, 0-9, hyphen (-), underscore (_), period (.), and tilde (~).

- Percent encode all other characters with %XY, where X and Y are hex characters 0-9 and uppercase A-F.
- Percent encode extended UTF-8 characters in the form %XY%ZA....
- Percent encode the space character as %20 (and not +, as common encoding schemes do).

Note

Currently all AWS service parameter names use unreserved characters, so you don't need to encode them. However, you might want to include code to handle parameter names that use reserved characters, for possible future use.

c. Separate the encoded parameter names from their encoded values with the equals sign (=) (ASCII character 61), even if the parameter value is empty.

d. Separate the name-value pairs with an ampersand (&) (ASCII code 38).

2. Create the string to sign according to the following pseudo-grammar (the `"\n"` represents an ASCII newline).

```
StringToSign = HTTPVerb + "\n" +
               ValueOfHostHeaderInLowercase + "\n" +
               HTTPRequestURI + "\n" +
               CanonicalizedQueryString <from the preceding step>
```

   The HTTPRequestURI component is the HTTP absolute path component of the URI up to, but not including, the query string. If the HTTPRequestURI is empty, use a forward slash (/).

3. Calculate an RFC 2104-compliant HMAC with the string you just created, your Secret Access Key as the key, and SHA256 or SHA1 as the hash algorithm.
   For more information, go to http://www.ietf.org/rfc/rfc2104.txt.
4. Convert the resulting value to base64.
5. Use the resulting value as the value of the *`Signature`* request parameter.

Important

The final signature you send in the request must be URL encoded as specified in RFC 3986 (for more information, go to http://www.ietf.org/rfc/rfc3986.txt. If your toolkit URL encodes your final request, then it handles the required URL encoding of the signature. If your toolkit doesn't URL encode the final request, then make sure to URL encode the signature before you include it in the request. Most importantly, make sure the signature is URL encoded *only once*. A common mistake is to URL encode it manually during signature formation, and then again when the toolkit URL encodes the entire request.

Example DescribeImages Request

```
https://ec2.amazonaws.com/
?Action=DescribeImages
&ImageId.1=ami-2bb65342
&Version=2011-12-01
&Expires=2008-02-10T12%3A00%3A00Z
&SignatureVersion=2
&SignatureMethod=HmacSHA256
&AWSAccessKeyId=<Your AWS Access Key ID>
```

Following is the string to sign.

```
GET\n
ec2.amazonaws.com\n
/\n
AWSAccessKeyId=<Your AWS Access Key ID>
&Action=DescribeImages
&Expires=2008-02-10T12%3A00%3A00Z
&ImageId.1=ami-2bb65342
&SignatureMethod=HmacSHA256
&SignatureVersion=2
&Version=2011-12-01
```

Following is the signed request.

```
https://ec2.amazonaws.com/
?Action=DescribeImages
&ImageId.1=ami-2bb65342
&Version=2011-12-01
&Expires=2008-02-10T12%3A00%3A00Z
&Signature=<URLEncode(Base64Encode(Signature))>
&SignatureVersion=2
&SignatureMethod=HmacSHA256
&AWSAccessKeyId=<Your AWS Access Key ID>
```

Making SOAP Requests

Topics

Endpoints

For information about this product's regions and endpoints, go to Regions and Endpoints in the Amazon Web Services General Reference. If you just specify the general endpoint (ec2.amazonaws.com), the us-east-1 endpoint is used. For more information about Regions, see Region and Availability Zone Concepts (p. 321).

WSDL and Schema Definitions

The Amazon EC2 web service can be accessed using the SOAP web services messaging protocol. This interface is described by a Web Services Description Language (WSDL) document which defines the operations and security model for the service. The WSDL references an XML Schema document which strictly defines the data types that might appear in SOAP requests and responses.

All schemas have a version number (the latest is 2011-12-01). The version number appears in the URL of a schema file, and in a schema's target namespace. This makes upgrading easy by differentiating requests based on the version number.

Note

In addition to the latest version, the service will support the older versions for some time, allowing customers plenty of time to upgrade.

The Amazon EC2 services API WSDL is available from the web at 'http://s3.amazonaws.com/ec2-downloads/ec2.wsdl'. At the time this document was released, the current API version was 2011-12-01.

The following are additional web service references.

- Web Service Description Language (WSDL)
- WS-Security BinarySecurityToken Profile

Programming Language Support in Amazon EC2

Since the SOAP requests and responses in the Amazon EC2 Web Service follow current standards, any programming language with the appropriate library support can be used. Languages known to have this support include C++, C#, Java, Perl, Python and Ruby.

Request Authentication

To prevent in-flight tampering, all SOAP requests should be sent over HTTPS. In addition, the service complies with the current WS-Security standard, requiring SOAP request messages to be hashed and signed for integrity and non-repudiation. WS-Security defines profiles which are used to implement various levels of security. Amazon EC2 secure SOAP messages use the BinarySecurityToken profile, consisting of an X.509 certificate with an RSA public key.

The following is the content of an insecure `RunInstances` operation:

```
<RunInstances xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
    <instancesSet>
        <item>
            <imageId>ami-60a54009</imageId>
            <minCount>1</minCount>
            <maxCount>3</maxCount>
        </item>
    </instancesSet>
    <groupSet/>
</RunInstances>
```

To secure the request, we add the BinarySecurityToken element. The Java libraries we supply rely on the Apache Axis project for XML security, canonicalization, and SOAP support. The Sun Java Web Service Developer's Pack supplies libraries of equivalent functionality.

The secure version of the request begins with the following:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">

  <SOAP-ENV:Header>
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
      <wsse:BinarySecurityToken
      xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-utility-1.0.xsd"
      EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary"
      ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
token-profile-1.0#X509v3"
      wsu:Id="CertId-1064304">....many, many lines of base64 encoded
      X.509 certificate...</wsse:BinarySecurityToken>
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:SignedInfo>
          <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-
exc-c14n#"></ds:CanonicalizationMethod>
         <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></ds:SignatureMethod>
          <ds:Reference URI="#id-17984263">
            <ds:Transforms>
              <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmld
sig#sha1"></ds:DigestMethod>
            <ds:DigestValue>0pjZ1+TvgPf6uG7o+Yp3l2YdGZ4=</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#id-15778003">
            <ds:Transforms>
              <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></ds:Transform>
            </ds:Transforms>
            <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmld
sig#sha1"></ds:DigestMethod>
            <ds:DigestValue>HhRbxBBmc2OO348f8nLNZyo4AOM=</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
       <ds:SignatureValue>bmVx24Qom4kd9QQtclxWIlgLk4QsQBPaKESi79x479xgbO9PEStXMi
HZuBAi9luuKdNTcfQ8UE/d
       jjHKZKEQRCOlLVy0Dn5ZL1RlMHsv+OzJzzvIJFTq3LQKNrzJzsNe</ds:SignatureValue>

        <ds:KeyInfo Id="KeyId-17007273">
          <wsse:SecurityTokenReference
              xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd" wsu:Id="STRId-22438818">
            <wsse:Reference URI="#CertId-1064304"
                            ValueType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3">
            </wsse:Reference>
          </wsse:SecurityTokenReference>
        </ds:KeyInfo>
      </ds:Signature>
      <wsu:Timestamp
          xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd" wsu:Id="id-17984263">
        <wsu:Created>2006-06-09T10:57:35Z</wsu:Created>
        <wsu:Expires>2006-06-09T11:02:35Z</wsu:Expires>
      </wsu:Timestamp>
    </wsse:Security>
  </SOAP-ENV:Header>
```

If you are matching this against requests generated by Amazon EC2 supplied libraries, or those of another vendor, the following are the most important elements:

Elements

- **BinarySecurityToken—**Contains the X.509 certificate in base64 encoded PEM format
- **Signature—**Contains an XML digital signature created using the canonicalization, signature algorithm, and digest method
- **Timestamp—**Requests to Amazon EC2 are valid within 5 minutes of this value to help prevent replay attacks

The Response Structure

In response to either a Query or SOAP request, the service returns an XML data structure that conforms to an XML schema defined as part of the EC2 WSDL. The structure of an XML response is specific to the associated request. In general, the response data types are named according to the operation performed and whether the data type is a container (can have children). Examples of containers include `groupSet` for security groups and `keySet` for key pairs (see the example that follows). Item elements are children of containers, and their contents vary according to the container's role.

Every EC2 response includes a request ID in a `requestId` element. The value is a unique string that AWS assigns. If you ever have issues with a particular request, AWS will ask for the request ID to help troubleshoot the issue. The following shows an example response.

```
<DescribeKeyPairsResponse xmlns="http://ec2.amazonaws.com/doc/2011-12-01/">
  <requestId>7a62c49f-347e-4fc4-9331-6e8eEXAMPLE</requestId>
  <keySet>
    <item>
      <keyName>gsg-keypair</keyName>
      <keyFingerprint>
         1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f
      </keyFingerprint>
    </item>
  </keySet>
</DescribeKeyPairsResponse>
```

Available Libraries

AWS provides libraries, sample code, tutorials, and other resources for software developers who prefer to build applications using language-specific APIs instead of SOAP and Query. These libraries provide basic functions (not included in the APIs), such as request authentication, request retries, and error handling so that it is easier to get started. Libraries and resources are available for the following languages:

- Java
- PHP
- Python
- Ruby
- Windows and .NET

For libraries and sample code in all languages for Amazon EC2, go to http://aws.amazon.com/code/Amazon-EC2. For libraries and sample code in all languages in Amazon Web Services, go to Sample Code & Libraries.

Technical FAQ

Topics

This section contains answers to commonly asked questions.

General Information FAQ

How many instances can I launch?

Each AWS account has a concurrent running instance limit. For new accounts, this limit is 20. If you need more than 20 instances, please complete the Amazon EC2 Instance Request Form and your request will be considered.

How do I sign a request?

See Making API Requests (p. 396).

Why do my instances take so long to start?

Amazon EC2 must move the images around the network before they can be launched. For big images and/or congested networks, this can take several minutes. To improve performance, images are cached. As you launch your images more frequently, it should be less noticeable.

If you need faster launch times, consider using AMIs backed by Amazon EBS.

Can I get a bigger/smaller/differently optimized virtual machine?

Yes. For more information, see Instance Families and Types (p. 69).

Is there a REST interface to Amazon EC2?

Not at present. You can use the Query API, SOAP API, or the command line tools.

How does Amazon EC2 handle load balancing?

With a service as flexible as Amazon EC2, you can use many types of load balancing systems. The load balancing instances can forward traffic to other systems. There are several open source solutions that are in wide use.

Does Amazon perform system maintenance?

Yes. Periodically, Amazon might perform maintenance that requires a reboot of your system. Make sure your instances can recover and restart after being rebooted.

Instances and AMIs Information FAQ

How durable are the instance stores?

Instance stores appear to an instance as a local disk. They will survive intentional and unintentional reboots of the instance unless the instance terminates or the underlying drive fails.

You should always back up or replicate important data.

What happens to my running instances if the machines on which they are running fail?

The instances terminate and need to be relaunched. The data on the instance stores is lost. Any data on Amazon EBS root volumes (for Amazon EBS-backed instances) is also lost by default. The default behavior of the root volume on the Amazon EBS-backed instance can be changed by setting the the value of the `deleteOnTermination` flag to `false`. For more information, see Root Device Storage (p. 160). The data on the non-root Amazon EBS volumes (if attached) is preserved, by default, on instance termination.

Always replicate important data: use Amazon EBS, or store it in Amazon S3.

What are the differences between stopping and terminating an instance?

The instance will first perform a normal shutdown in both cases. If the instance was stopped, it will then transition to a stopped state, all of its Amazon EBS volumes will remain attached, and it can then be started again at a later time. If the instance was terminated, attached Amazon EBS volumes will be deleted if the volume's `deleteOnTermination` flag is set to true. The instance itself will also be deleted, and the instance cannot be started again at a later date.

If you wish to disable termination via the `TerminateInstances` API for increased safety, ensure that the `disableApiTermination` attribute is set to true for the instance. To control the behavior of an on instance shutdown such as shutdown –h in Linux or shutdown in Windows, set the `instanceInitiatedShutdownBehavior` instance attribute to `stop` or `terminate` as desired. Instances that have Amazon EBS volume root devices will default to `stop`, and instances with instance-store root devices will always be terminated as the result of an on instance shutdown.

How does stopping and starting an instance affect how it is charged?

You will not be charged for additional instance hours while the instance is in a stopped state. A full instance hour will be charged for every transition from a stopped state to a running state, even if this happens multiple times within a single hour. If the instance type was changed while the instance was stopped, you will be charged at the new instance type's rate once the instance has been started. All of the associated Amazon EBS usage of your instance, including root device usage, will be charged at the typical Amazon EBS prices.

What can I do with a stopped instance?

You can attach or detach Amazon EBS volumes. You can use the `CreateImage` call to create an AMI from the instance. You can change the kernel, RAM disk, and instance type with the `ModifyInstanceAttribute` call. You can restart the instance with the `StartInstances` call, and terminate the instance with the `TerminateInstances` call.

What limitations exist for stopped instances?

In addition to the limit on running instances, there is an additional limit on the overall number of instances that you can have (whether running, stopped, or in any other state except for terminated.) This overall instance limit is 2 times your running instance limit. The running instance limit can be increased upon request via the instance limit request form.

Do I need to share the Amazon EBS snapshots that an AMI references in order to share the AMI?

No, only the AMI itself needs to be shared. The system will automatically provide the instance access to the referenced Amazon EBS snapshots for the purposes of the launch.

How can I control how block devices are exposed to my instance?

See Block Device Mapping (p. 167).

Operation Information FAQ

How do I handle time synchronization between instances?

You can set up NTP (Network Time Protocol). For more information, go to www.ntp.org. NTP is particularly important if you plan on using any AWS products (such as Amazon S3 or Amazon EC2) from within an instance, because requests to these services must be timestamped.

Is there a method for an instance to discover its own instance ID?

From within your instance you can use REST-like queries to http://169.254.169.254/latest/ to retrieve various instance-specific metadata, including the instance ID. For more information, see Using Instance Metadata (p. 96).

Can I pass arbitrary configuration values to an instance at launch time?

Yes, although the size of the data is limited to 16K. For more information, see User Data Retrieval (p. 99).

Is there a way to run a script on instance termination?

Amazon EC2 tries to shut an instance down cleanly (running system shutdown scripts), but there is only a short time available. In some cases (e.g., hardware failure), this does not happen.

Because there is no way to ensure shutdown scripts run, have a strategy to deal with abnormal terminations.

How can I allow other people to launch my AMIs?

You can allow other AWS accounts to launch your AMIs by modifying the AMI's `launchPermission` attribute. You can grant public launch permissions or explicit permissions to specific AWS accounts. For more information, see Sharing AMIs Safely (p. 47).

Why do I need to reregister a rebundled Amazon EC2 instance store-backed AMI? Can I keep the same AMI ID?

An AMI ID is associated with the physical bits in an image. To protect users from images being modified, we require you to reregister Amazon EC2 instance store-backed AMIs after rebundling.

How can I push an Amazon S3 object to an Amazon EBS volume?

You can push an Amazon S3 object to an Amazon EBS volume by mounting the Amazon EBS volume on an Amazon EC2 instance and then using a data transfer application (e.g., FTP/SFTP, SCP) on your running instance to transfer the data.

Can I pass JVM properties to the command line tools?

Yes. By setting the environment variable `EC2_JVM_ARGS`, you can pass arbitrary JVM properties to the command line tools.

Can I use a proxy with the command line tools?

Yes. By passing in JVM properties through the `EC2_JVM_ARGS` environment variable, you can specify proxy settings for the command line tools. For example, in Linux and UNIX:

```
export EC2_JVM_ARGS="-Dhttp.proxyHost=http://my.proxy.com -Dhttp.proxyPort=8080"
```

Properties for configuring a proxy are described in the following table.

Setting	Description
https.proxyHost	HTTPS proxy host
https.proxyPort	HTTPS proxy port
http.proxyHost	HTTP proxy host
http.proxyPort	HTTP proxy port
http.proxyRealm	Proxy realm (https and http)
http.proxyUser	Proxy username (https and http)
http.proxyPass	Proxy password (https and http)

Note

`https.proxyHost` should be used when `EC2_URL` points to an https host, and `http.proxyHost` when `EC2_URL` points to an http host.

Instance Types and Architectures FAQ

What happened to the original instance type?

The original instance type is still available. It is called the small instance (m1.small) and it has the same technical specifications.

Will the original instance type be retired soon?

There are no plans to retire the original instance type.

If I do not specify an instance type at launch, what type of instance will I get?

You will get a m1.small Amazon EC2 instance type.

Does my instance limit apply to all instance types or is there a separate limit for each type?

The instance limit applies to the sum of all instances, regardless of type. There is no separate instance limit per type.

Can I mix instance types, or do I have to use the same type for all of my instances?

You can launch any combination of instance types. Choose the instance types that have the most appropriate memory, CPU, and storage for each function within your application.

How do I select the right instance type?

Amazon EC2 instances are grouped into several families. For more information, refer to Instance Families and Types (p. 69).

Tip

One of the advantages of Amazon EC2 is that you pay by the instance hour, which makes it convenient and inexpensive to test the performance of your application on different instance families and types. A good way to determine the most appropriate instance family and instance type is to launch test instances and benchmark your application.

When should I use High-CPU instance types (c1.medium and c1.xlarge)?

High-CPU instance types have a proportionately higher ratio of CPU to memory and are well suited for compute-intensive applications. To determine whether they are appropriate for you, launch an instance and benchmark your own application on different instance types and calculate which is most appropriate.

Which instance types are 32-bit and which are 64-bit?

The small (m1.small) and High-CPU medium (c1.medium) instances are 32-bit. The large (m1.large), extra large (m1.xlarge), High-CPU extra large (c1.xlarge), cluster compute (cc1.4xlarge), cluster GPU (cg1.4xlarge), and High-Memory (m2.xlarge, m2.2xlarge, m2.4xlarge) instances are 64-bit. The micro instances (t1.micro) are available in either 32-bit or 64-bit.

Can I launch any AMI on any type of instance?

No. You must use 64-bit AMIs on large (m1.large), extra large (m1.xlarge), High-Memory (m2.xlarge, m2.2xlarge, m2.4xlarge), cluster compute (cc1.4xlarge), cluster GPU (cg1.4xlarge), and High-CPU extra large (c1.xlarge) instances. You must use 32-bit AMIs on small (m1.small) and High-CPU medium (c1.medium) instances. Cluster instances must be launched on Linux/UNIX HVM AMIs. You can use either 32-bit or 64-bit AMIs when launching micro instances (t1.micro).

Can I use my own kernel?

Yes. For more information, see Enabling Your Own Linux Kernels (p. 61).

Do I have to do anything special to bundle the large or extra large Amazon EC2 instance store-backed instances?

Make sure to use the latest AMI Tools.

Can I build an AMI that works on both 32-bit and 64-bit instances?

No, an AMI is either a 32-bit AMI or a 64-bit.

Can I run 32- bit applications on 64-bit AMIs?

You can run a 32-bit application on a 64-bit host if the Linux/UNIX kernel is compiled with IA32 emulation and the correct 32-bit libraries are available.

By default, the Amazon DomU Kernel has IA32 emulation enabled and there are many public AMIs that include preinstalled 32-bit libraries. If the library you require is not included with the AMI, you can install it using standard tools (e.g., yum).

How fast is the disk?

The instance stores on large and extra large instances have higher and more consistent I/O performance than the original (small) instance. Amazon EBS volumes perform consistently for all instance types.

Note

The first write to any given block of the disk will be slower than subsequent writes. For more information, see Disk Performance Optimization (p. 159)

Can I RAID the spindles exposed on large and extra large instances?

Yes, you can use software RAID on top of the exposed spindles.

Note

The initial RAID setup might take a long time. For more information, see Disk Performance Optimization (p. 159)

IP Information FAQ

How do I host a public domain if I have to DHCP an IP address?

You can use a dynamic DNS service, such as DynDNS or ZoneEdit. Alternatively, you can map an elastic IP address to your instance and avoid the propagation delays possible with a dynamic DNS solution.

Why do I get an internal (RFC 1918) IP address when I look up a DNS name that I expect to map to my instance's external IP address?

The Amazon EC2 DNS servers return the internal IP address when asked about an instance's public DNS name. In this way, DNS lookups that would resolve to a public Amazon EC2 IP address will be translated to the correct internal IP address. This only works when using the Amazon EC2 DNS servers from an Amazon EC2 instance.

Why is Amazon EC2 Using NAT?

Public IP space is a limited resource. Amazon EC2 is adopting NAT to ensure that we are able to efficiently make use of our public Internet addresses.

Furthermore, the new NAT networking will enable Amazon to deliver new features in the future. For example, some users might not want external addresses. This would allow for non-Internet routable clusters, which will further preserve IPs and increase security for those not running public facing servers.

Can I use a static IP in my instances?

Not at present. Your image must be configured as a DHCP client and it will be assigned an IP address. Currently, all instances come with Internet- addressable IP addresses. If you enable access through the firewall from the "world", you can address them from anywhere.

How does the instance know its public and private addresses?

From within the instance, issue the following HTTP queries:

To obtain the internal IP address:

```
curl http://169.254.169.254/latest/meta-data/local-ipv4
```

To obtain the public IP address:

```
curl http://169.254.169.254/latest/meta-data/public-ipv4
```

Why am I limited to 5 Elastic IP addresses?

Public (IPV4) Internet addresses are a scarce public resource. Amazon EC2 is committed to helping use that space efficiently.

By default, all accounts are limited to 5 Elastic IP addresses. If you need more than 5 Elastic IP addresses, please complete the Amazon EC2 Elastic IP Address Request Form. We will ask you to think through your use case and help us understand your need for additional addresses.

Is my Elastic IP addressed fixed to a single instance?

Unlike a traditional dedicated IP addresses, an Elastic IP can be assigned to many different instances over time.

Is there a minimum usage required for Elastic IP addresses?

When operating within the 5 address limit, you can leave addresses unattached as you need. However, we reserve the right to reclaim Elastic IP addresses that are chronically underutilized.

Is there a charge for Elastic IP addresses?

To ensure our customers are efficiently using Elastic IP addresses, we impose a small hourly charge when these IP addresses are not mapped to an instance. When these IP addresses are mapped to an instance, they are free of charge. To avoid charges for Elastic IP addresses that you are not using, use `ReleaseAddress`.

Do I need one Elastic IP address for each instance that I have running?

No. By default, every instance comes with a private IP address and an Internet routable public IP address. These addresses are fixed for the life of the instance (exception: Amazon EBS-backed instances typically change IP addresses when you stop and restart them). We believe the private and public IP addresses should be adequate for many applications where you do not need a long lived Internet routable end point (e.g., compute clusters, web crawling, and backend services).

What happens to an Elastic IP address if I stop the instance associated with it?

The Elastic IP address is disassociated from the instance when the instance is stopped.

Why don't you use IPV6 addresses?

Because of the scarcity of IPV4 Internet address, Amazon EC2 will be actively investigating the use of IPV6 addresses. We believe this is the only tenable long term solution. We don't yet have a timeline for introducing IPV6 addresses.

Can I launch an instance with no public IP address?

You cannot currently launch an instance without a public IP address. We understand that for many applications, it is desirable to have no Internet routable IP address (e.g., internal databases).

Note

Instances you launch into a VPC do not have a public IP address by default. For more information about Amazon Virtual Private Cloud, go to the Amazon Virtual Private Cloud Getting Started Guide.

How long does it take to remap an Elastic IP address?

After you successfully make an API call to remap an IP address, it will usually occur within a few minutes.

Will I be charged for the time when my IP address is unattached because my instance failed?

You are not charged until your Elastic IP address has been unattached for a full hour. As long as you are monitoring your instances, you will have plenty of time to reattach your instance before the charge is metered.

Am I limited to 100 Elastic IP remaps per month?

No. The first 100 remaps per account are free. After that, there will be a charge for each remap.

Can I configure the reverse DNS record for my Elastic IP address?

Yes, you can configure the reverse DNS record of your Elastic IP address (submit the Request to Remove Email Sending Limitations form). Note that a corresponding forward DNS record pointing to that Elastic IP address must exist before we can create the reverse DNS record.

Region and Availability Zone FAQ

For a conceptual overview of regions and endpoints, see Using Regions and Availability Zones (p. 321).

Why aren't Regions tightly integrated with each other?

We isolate the Regions from each other to achieve greater fault tolerance, improve stability, and to help prevent issues within one Region from affecting another. To simplify using instances across regions, we provide tools such as `ec2-migrate-image` and `ec2-migrate-manifest`. For more information, go to Amazon Elastic Compute Cloud Command Line Reference.

How do I interact with EC2 in different Regions?

Use the Region-specific service endpoint for the Region you want. To get a list of Regions and their endpoints, use the `DescribeRegions` API (i.e., the **ec2-describe-regions** command), for example:

```
PROMPT> ec2-describe-regions
REGION  ap-northeast-1       ec2.ap-northeast-1.amazonaws.com
REGION  ap-southeast-1       ec2.ap-southeast-1.amazonaws.com
..
```

For more information, see Specifying the Region to Use (p. 325).

How do I launch an AMI in another Region?

For Amazon EC2 instance store-backed AMIs: Copy your AMI from its current bucket to a bucket located in the Region where you want to launch the AMI, and register the AMI. For example, to launch a US-based AMI in the EU Region, copy the AMI to an Amazon S3 bucket that was created with an EU location constraint. After the AMI is copied, you must register the AMI and use the obtained AMI ID for launches in the new Region.

Also, make sure to give read access to the bucket, image manifest, and image parts to ec2-bundled-images@amazon.com for Windows AMIs, and za-team@amazon.com for Linux AMIs.

What tools are available to help migrate my AMIs to a new Region?

The API Tools contain a new command called `ec2-migrate-image`. It is designed to help migrate AMIs to a new Region. Run ec2-migrate-image --help for more details. Also go to Amazon Elastic Compute Cloud Command Line Reference.

Can I use the same SSH key pair across Regions?

This question refers only to the key pair used for SSH connections to the instance. Don't confuse this with your AWS Account ID and other credentials, which are global and work in all Regions.

The SSH key pairs that you create with `ec2-add-keypair`, `CreateKeyPair`, or in the AWS Management Console work only in the Region where you create them. However, you can optionally create an RSA key pair with a third-party tool and upload the public key to AWS. That key pair works in all Regions. For more information, go to ec2-import-keypair in the *Amazon Elastic Compute Cloud Command Line Reference* or ImportKeyPair in the *Amazon Elastic Compute Cloud API Reference*.

How do I launch an Amazon EBS volume from a snapshot across Regions?

At this time, snapshots cannot be copied across Regions. However, data on Amazon EBS volumes can be copied across Regions out of band, For example, you can run an instance in the Region with the source volume, run an instance in the destination Region with a new volume attached, and use rsync or some other file copy mechanism to copy data.

If I make service calls to the ec2.amazonaws.com service endpoint, where will my instances launch?

They will launch in the original Amazon EC2 `ec2.us-east-1.amazonaws.com` Region.

Can instances use group-based firewall rules across Regions?

No. Group-based firewall rules only work within a Region. If you need instances to communicate with each other across Regions, you should use CIDR based firewall rules. To simplify IP address management, you can use firewall rules in combination with Elastic IP addresses.

Note

Because inter-Region traffic crosses the public Internet, encrypt all sensitive data.

How do I use the command line tools with multiple Regions?

By default, the command line tools use the original `ec2.us-east-1.amazonaws.com` endpoint (the us-east-1 Region). For information about changing the Region, see Specifying the Region to Use (p. 325).

What is the cost for data transfer between Regions?

Data transferred from one Region to another is charged at both sides at the Internet data transfer rate.

Can I assume that my Availability Zone us-east-1a is the same location as someone else's Availability Zone us-east-1a?

No. Currently, we do not support cross-account proximity. Each account's Availability Zones are independent. For example, the `us-east-1a` Availability Zone for one account might be in a different location than for another account.

How can I make sure that I am in the same Availability Zone as another developer?

We do not currently support the ability to coordinate Availability Zones between AWS accounts. We are seeking customer feedback to understand the types of use cases for proximity control between accounts. We will use this feedback to determine how and when we might provide Availability Zone control between accounts.

Regional data transfer seems like such a small charge, why are you complicating my bill with this?

We anticipate that for most common use cases, Regional data transfer will only constitute a very small portion of your monthly usage charges. There are valid use cases that involve moving large amounts of data between Availability Zones. In these cases, the Regional data transfer can be a significant cost.

We try to enable as many use cases as possible while charging you only for what you use. Because of the large potential differences in the way developers could use Regional data transfer, we think it is appropriate to break this cost out rather than amortize it across other charges.

If I have two instances in different Availability Zones, how will I be charged for Regional data transfer?

Each instance is charged for its data in and data out. Therefore, if data is transferred between these two instances, it is charged out for the first instance and in for the second instance.

If I transfer data between Availability Zones using public IP addresses, will I be charged twice for Regional data transfer (once because it crosses Availability Zones, and once because I use public IP addresses)?

No. Regional data transfer rates apply if at least one of the following cases is true, but are only charged once for a given instance even if both are true:

- The other instance is in a different Availability Zone, regardless of which type of address is used
- Public or Elastic IP addresses are used, regardless of which zone the other instance is in

Why are my Amazon EC2 resources not visible in the EU Region or other Region?

Amazon EC2 Regions are isolated from each other. Resources such as SSH key pairs from a `CreateKeyPair` call, security groups, and AMIs, are not replicated between Regions. For more information, see Appendix A: Resources (p. 421).

Windows Instances FAQ

How can I mount or access a CD from the instance?

Select from the following:

- To create an ISO image out of the CD, upload it to your Amazon S3 bucket and download it to the instance. Then, use any standard ISO mounting tool to access it.
- To use Remote Desktop, specify the CD ROM drive letter from the **Local Resources** tab of the **Local Devices and Resources** page on the Remote Desktop client.

Monitoring, Errors, and Unexpected Behavior FAQ

How do I monitor my systems?

You can use DescribeInstances to check whether an instance appears to be running.

For more advanced monitoring, consider Amazon CloudWatch. Amazon CloudWatch runs a monitoring services that collects raw measurement data, such as CPUUtilization (percentage of Amazon EC2 compute units used by an instance) and DiskWriteBytes (number of bytes written in a minute). For more information, see Monitoring Your Instances and Volumes with CloudWatch (p. 348).

Why can't I "talk" to my instances?

There are a few common reasons for broken connectivity to your instance.

Amazon EC2 changes the state of your instance to `running` after your operating system starts booting. Depending on your AMI, there will be a delay before the instance is fully set up and functional.

If your instance has been running for several minutes, you verify you authorized the appropriate access to your host through the Amazon EC2 firewall. If you have launched your instances without specifying a security group, the `default` group is used. Permissions on the `default` group are very strict and disallow all access from the Internet and other groups. You will need to modify the permissions of your `default` group or set up a new group with appropriate permissions. For more information, see Security Group Concepts (p. 304)

If this doesn't solve your issue, make sure you authorized port 22 and try to open an SSH connection with verbose output. Use the man page for the exact syntax of your system, but the command is likely to be similar to **`ssh -vv root@[hostname]`**. This output is very useful if you are posting to the forum.

Why did my instance terminate immediately after launch?

Launch errors can be the result of an internal error during launch or a corrupt Amazon EC2 image. Internal errors are rare, as we actively test for and isolate suspect hosts. Consult the DescribeInstances operation for details on why your instance failed to launch.

Note

The `ec2-describe-instances` command line tool does not provide this information. Use the `-v` flag to read the detailed SOAP response and get detailed information.

You can also attempt to launch the image again. If this proves to be a persistent problem (especially with a shared image), post to the AWS forums.

I ran `shutdown` from within an SSH session, but my instance still shows up as running when I query it with `DescribeInstances`, and I can't shell into it.

To shut down an instance, use the `TerminateInstances` call (`ec2-terminate-instances`) on the command line. You can also use `shutdown -h`, but you must verify the instance shutdown using the `DescribeInstances` call.

Why are my instances stuck in a pending state (or a shutting-down state)?

This situation is rare and might be the result of a software error or misconfiguration.

We actively monitor for this; please contact us if it occurs.

Why do I get an "AuthFailure: User is not AMI creator" error when I try to register an image?

Make sure that you are using the correct user ID (AWS account ID) and certificate to create and upload the image. You must use the same ID and certificate to register the image with Amazon EC2.

Reserved Instances FAQs

What is a Reserved Instance?

Reserved Instances give you the option to make a low, one-time payment for each instance you want to reserve and in turn receive a significant discount on the hourly usage charge for that instance. For more information, see Reserved Instances (p. 224).

How is a Reserved Instance different than an On-Demand Instance?

Functionally, Reserved Instances and On-Demand instances are the same. They are launched and terminated in the same way, and they function identically once running. This makes it easy for you to seamlessly use both Reserved and On-Demand Instances without making any changes to your code. The only difference is that with a Reserved Instance, you pay a low, one-time payment and receive a lower usage rate to run the instance than with an On-Demand Instance.

How do I purchase and start up a Reserved Instance?

You purchase an EC2 Reserved Instance by calling the `PurchaseReservedInstancesOffering` API method. Launching a Reserved Instance is no different than launching an On-Demand Instance. You simply use the `RunInstances` call or launch an instance via the AWS Management Console. Amazon EC2 will optimally apply the cheapest rate that you are eligible for in the background.

How do I control which instances are billed at the Reserved Instance rate?

The `RunInstances` call does not distinguish between On-Demand and Reserved Instances. When computing your bill, our system automatically optimizes which instances are charged at the lower Reserved Instance rate to ensure you always pay the lowest amount.

How many Reserved Instances can I purchase?

You can purchase up to 20 Reserved Instances per Availability Zone each month with the EC2 APIs. If you need additional Reserved Instances, complete the Registration Form.

Can a Reserved Instance that I've bought for a particular instance type (i.e. High-CPU Extra Large Instance) be applied to a different instance type that I am running (i.e. Standard Large Instance)?

No. Each Reserved Instance is associated with a specific instance type, and can only be applied to that instance type for the duration of the Reserved Instance term.

Can I move a Reserved Instance from one Region or Availability Zone to another?

No. Each Reserved Instance is associated with a specific Region and Availability Zone, which is fixed for the lifetime of the Reserved Instance and cannot be changed.

Do I need to specify an Availability Zone when I launch my instances in order to take advantage of my Reserved Instances?

Yes. When you purchase a Reserved Instance you specify the Availability Zone in which you want to reserve that instance. In order to use that Reserved Instance, you need to ensure that you launch your instance in that same Availability Zone. Additionally, you can purchase a Reserved Instance in an Availability Zone where you already have a running instance, and the Reserved Instance will automatically get applied to that existing instance.

Can I cancel a Reserved Instance?

The one-time payment for a Reserved Instance is not refundable. However, you can choose not to run or entirely stop using your Reserved Instance at any time, at which point you will not incur any further usage charges.

What happens when my Reserved Instances term comes to an end?

Any instances that you have that are still running will continue to run, but will be charged at the standard On-Demand hourly rate.

When are Reserved Instances activated?

A Reserved Instance is activated once your one-time payment has successfully been authorized. You can follow the status of your Reserved Instance on the AWS Account Activity page.

Paid AMIs FAQ

Do you support paid AMIs that are backed by Amazon EBS?

Not at this time.

Note

You can still share AMIs without charging. Public and paid AMIs can be listed in the Resource Center.

How can I determine the paid AMIs that are available?

By describing images (`ec2-describe-images`) with the "-a" flag and looking for AMIs that have a product code. For example, if you run `ec2dim -a`, the result includes an AMI with the ID `ami-bd9d78d4`. This is our Demo Paid AMI with product code A79EC0DB.

How can I determine if a particular AMI is paid?

By describing the image (`ec2-describe-images`). An AMI is a paid AMI if a product code is returned. Example: run ec2dim ami-bd9d78d4, and the results include a product code (A79EC0DB).

Is there anything that prevents a paid AMI from being rebundled? How can this be restricted?

Paid AMIs are comparable to shared AMIs with regards to rebundling and trying to restrict rebundling. If you allow a user running the AMI to see all of its contents (e.g. by giving root access to the AMI), the user could rebundle these into their own AMI.

Why can't I query a particular AMI's attributes to see if the AMI is paid?

Only the owner of an AMI can query the AMI attributes. However, anyone can tell if an AMI is paid by describing images (ec2dim). An AMI is paid if a product code is returned. Example: run ec2dim -a amazon, and the AMI with ID ami-bd9d78d4 will be returned with a product code (A79EC0DB).

Who can use the `ec2-confirm-product-instance` command?

Only the owner of the AMI can use this command. Owners use this command with supported AMIs to determine if a supported instance with a given product code attached is up and running.

Will the product code be inherited by the rebundled AMI?

If your customer uses AWS tools to rebundle the AMI, the product code associated with the AMI is inherited by the rebundled AMI. When launching the rebundled AMI the customer is still billed for usage based on your price.

Note

This is a convenience feature and not a guarantee that the product code will always be attached to rebundled AMIs.

Note that the customer's workflow could bundle the AMI outside of Amazon EC2, or the customer could use modified versions of the AWS tools, preventing the product code from being inherited.

Will the kernel/RAM disk be inherited by the rebundled AMI?

If you rebundle an AMI, it inherits the kernel and RAM disk from the source AMI unless you specify a different kernel and RAM disk.

Note

This is a convenience feature and not a guarantee that the kernel/RAM disk will always be attached to rebundled AMIs.

I created my paid AMIs with one AWS account, but I want to sell them using a different AWS account. Can I transfer them?

No, you can't automatically transfer AMIs from one account to another. You would have to upload them again using the second AWS account and then register them with DevPay using that account. Alternately, you could leave the AMIs with the original account (the AMI owner account) and register them with DevPay using another AWS account (the product owner account). You could then use the AMI owner account to associate the product code with the AMIs. However, keep in mind that only the product owner (and not the AMI owner in this case) can use the `ec2-confirm-product-instance` command, which confirms that an instance is running an AMI associated with the product owner's product code.

How do I prevent someone from stripping the product code from my paid AMI?

If you do not provide root access to your AMI, it cannot be rebundled. If you provide root access, our tools attempt to preserve the product code.

To increase security, we recommend that you configure your application to check the instance metadata to verify that the product code is intact.

Kernels and RAM Disk FAQ

What are user selectable kernels?

Amazon EC2 provides user selectable kernels which enables you to select a kernel when bundling an AMI or launching an instance. User selectable kernels are useful for keeping your instances up to date with security fixes and updates, being able to use functionality provided by new distributions, and for using specialty applications that have unique timing requirements.

How do I find user selectable kernels?

Use **ec2-describe-images -o amazon --filter "image-type=kernel"**. This lists all public kernels that are currently available.

What type of dependencies do kernels have?

Kernels are most likely to require a RAM disk that contains required drivers (e.g., Xen drivers, video drivers, and so on). If you launch a kernel without a required RAM disk, it will not work properly.

How do I know a kernel/AMI combination will work together?

If you are concerned about whether the kernel/image combination will work well together, Amazon provides several AMIs that have tested combinations that you can use as a starting point for your AMIs or AMIs that you can use as a foundation for a public AMIs. If you require a certified kernel/ AMI combination, you can find them as paid AMIs through organizations such as RedHat. For more information, see Paying for AMIs (p. 104).

Can I use my own kernel?

Yes. For more information, see Enabling Your Own Linux Kernels (p. 61).

How do I know which kernels are compatible with PVGRUB?

For a list of compatible PVGRUB kernels, see Compatible PV-GRUB Kernels (p. 65). Some Linux distributions provide kernels that are not compatible with Amazon EC2. We are working with vendors to ensure that the most popular AMIs provide kernels that work with Amazon EC2, and we have tested a number of these AMIs and found them to be compatible with PVGRUB. Unfortunately, it is not possible to support every kernel that is or can be compiled. To avoid the situation in which a kernel does not work consistently or at all, we recommend that you use a known good kernel, select a non-PVGRUB AKI, or seek support from your AMI vendor.

Error Messages FAQ

Why do I get an "InsufficientInstanceCapacity" error when I try to launch an instance?

This error indicates that we do not currently have enough available capacity to service your request.

If you are requesting a large number of instances, there might not be enough server capacity to host them. You can try again later or specify a smaller number of instances.

Why do I get an "InstanceLimitExceeded" error when I try to launch an instance?

This error indicates you reached your concurrent running instance limit. For new AWS accounts, the limit is 20.

If you need additional capacity, please complete the form at http://aws.amazon.com/contact-us/ec2-request.

Why can't I retrieve my instance-specific data from within a running instance when querying http://169.254.169.254/latest/?

The Parameterized Launches feature is available to instances that were launched after the feature was released. If you launched your instance before this, the data will not be available. If you want to use this functionality, relaunch your instances.

If you still experience problems retrieving the data after relaunching your instance, check the following:

- Verify you are using the correct base URI (http://169.254.169.254/latest/)
- Verify you are using the correct URI for the data you are trying to retrieve. Depending on the data, a trailing '/' might be required
- Verify you specified launch data when launching your instances. If not, you will get a HTTP error response (404) when trying to retrieve the user data

Note

Instance metadata is always available, even if you do not specify it at instance launch.

Why do I get keep getting `"Request has expired"` errors?

To reduce the risk of replay attacks, our requests include a timestamp. This and the most important parts of the request are signed to ensure the message (including the timestamp) cannot be modified without detection.

If the difference between the timestamp in the request and the time on our servers is larger than 5 minutes, the request is too old (or too new) and an error is returned.

You need to ensure that your system clock is accurate and configured to use the correct time zone. For more information, go to NTP.

My instance is in a "running" state but shows 272 for its state code. What does that mean?

This typically indicates some sort of issue with the host running the instance. First try rebooting the instance (through the AWS Management Console, or with `ec2-reboot-instances` or `RebootInstances`). Caution: for Windows instances, this operation performs a hard reboot that might result in data corruption.

If the reboot doesn't work, post a message to the EC2 forums with the instance ID. Typically someone from the EC2 team can get your instance back to a normal state. You can search for "code 272" in the forums and find previous messages others have posted.

Miscellaneous FAQ

What runlevel do instances start in?
All Linux instances are started in runlevel 4, regardless of the instance configuration.

Are there any special requirements to use FTP?
The File Transfer Protocol (FTP) has a PORT command by which a client sends its address back to the server. The server then connects to the client at that address to send the file data. If the client looks up its own internal address and sends this to the server, the connection will fail. In this specific case, there are two solutions to the problem. First, configure the client to send its public IP address. Second, the client can use "passive FTP" which makes connections only to the server, rather than from the server to the client. In general, applications which encode local addresses and port numbers in data sent to external servers might have problems with NAT. Care must always be taken to send the public address, rather than the internal one.

We recommend using passive mode unless it is not supported by the FTP server.

Appendices

Topics

Appendix A: Resources

The following table describes which Amazon EC2 resources are global, Regional, or Availability Zone-based.

Resource	Type	Description
AWS Account	Global	You use the same AWS account in all Regions.
DevPay Product Codes	Global	You use the same DevPay product codes throughout all Regions.
Amazon EC2 System Identifiers	Regional	Includes the AMI ID, Instance ID, EBS Volume ID, EBS Snapshot ID, and so on.
Instances	Availability Zone	Instances are tied to Availability Zones. However, the instance ID is tied to the Region.
AMIs	Regional	AMIs are tied to the Region where its files are located within Amazon S3.
Security Groups	Regional	Security groups are not copied across Regions. Instances within the Region cannot communicate with instances outside the Region using group-based firewall rules. Traffic from instances in another Region is seen as WAN bandwidth.

Resource	Type	Description
SSH Key Pairs	Regional or Global	The SSH key pairs that you create with `ec2-add-keypair`, `CreateKeyPair`, or in the AWS Management Console work only in the Region where you create them. However, you can optionally create an RSA key pair with a third-party tool and upload the public key to AWS. That key pair works in all Regions. For more information, go to ec2-import-keypair in the *Amazon Elastic Compute Cloud Command Line Reference* or ImportKeyPair in the *Amazon Elastic Compute Cloud API Reference*.
User-Supplied Identifiers	Regional	Includes security group names, SSH key pair names, and so on. Although you can create the same names in multiple Regions, they have no relationship to each other.
Elastic IP Addresses	Regional	Elastic IP addresses are tied to a Region and cannot be mapped across Regions.
EBS Volumes	Availability Zone	An Amazon EBS volume must be located within the same Availability Zone as the instance to which it attaches.
EBS Snapshots	Regional	Snapshots are tied to Regions and can only be used for volumes within the same Region.

Appendix B: Metadata Categories

The data available to instances is categorized into metadata and user-supplied data. The following table lists the categories of metadata. For more information about metadata, see Using Instance Metadata (p. 96).

Data	Description	Version Introduced
`ami-id`	The AMI ID used to launch the instance.	1.0
`ami-launch-index`	The index of this instance in the reservation.	1.0
`ami-manifest-path`	The manifest path of the AMI with which the instance was launched.	1.0
`ancestor-ami-ids`	The AMI IDs of any instances that were rebundled to create this AMI.	2007-10-10
`block-device-mapping/`	Defines native device names to use when exposing virtual devices.	2007-10-10
`instance-action`	Notifies the instance that it should reboot in preparation for bundling. Possible values: `none` | `shutdown`,`bundle-pending`.	2008-09-01
`instance-id`	The ID of this instance.	1.0

Data	Description	Version Introduced
instance-type	The type of instance. For more information, see Instance Families and Types (p. 69).	2007-08-29
local-hostname	The local hostname of the instance.	2007-01-19
local-ipv4	The private IP address of the instance.	1.0
kernel-id	The ID of the kernel launched with this instance, if applicable.	2008-02-01
mac	The instance's MAC address.	2011-01-01
network/interfaces/macs/MAC/local-hostname	The instance's local host name.	2011-01-01
network/interfaces/macs/MAC/local-ipv4s	The private IP addresses associated with the instance.	2011-01-01
network/interfaces/macs/MAC/mac	The instance's Media Access Control (MAC) address.	2011-01-01
network/interfaces/macs/MAC/public-ipv4s	The elastic IP addresses associated with the instance.	2011-01-01
network/interfaces/macs/MAC/security-group-ids	The IDs of the security groups the instance belongs to. Returned for VPC instances only.	2011-01-01
network/interfaces/macs/MAC/subnet-id	The ID of the Amazon VPC subnet the instance is in. Returned for VPC instances only.	2011-01-01
network/interfaces/macs/MAC/subnet-ipv4-cidr-block	The CIDR block of the Amazon VPC subnet the instance is in. Returned for VPC instances only.	2011-01-01
network/interfaces/macs/MAC/vpc-id	The ID of the VPC the instance is in. Returned for VPC instances only.	2011-01-01
network/interfaces/macs/MAC/vpc-ipv4-cidr-block	The CIDR block of the VPC the instance is in. Returned for VPC instances only.	2011-01-01
placement/availability-zone	The Availability Zone in which the instance launched.	2008-02-01
product-codes	Product code associated with the instance, if any.	2007-03-01
public-hostname	The public hostname of the instance. Not returned for instances in a VPC (they have only private IP addresses).	2007-01-19
public-ipv4	The public IP address. If an elastic IP address is associated with the instance, the value returned is the elastic IP address.	2007-01-19
public-keys/	Public keys. Only available if supplied at instance launch time.	1.0
ramdisk-id	The ID of the RAM disk launched with this instance, if applicable.	2008-02-01
reservation-id	ID of the reservation.	1.0

Data	Description	Version Introduced
`security-groups`	The names of the security groups the instance is launched in. Only available if supplied at instance launch time. For instances in a VPC, returns the IDs of the security groups the instance is in. If you change the instance's group membership, the metadata updates accordingly.	1.0

You can provide user data when you launch an instance, or when the instance is in a stopped state (for EBS-backed instances). User-supplied data is treated as opaque data: what you give us is what you get back.

Note

- All instances launched together get the same user-supplied data. You can use the AMI launch index as an index into the data.
- User data is limited to 16K. This limit applies to the data in raw form, not base64 encoded form.
- The user data must be base64 encoded before being submitted to the API. The API command line tools perform the base64 encoding for you. The data is in base64 and is decoded before being presented to the instance.

Appendix C: Windows Configuration Service

Before bundling or creating a Windows instance, you can configure the instance using the EC2Config service. The EC2Config service sets up and initializes the instance during startup, prepares the service for bundling, and manages the event log.

There are three EC2Config files that you can modify: `Config.xml`, `BundleConfig.xml`, and `EventLogConfig.xml`.

Note

By default, the EC2Config service is installed on all Amazon EC2 public Windows AMIs (Program Files\Amazon\Ec2ConfigService\). The files discussed in the following sections are in the `Settings` subdirectory.

Config.xml File

This section describes the `Config.xml` file.

Config.xml File

- **Ec2SetPassword—**Generates a new password on instance launch.
  By default, Amazon EC2 disables this after the first launch. To continue generating random passwords, set this to `Enabled`.
- **Ec2SetComputerName—**When enabled, sets the hostname to the internal DNS name of the instance and reboots.

- **Ec2InitializeDrives—**Initializes and formats the instance stores during startup. For more information on instance storage, see Amazon EC2 Instance Storage (p. 151).
- **Ec2ConfigureRDP—**Sets up a self-signed certificate on the instance, so users can securely access the instance using Remote Desktop.
- **Ec2OutputRDPCert—**Copies the Remote Desktop certificate information to the console, so the user can verify it against the thumbprint.
- **Ec2EventLog—**Puts eventlog entries on the console based on the configuration of the eventlogconfig file.

BundleConfig.xml File

The `BundleConfig.xml` file controls how the EC2Config service prepares an instance for bundling. This includes configuring sysprep on the system, changing the state of the Ec2ConfigureRDP plugin, and shutting down the instance for bundling. To not use sysprep, change the value of `SetSysprep` to `No`. To not set the Remote Desktop Certificate, set the value of `SetRDPCertificate` to `No`.

EventLogConfig.xml File

This section describes the `EventLogConfig.xml` file.

EventLogConfig.xml File

- **Category—**Event log key to monitor.
  For more information, go to the Microsoft Web Site.
- **Errortype—**The type of error (i.e., Error, Warning, Information).
  For more information, go to the Microsoft Web Site.
- **AppName—**The event source or application that logged the event.
  For more information, go to the Microsoft Web Site.
- **NumEntries—**The number of events stored for this category.
- **LastMessageTime—**To prevent the same message from being pushed repeatedly, the service updates this every time it pushes a message.

Example

The following are examples of event log entries. The first entry pushes the last 3 errors from system category, regardless of the application that generated the LastMessage entry. The second entry pushes the last 3 error entries written by Ec2Config generated after LastMessageTime.

```
 <EventLogConfig>
  <Event>
    <Category>System</Category>
    <ErrorType>Error</ErrorType>
    <NumEntries>3</NumEntries>
    <LastMessageTime>2008-09-10T00:00:00.000Z</LastMessageTime>
    <AppName></AppName>
  </Event>
  <Event>
    <Category>Application</Category>
    <ErrorType>Error</ErrorType>
    <NumEntries>3</NumEntries>
    <LastMessageTime>2008-09-10T00:00:00.000Z</LastMessageTime>
    <AppName>Ec2Config</AppName>
  </Event>
</EventLogConfig>
```

Amazon EC2 Resources

The following table lists related resources that you'll find useful as you work with this service.

Resource	Description
Amazon Elastic Compute Cloud Getting Started Guide	Provides a quick tutorial of the service based on a simple use case. Examples and instructions are included.
Amazon Elastic Compute Cloud User Guide	Provides conceptual information about Amazon EC2 and describes how to use Amazon EC2 features using the AWS Management Console, command line tools, and Query API.
Amazon Elastic Compute Cloud API Reference	Contains a comprehensive description of the API actions, data types, and errors.
Amazon Elastic Compute Cloud Command Line Reference	Contains a comprehensive description of all the command line tools and their options.
Amazon EC2 Technical FAQ	Covers the top questions developers have asked about this product.
Amazon EC2 Release Notes	Give a high-level overview of the current release. They specifically note any new features, corrections, and known issues.
AWS Developer Resource Center	A central starting point to find documentation, code samples, release notes, and other information to help you build innovative applications with AWS.
AWS Management Console	The console lets you perform most of the functions of Amazon EC2 and other AWS products without programming.
Discussion Forums	A community-based forum for developers to discuss technical questions related to Amazon Web Services.
AWS Support Center	The home page for AWS Technical Support, including access to our Developer Forums, Technical FAQs, Service Status page, and AWS Premium Support (if you are subscribed to this program).

Resource	Description
AWS Premium Support Information	The primary web page for information about AWS Premium Support, a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.
Amazon EC2 Product Information	The primary web page for information about Amazon EC2.
Form for questions related to your AWS account: Contact Us	This form is *only* for account questions. For technical questions, use the Discussion Forums.
Terms of Use	Detailed information about the copyright and trademark usage at Amazon.com and other topics.

Glossary

Amazon machine image (AMI)
An Amazon Machine Image (AMI) is an encrypted machine image stored in Amazon S3. It contains all the information necessary to boot instances of your software.

Amazon EBS
A type of storage that enables you to create volumes that can be mounted as devices by Amazon EC2 instances. Amazon EBS volumes behave like raw unformatted external block devices. They have user supplied device names and provide a block device interface. You can load a file system on top of Amazon EBS volumes, or use them just as you would use a block device.

Amazon EBS-backed AMI
An instance launched from an AMI backed by Amazon EBS uses an Amazon EBS volume as its root device. See Amazon EBS.

Instance store-backed AMI
An instance launched from an Amazon S3backed AMI uses an instance store as its root device. See instance store.

Availability Zone
A distinct location within a Region that is engineered to be insulated from failures in other Availability Zones and provides inexpensive, low latency network connectivity to other Availability Zones in the same Region.

compute unit
An Amazon-generated measure that enables you to evaluate the CPU capacity of different Amazon EC2 instance types.

EBS
See Amazon EBS.

Elastic Block Store
See Amazon EBS.

elastic IP address
A static public IP address designed for dynamic cloud computing. Elastic IP addresses are associated with your account, not specific instances. Any elastic IP addresses that you associate with your account remain associated with your account until you explicitly release them. Unlike traditional static IP addresses, however, elastic IP addresses allow you to mask instance or Availability Zone failures by rapidly remapping your public IP addresses to any instance in your account.

ephemeral store
See *instance store*.

explicit launch permission
Launch permission granted to a specific AWS account.

filter
Criterion you specify to limit the results when you list or describe your EC2 resources.

gibibyte (GiB)

A contraction of giga binary byte, a gibibyte is 2^30 bytes or 1,073,741,824 bytes. A gigabyte is 10^9 or 1,000,000,000 bytes.

group

See security group.

image

See *Amazon machine image*.

instance

Once an AMI has been launched, the resulting running system is referred to as an instance. All instances based on the same AMI start out identical and any information on them is lost when the instances are terminated or fail.

instance store

Every instance includes a fixed amount of storage space on which you can store data. This is not designed to be a permanent storage solution. If you need a permanent storage system, use Amazon EBS.

instance type

A specification that defines the memory, CPU, storage capacity, and hourly cost for an instance. Some instance types are designed for standard applications, whereas others are designed for CPU-intensive applications, or memory-intensive applications, etc.

launch permission

AMI attribute allowing AWS accounts to launch an AMI

Linux

Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.

maximum price

The maximum price you will pay to launch one or more Spot Instances. If your maximum price exceeds the Spot Price and your restrictions are met, Amazon EC2 launches instances on your behalf.

paid AMI

An AMI that you sell to other Amazon EC2 users. For more information, refer to the *Amazon DevPay Developer Guide*.

private IP address

All Amazon EC2 instances are assigned two IP addresses at launch: a private address (RFC 1918) and a public address that are directly mapped to each other through Network Address Translation (NAT).

public AMI

An AMI that all AWS accounts have launch permissions for.

public data sets

Sets of large public data sets that can be seamlessly integrated into AWS cloud-based applications. Amazon stores the data sets at no charge to the community and, like with all AWS services, you pay only for the compute and storage you use for their your applications. These data sets currently include data from the Human Genome Project, the U.S. Census, Wikipedia, and other sources.

public IP address

All Amazon EC2 instances are assigned two IP addresses at launch: a private address (RFC 1918) and a public address that are directly mapped to each other through Network Address Translation (NAT).

region

A geographical area in which you can launch instances (e.g., US, EU).

reservation

A collection of instances started as part of the same launch request.

Reserved Instance

An additional Amazon EC2 pricing option. With Reserved Instances, you can make a low one-time payment for each instance to reserve and receive a significant discount on the hourly usage charge for that instance.

resource

A general term that refers to the objects you work with in Amazon EC2. This includes instances, images, Amazon EBS volumes, snapshots, etc.

security group

A security group is a named collection of access rules. These access rules specify which ingress (i.e., incoming) network traffic should be delivered to your instance. All other ingress traffic will be discarded.

shared AMI

AMIs that developers build and make available for other AWS developers to use.

Solaris

Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.

snapshot

Amazon EBS provides the ability to create snapshots or backups of your Amazon EBS volumes and store them in Amazon S3. You can use these snapshots as the starting point for new Amazon EBS volumes and to protect your data for long term durability.

Spot Instance

A type of instance that you can bid on to take advantage of unused Amazon EC2 capacity.

Spot Price

The current price for Spot Instances. If your Spot Instance request exceeds this price and your restrictions are met, Amazon EC2 launches instances on your behalf.

supported AMIs

These AMIs are similar to paid AMIs, except that you charge for software or a service that customers use with their own AMIs.

tag

Metadata of your choice (consisting of up to 10 key-value pairs) that you can optionally assign to EC2 resources.

tebibyte (TiB)

A contraction of tera binary byte, a tebibyte is 2^40 bytes or 1,099,511,627,776 bytes. A terabyte is 10^12 or 1,000,000,000,000 bytes.

UNIX

Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.

Windows

Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.

Document History

This documentation is associated with the 2011-12-01 release of Amazon EC2. This guide was last updated on 29 December 2011.

The following table describes the important changes since the last release of the Amazon EC2 documentation set.

Change	Description	Release Date
Elastic Network Interfaces (ENIs) for EC2 Instances in Amazon Virtual Private Cloud	Added new section about elastic network interfaces (ENIs) for EC2 instances in a VPC. For more information, see Elastic Network Interfaces (p. 366).	21 December 2011
New GRU Region and AKIs	Added information about the release of new AKIs for the SA-East-1 Region. This release deprecates the AKI version 1.01. The current AKI version 1.02 will continue to be backward compatible.	14 December 2011
New Offering Types for Amazon EC2 Reserved Instances	Starting with API version 2011-11-01, you can choose from a variety of Reserved Instance offerings that address your projected use of the instance: *Heavy Utilization*, *Medium Utilization*, and *Light Utilization*. See Reserved Instances (p. 224).	01 December 2011
Amazon EC2 Instance Status	You can now view additional details about the status of your instances, including scheduled events planned by AWS that might have an impact on your instances. These operational activities include instance reboots required to apply software updates or security patches, or instance retirements required where there are hardware issues. See Monitoring the Status of Your Instances (p. 355).	16 November 2011
Amazon EC2 Cluster Compute Instance Type	Added support for Cluster Compute Eight Extra Large (cc2.8xlarge).	14 November 2011
New PDX Region and AKIs	Added information about the release of new AKI's for the new US-West 2 Region.	8 November 2011

Change	Description	Release Date
Amazon EC2 Spot Instances in Amazon VPC	Added information about the support for Amazon EC2 Spot Instances in Amazon VPC. With this update, users will be able to launch Spot Instances in the Amazon Virtual Private Cloud (Amazon VPC). By launching Spot Instances in Amazon VPC, users of Spot Instances can enjoy all of the controls and advanced security options of Amazon VPC. For more information, see Launch Spot Instances in Amazon Virtual Private Cloud (p. 259).	11 October 2011
Updated Content for Amazon EC2 Spot Instances	Included a restructured and expanded *Spot Instances* chapter that includes new sections on Getting Started, Architecting for Interruptions, and Bidding Strategies. For more information, see Spot Instances (p. 237).	11 October 2011
Amazon Linux AMI Basics	Added information about the release of Amazon Linux AMI 2011.09. This update removes the beta tag from the Amazon Linux AMI, supports the ability to lock the repositories to a specific version, and provides for notification when updates are available to installed packages including security updates. For more information, see Amazon Linux AMI Basics (p. 55).	26 September 2011
Updated Content	Added information about how to connect to instances. For more information, see Connecting to Instances (p. 111). Moved *Appendix D: Connecting to a Linux/UNIX Instance from Windows Using PuTTY* to the Using Instances section. For more information, see Connecting to Linux/UNIX Instances from Windows Using PuTTY (p. 117).	16 September 2011
Simplified VM import process for users of the CLI tools	The VM Import process for CLI users is simplified with the enhanced functionality of `ec2-import-instance` and `ec2-import-volume`, which now will perform the upload of the images into Amazon EC2 after creating the import task. In addition, with the introduction of the `ec2-resume-import` command, users can restart an incomplete upload at the point the task stopped. For more information, see Importing Your Virtual Machine into Amazon EC2 (p. 218) in the *Amazon Elastic Compute Cloud User Guide*.	15 September 2011
Added support for VHD file format	VM Import can now import virtual machine image files in VHD format. The VHD file format is compatible with the Citrix Xen and Microsoft Hyper-V virtualization platforms. With this release, VM Import now supports RAW, VHD and VMDK (VMware ESX-compatible) image formats. For more information, see Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2 (p. 211) in the *Amazon Elastic Compute Cloud User Guide*.	24 August 2011
Added support for Microsoft Windows Server 2003 R2	VM Import now supports Windows Server 2003 (R2). With this release, VM Import supports all versions of Microsoft Windows Server supported by Amazon EC2.	24 August 2011
Added temporary security credentials section	New section describes how to pass temporary security credentials with Amazon EC2 API requests. For more information, see Using Temporary Security Credentials (p. 394).	03 August 2011

Change	Description	Release Date
Update to the Amazon EC2 VM Import Connector for VMware vCenter	Added information about the 1.1 version of the Amazon EC2 VM Import Connector for VMware vCenter virtual appliance (Connector). This update includes proxy support for Internet access, better error handling, improved task progress bar accuracy, and several bug fixes. For more information, see Using Instances of Your Virtual Machine in Amazon EC2 (p. 194).	27 June 2011
Enabling Linux AMI to run user-provided kernels	Added information about the AKI version change from 1.01 to 1.02. This version updates the PVGRUB to address launch failures associated with t1.micro Linux instances. For more information, go to Enabling Your Own Linux Kernels.	20 June 2011
Restructured User Guide with a new section on Using Storage	Implemented the following updates to the *Amazon EC2 User Guide*: • Added information about the primary data storage options supported by Amazon EC2. These storage options include Amazon EBS, Amazon S3, and Amazon EC2 instance stores. For more information, see Using Storage (p. 125). • Consolidated the storage-related information into the Using Storage (p. 125) section.	10 June 2011
Spot Instances Availability Zone Pricing Changes	Added information about the Spot Instances Availability Zone pricing feature. In this release, we've added new Availability Zone pricing options as part of the information returned when you query for Spot Instance requests and Spot Price history. These additions make it easier to determine the price required to launch a Spot Instance into a particular Availability Zone. For more information, see Spot Instances (p. 237).	26 May 2011
AWS Identity And Access Management	Added information about AWS Identity and Access Management (IAM), which enables users to specify which Amazon EC2 actions a user can use with Amazon EC2 resources in general. For more information, see Using AWS Identity and Access Management (p. 390).	26 April 2011
Amazon Linux AMI Guide	Added information about using Amazon Linux AMI. Amazon Linux AMI is supported and maintained Linux image provided by Amazon Web Services (AWS) for use on Amazon EC2. For more information, see Amazon Linux AMI Basics (p. 55).	26 April 2011
Enabling Linux AMI to run user-provided kernels	Added information about enabling a Linux AMI to use PVGRUB Amazon Kernel Image (AKI) to run a user-provided kernel. For more information, go to Enabling Your Own Linux Kernels.	26 April 2011
Updated Content	Updated information about connecting to instances. For more information, see Connecting to Instances (p. 111) and Connecting to Linux/UNIX Instances from Windows Using PuTTY (p. 117).	18 April 2011

Change	Description	Release Date
Dedicated Instances	Launched within your Amazon Virtual Private Cloud (Amazon VPC), Dedicated Instances are instances that are physically isolated at the host hardware level. Dedicated Instances let you take advantage of Amazon VPC and the AWS cloud, with benefits including on-demand elastic provisioning and pay only for what you use, while isolating your Amazon EC2 compute instances at the hardware level. For more information, go to Using EC2 Dedicated Instances Within Your VPC in the *Amazon Virtual Private Cloud User Guide*.	27 March 2011
Reserved Instances Updates to the AWS Management Console	Updates to the AWS Management Console now make it easier for users to view their Reserved Instances and purchase additional Reserved Instances, including Dedicated Reserved Instances. For more information, see Reserved Instances (p. 224).	27 March 2011
Support for Windows Server 2008 R2	Amazon EC2 now provides you with several pre-configured Windows Server 2008 R2 AMIs. All of these AMIs are immediately available for use in every Region and in most 64-bit instance types, excluding t1.micro and HPC families. The AMIs will support multiple languages. For more information, see Windows Instance Types (p. 72).	15 March 2011
New Amazon Linux reference AMI	Added information about the new Amazon Linux reference AMI, which replaces the CentOS reference AMI. Removed information about the CentOS reference AMI, including the section named Correcting Clock Drift for Cluster Instances on CentOS 5.4 AMI. For more information, see Reference AMI (p. 336).	15 March 2011
Updated Metadata Information	Updated the information about metadata to reflect changes in the 2011-01-01 release. For more information, see Using Instance Metadata (p. 96) and Appendix B: Metadata Categories (p. 422).	11 March 2011
Updated Security Group Information	Updated the section about security groups. For more information, see Using Security Groups (p. 304).	11 March 2011
Updated Amazon VPC Information	Updated the section about Amazon VPC. For more information, see Amazon Virtual Private Cloud (p. 364).	11 March 2011
Amazon EC2 VM Import Connector for VMware vCenter	Added information about the Amazon EC2 VM Import Connector for VMware vCenter virtual appliance (Connector). The Connector is a plug-in for VMware vCenter that integrates with VMware vSphere Client and provides a graphical user interface that you can use to import your VMware virtual machines to Amazon EC2. For more information, see Using Instances of Your Virtual Machine in Amazon EC2 (p. 194).	3 March 2011
New link	This service's endpoint information is now located in the Amazon Web Services General Reference. For more information, go to Regions and Endpoints in Amazon Web Services General Reference.	2 March 2011

Change	Description	Release Date
Force Volume Detachment	You can now use the AWS Management Console to force the detachment of an Amazon EBS volume from an instance. For more information, see Detaching an Amazon EBS Volume from an Instance (p. 145).	23 February 2011
Instance Termination Protection	You can now use the AWS Management Console to prevent an instance from being terminated. For more information, see Enabling Termination Protection for an Instance (p. 93).	23 February 2011
Correcting Clock Drift for Cluster Instances on CentOS 5.4 AMI	Added information about how to correct clock drift for cluster instances running on Amazon's CentOS 5.4 AMI.	25 January 2011
Restructured User Guide	Implemented the following updates to the Amazon EC2 User Guide: • Consolidated all developer guide information into this user guide. • Restructured and updated the information about creating AMIs. For more information, see Creating Your Own AMIs (p. 17). • Updated the introduction to the user guide. For more information, see Introduction to Amazon EC2 (p. 3)	14 January 2011
VM Import	Added information about VM Import, which allows you to import a virtual machine or volume into Amazon EC2. For more information, see Using the Command Line Tools to Import Your Virtual Machine to Amazon EC2 (p. 211).	15 December 2010
Basic Monitoring for Instances	Added information about Basic Monitoring for EC2 instances. For more information, see Monitoring Instances (p. 348).	12 December 2010
Updated Topic	Updated the topic about how to change an Amazon EBS-backed instance's root device volume to persist on instance termination. For more information, see Changing the Root Volume to Persist (p. 165).	20 November 2010
New Instance Type	Amazon EC2 offers cluster GPU instances (cg1.4xlarge) for high-performance computing (HPC) applications. For more information, see Cluster Instance Concepts (p. 335).	14 November 2010
Filters and Tags	Added information about listing, filtering, and tagging resources. For more information, see Listing and Filtering Your Resources (p. 270) and Using Tags (p. 275).	19 September 2010
Idempotent Instance Launch	Added information about ensuring idempotency when running instances. For more information, see Ensuring Idempotency (p. 91).	19 September 2010
New Instance Type	Amazon EC2 offers the t1.micro instance type for certain types of applications. For more information, see Micro Instances (p. 328).	8 September 2010

Change	Description	Release Date
Support for AWS Identity and Access Management	Amazon EC2 now integrates with AWS Identity and Access Management (IAM). For more information, see AWS Identity and Access Management (p. 10).	2 September 2010
New Instance Type	Amazon EC2 offers cluster compute instances for high-performance computing (HPC) applications. For more information, see Cluster Instance Concepts (p. 335).	12 July 2010
Amazon VPC IP Address Designation	Amazon VPC users can now specify the IP address to assign an instance launched in a VPC. For more information, see IP Addresses for Instances in a VPC (p. 290).	12 July 2010
Amazon CloudWatch Monitoring for Amazon EBS Volumes	Amazon CloudWatch monitoring is now automatically available for Amazon EBS volumes. For more information, see Monitoring Amazon EBS Volumes (p. 353).	14 June 2010
New Region	Amazon EC2 now supports the Asia Pacific (Singapore) Region. The new endpoint for requests to this Region is ec2.ap-southeast-1.amazonaws.com.	28 April 2010
Enhanced Information about Amazon EBS-Backed AMIs	We've added detailed information about Amazon EBS-backed AMIs. For more information, see the following sections: • Basics of Amazon EBS-Backed AMIs and Instances (p. 177) • Block Device Mapping (p. 167) • Using Amazon EBS-Backed AMIs and Instances (p. 177)	28 April 2010
New Instance Type	Amazon EC2 now supports a High-Memory Extra Large instance type. For more information, see Available Instance Types (p. 70).	22 February 2010
Reserved Instances with Windows	Amazon EC2 now supports Reserved Instances with Windows. For more information about Reserved Instances, see Reserved Instances (p. 224).	22 February 2010
Clarification about Spot Instances	Clarified that you can't stop and start Spot Instances that use an Amazon EBS root device. For more information about Spot Instances, see Spot Instances (p. 237).	1 February 2010
Command Line Tools Information	Information on how to get started with the command line tools is now available in the User Guide. For more information, see Getting Started with the Command Line Tools (p. 384).	26 January 2010

Index