
Amazon Elastic Compute Cloud

Microsoft Windows Guide

API Version 2012-10-01



Amazon Web Services

Amazon Elastic Compute Cloud: Microsoft Windows Guide

Amazon Web Services

Copyright © 2012 Amazon Web Services LLC or its affiliates. All rights reserved.

The following are trademarks or registered trademarks of Amazon: Amazon, Amazon.com, Amazon.com Design, Amazon DevPay, Amazon EC2, Amazon Web Services Design, AWS, CloudFront, EC2, Elastic Compute Cloud, Kindle, and Mechanical Turk. In addition, Amazon.com graphics, logos, page headers, button icons, scripts, and service names are trademarks, or trade dress of Amazon in the U.S. and/or other countries. Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon.

All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide**

Welcome	1
What is Amazon EC2?	3
Getting Started	8
Deploying a WordPress Blog	20
Amazon EC2 Infrastructure	34
Controlling Access: Security Groups and Credentials	40
Windows AMIs	43
Amazon Windows AMI Basics	43
Choosing a Windows AMI	46
Using EC2ConfigService	48
Creating Your Own Windows AMI	57
Creating an Amazon EBS-Backed Windows AMI	57
Creating an Instance Store-Backed Windows AMI	59
Shared Windows AMIs	62
Paid Windows AMIs	67
Setting Up a Windows HPC Cluster	71
Installing the Command Line Tools	81
Using PowerShell with the AWS SDK for .NET	87
Document History	99
Glossary	96
Index	100

Amazon Elastic Compute Cloud Microsoft Windows Guide

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable 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 the AWS Management Console, API actions, or command line tools and utilities. This guide will get you started using Amazon EC2 with the Windows Server operating system.

What's New?

Description	Relevant Sections
Configuring Windows Powershell to work with the .NET SDK and some code samples to help get you started.	Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET (p. 87)

How to Use this Guide

The following table lists the complete contents of this guide.

How Do I?	Relevant Topics
Get a brief overview of Amazon EC2	What is Amazon EC2? (p. 3)
Get up and running right away with Amazon EC2	Getting Started with Amazon EC2 Windows Instances (p. 8)
Control access to my Amazon EC2 instances	Controlling Access: Security Groups and Credentials (p. 40)
Learn the basic concepts for interacting with EC2	Amazon EC2 Infrastructure (p. 34)

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Additional Resources**

How Do I?	Relevant Topics
Set up a WordPress blog on an Amazon EC2 instance	Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 20)
Get started with the command line tools	Installing the Amazon EC2 Command Line Tools on Windows (p. 81)
Get detailed information on how to use Windows AMIs	Windows Amazon Machine Images (AMI) (p. 43)
Use Windows PowerShell with Amazon EC2	Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET (p. 87)
Set up an HPC Cluster using Amazon EC2	Setting Up a Windows HPC Cluster on Amazon EC2 (p. 71)

Additional Resources

Use the following table to find more information about Amazon EC2.

How Do I?	Relevant Sections
Get a general product overview and information about pricing	Amazon EC2 product page
Set up AWS web application hosting	Getting Started Guide AWS Web Application Hosting for Microsoft Windows
Get detailed information on how to use Amazon EC2	Using Amazon EC2
Find available libraries for programmatically accessing Amazon EC2	Available Libraries
Get started using the Query or SOAP API for Amazon EC2	Making API Requests

What is Amazon EC2?

Topics

- [Overview \(p. 3\)](#)
- [How Does Amazon EC2 Work? \(p. 3\)](#)
- [Differences Between Windows Server and an Amazon EC2 Windows Instance \(p. 4\)](#)
- [Designing Your Applications to Run on Amazon EC2 Windows Instances \(p. 6\)](#)
- [How You're Charged for Amazon EC2 \(p. 7\)](#)
- [Tips and Tricks for Windows Users \(p. 7\)](#)

Overview

Amazon Elastic Compute Cloud (Amazon EC2) is an Amazon Web Service (AWS) you can use to access servers, software, and storage resources across the Internet in a self-service manner. With Amazon EC2 you basically rent infrastructure comprising virtual servers and/or storage devices by the hour. You use these virtual servers to install, run, and process your applications at any time, for as long as you need, and for any legal purpose. After your requirement is fulfilled, you can either terminate the usage of the entire infrastructure or reduce the capacity and keep it in maintenance mode until you need to scale it up again. You pay for only what you use, and there is no minimum charge. With Amazon EC2 you do not need to invest in expensive hardware and have it sitting idle when your traffic or compute requirement is low.

How Does Amazon EC2 Work?

How does Amazon EC2 work with your Windows environment? Amazon EC2 provides templates known as Amazon Machine Images (AMIs) that contain pre-configured software such as an operating system, application server, and applications. You use these templates to launch your server instances, which are running copies of the AMI. After you launch your instance, you can use it just like a physical server. You can also launch multiple instances of an AMI, thus replicating the same configuration across each of the instances.

Amazon publishes a large selection of AMIs that contain software configurations specific to the Windows platform. In addition, members of the AWS developer community have published their own custom Windows AMIs. You might only need to use the Windows AMIs that Amazon or other reputable sources provide, and you can simply customize the resulting Windows instances (by running a script) to provide

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Differences Between Windows Server and an Amazon
EC2 Windows Instance**

the data or software you need each time you launch an instance. You can also create custom Windows AMIs with pre-installed and pre-configured applications. These AMIs can then be launched quickly and efficiently to become part of a live deployment. For detailed information on Amazon Windows AMIs, see [Windows Amazon Machine Images \(AMI\) \(p. 43\)](#) and for information on using AMIs and Instances, see [Using Amazon EC2](#).

Differences Between Windows Server and an Amazon EC2 Windows Instance

Amazon EC2 infrastructure is composed of virtual servers accessed via the Internet. These are commonly called *cloud servers*. By using Amazon EC2, you eliminate the need to buy and maintain expensive hardware. However, before you begin launching Amazon EC2 windows instances, you should be aware that the architecture of applications running on cloud servers can differ significantly from the architecture for traditional application models running on your hardware. Implementing applications on cloud servers requires a fundamental shift in your design process.

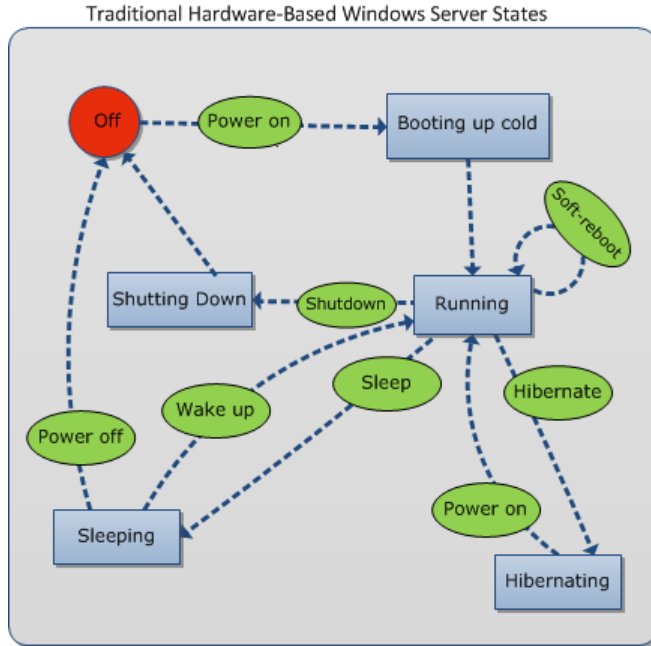
The following table describes some key differences between a Windows Server and an Amazon EC2 Windows instance.

Amazon EC2 Windows Instance	Windows Server
Designed to be deployed and terminated on demand.	Cannot be easily discarded after it is set up.
Resources and capacity are scalable.	Resources and capacity are physically limited.
You pay for the usage of the infrastructure. Billing stops as soon as the instance is terminated.	You pay for the infrastructure, whether you use it or not.
Does not occupy physical space and does not require regular maintenance.	Occupies physical space and has to be maintained on a regular basis.

After you launch your Amazon EC2 Windows instance, it behaves a lot like a traditional hardware-based Windows Server. For example, both a Windows Server and an Amazon EC2 instance can be used to run your web applications, conduct batch processing, or manage applications requiring large-scale computations. However, there are important differences between the server hardware model and the cloud compute model. The way an Amazon EC2 instance runs is not the same as the way a traditional Windows Server runs.

A traditional Windows Server goes through a number of phases from the time it is booted up through the time it is shut down, as the following diagram shows.

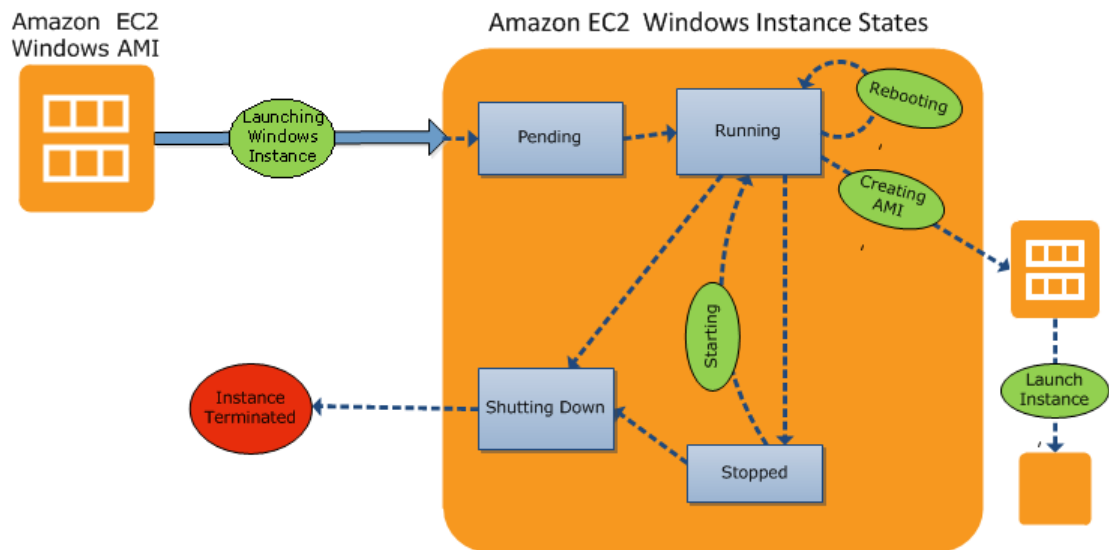
**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Differences Between Windows Server and an Amazon
EC2 Windows Instance**



A traditional hardware-based Windows Server starts with a push of a power button. This is called *cold booting*. When the server is up and running, you can choose to either keep the server running until it is time to shut it down, keep it in a sleep state for a specific duration of time, or keep it in a state of hibernation. The server is powered down during the hibernating and sleep states. These states can be brought back to the running state by powering the Windows Server on. However, after the server is powered off, the only way to get it up and running is by cold booting.

When your traditional Windows Server is powered off, all the resources associated with that server remain intact and in the state they were in when you switched it off. The information you stored on the hard drives persists and is ready to be accessed whenever needed.

An Amazon EC2 Windows instance has a number of similarities with the traditional hardware-based server, as you can see by comparing the following diagram with the previous diagram.



**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Designing Your Applications to Run on Amazon EC2
Windows Instances**

An Amazon EC2 Windows instance starts with the launch of the instance. Next, it briefly goes into the pending state while registration takes place. Then it moves to the running state, where instances can be rebooted, stopped, and then re-started. The Windows instance remains active until you initiate a shutdown process that terminates the instance. You can create an image of your instance and launch additional instances while your Amazon EC2 Windows instance is in the running state. This feature allows you to scale your infrastructure on demand.

Note

After an Amazon EC2 Windows instance is terminated, its infrastructure is no longer available to you. If you want to continue working with the same infrastructure, you have to launch a new instance.

You have control over Amazon EC2 instances and the resources that come with them, as long as they are in running or in stopped states. After the instance is terminated, you can choose to launch another instance of the same configuration, or a different configuration that meets a different requirement.

Designing Your Applications to Run on Amazon EC2 Windows Instances

It is extremely important that you consider the differences mentioned in the previous section when you design your applications to run on Amazon EC2 Windows instances.

Applications built for Amazon EC2 use the underlying computing infrastructure on an as-needed basis. They draw on necessary resources (such as storage and compute) on demand in order to perform a job, and relinquish the resources when done. In addition, they often dispose of themselves after the job is done. While in operation, the application scales up and down elastically based on resource requirements. An application running on an Amazon EC2 instance can terminate and recreate the various components at will in case of infrastructure failures.

When designing your Windows applications to run on Amazon EC2, you can plan for rapid deployment and rapid reduction of compute and storage resources, based on your changing needs.

When you run an Amazon EC2 Windows instance you don't need to provision the exact system package of hardware, software, and storage, the way you do with Windows Server. Instead, you can focus on using a variety of cloud resources to improve the scalability and overall performance of your Windows application.

With Amazon EC2, designing for failure and outages is an integral and crucial part of the architecture. As with any scalable and redundant system, architecture of your system should account for compute, network, and storage failures. You have to build mechanisms in your applications that can handle different kinds of failures. The key is to build a modular system with individual components that are not tightly coupled, can interact asynchronously, and treat each other as black boxes that are independently scalable. Thus, if one of your components fails or is busy, you can launch more instances of that component without breaking your current system.

Another key element to designing for failure is to distribute your application geographically. Replicating your application across geographically distributed regions improves high availability in your system. For more information, see [Using Regions and Availability Zones](#).

Amazon EC2 infrastructure is programmable and you can use scripts to automate the deployment process, to install and configure software and applications, and to bootstrap your virtual servers.

You should implement security in every layer of your application architecture running on an Amazon EC2 Windows instance. If you are concerned about storing sensitive and confidential data within your Amazon EC2 environment, you should encrypt the data before uploading it. On Amazon EC2, file encryption depends on the operating system.

How You're Charged for Amazon EC2

With Amazon 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

Important

You are billed starting when you launch the instance and charged for the time that the instance is running even if it remains idle.

- Data transfer
- Storage

For a complete list of charges and specific prices, go to the [Amazon EC2 pricing page](#). To calculate the cost of a sample provisioned environment, go to [AWS Economics Center](#) and use [Amazon EC2 Cost Comparison Calculator](#).

To see your bill, go to [AWS Account Activity page](#).

Tips and Tricks for Windows Users

This section contains some tips and tricks you can use while working with Amazon EC2.

- For the best experience using Internet Explorer, run the latest version.
- If you open an RDP session and are prompted for a domain (e.g., the user name displays as **IP-1024BB\Administrator**), in the **Remote Desktop** dialog box, click **Options**, and delete the text before **Administrator**.
- The easiest way to connect to an instance is from within the EC2 console: right-click the instance, and then click **Connect**.

An instance's public DNS name can change (for example, when the instance is rebooted). If you are using a cached RDP session and cannot connect to your instance, that might be the reason. When you connect using the console, the DNS public name is retrieved automatically so you connect using the current DNS public name.

- Don't launch an instance without a key pair. Without the key pair, you'll be unable to connect to your instance.
- After you launch and connect to an instance, do two things:
 1. Log into the instance and change your administrator password.
 2. While still logged in, create another user account with administrator permissions. This account can be useful if you forget the original administrator password account or if you have a problem using the original administrator account.

Getting Started with Amazon EC2 Windows Instances

To get started using Amazon Elastic Compute Cloud (Amazon EC2) Windows instances, complete the steps shown in the following table. You'll primarily use the AWS Management Console, a point-and-click web-based interface. You can also watch this short video to get started: [Getting Started with Amazon EC2: Launching a Windows Instance](#).

To get started with EC2

1. [Sign Up for EC2 \(p. 8\)](#)
2. [Launch a Windows Instance \(p. 9\)](#)
3. [Connect to Your Windows Instance \(p. 12\)](#)
4. [Create an Elastic IP Address \(p. 14\)](#)
5. [Create a CloudWatch Alarm to Monitor Your Instance \(p. 15\)](#)
6. [Clean Up \(p. 18\)](#)

Sign Up for EC2

When you create an AWS account, AWS automatically signs up the account for all AWS services, including Amazon EC2. You are charged only for the services that you use. If you already have an AWS account, skip to the next step. If you don't already have an AWS account, use the following procedure to create one.

To create an AWS account

1. Go to <http://aws.amazon.com>, and click **Sign Up Now**.
2. Follow the on-screen instructions.
Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad.

Launch a Windows Instance

Now that you're signed up for AWS, you're ready to start "computing" in the cloud. The first thing you'll do is to launch an instance using the AWS Management Console. An instance is a virtual server in the cloud. Amazon EC2 enables you to set up and configure the operating system and applications that run on your instance.

You can choose to launch one of the following instances:

- An instance within the Free Usage Tier. The Free Usage Tier enables you to launch and use an Amazon EC2 Micro instance free for 12 months. For more information about the Free Usage Tier, see the [AWS Free Usage Tier product page](#) and [Getting Started with AWS Free Usage Tier](#).
- An regular instance (not within the Free Usage Tier). You'll incur the standard Amazon EC2 usage fees for the instance until this tutorial shows you how to terminate it in the last step. The total charges to complete this tutorial are minimal (typically less than a few dollars). For more information about Amazon EC2 usage rates, see the [Amazon EC2 product page](#).

To launch an instance

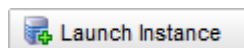
1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.

Use the email address and password that you specified when signing up for Amazon EC2.

2. From the navigation bar, select the Region for the instance. For this tutorial, you can use the default Region. Otherwise, this choice is important because some EC2 resources can be shared between Regions, while others can't. For example, if you'd like to connect your instance to an existing EBS volume, you must launch the instance in the same Region as the volume.



3. From the Amazon EC2 console dashboard, click **Launch Instance**.



The **Create a New Instance** page includes these ways to launch an instance:

Amazon Elastic Compute Cloud Microsoft Windows Guide

Launch a Windows Instance

- The **Classic Wizard** offers you more granular control and advanced settings for configuring your instance.
- The **Quick Launch Wizard** automatically configures many selections for you, so that you can get started quickly.

This tutorial guides you through the **Quick Launch Wizard**.

4. On the **Create a New Instance** page, click **Quick Launch Wizard**.
5. Optional: In **Name Your Instance**, enter an instance name that has meaning for you.
6. Under **Choose a Key Pair**, choose from any existing key pairs that you have created or create a new key pair. For this example, we'll create a key pair:

Important

Do not select the **None** option. If you launch your instance without a key pair, you can't connect to it.

- a. Click **Create New**.
 - b. Type a name for your key pair and then click **Download**. You'll need the contents of the private key to connect to your instance after you launch it. Amazon Web Services doesn't have the private portion of a key pair.
 - c. Save your private key in a safe place on your computer. Note the location because you'll need the key to connect to your instance.
7. Under **Choose a Launch Configuration**, the Quick Launch Wizard displays a list of basic configurations called Amazon Machine Images (AMI) that you can choose from to launch your instance. An AMI contains everything needed to create a new instance of a server, for example, a web server or a database server. In this tutorial, we'll use Microsoft Windows Server 2008 with a 64-bit operating system. If the configuration is marked with a star, this indicates that it's within the [Free Usage Tier](#).

Important

If you launch a regular instance, you're billed from the time that you launch the instance until the instance is no longer running, even if it remains idle.

Create a New Instance Cancel X

Select an option below:

- Classic Wizard**
Launch an On-Demand or Spot instance using the classic wizard with fine-grained control over how it is launched.
- Quick Launch Wizard**
Launch an On-Demand instance using an editable, default configuration so that you can get started in the cloud as quickly as possible.
- AWS Marketplace**
AWS Marketplace is an online store where you can find and buy software that runs on AWS. Launch with 1-Click and pay by the hour.

Name Your Instance: Pick a meaningful name, e.g. Web Server

Choose a Key Pair:
Public/private key pairs allow you to securely connect to your instance after it launches.

Select Existing Create New None

Name: Download
Please note that you need to download the key pair before you can continue.

Choose a Launch Configuration:

Microsoft Windows Server 2008 Base Microsoft Windows 2008 R1 SP2 Datacenter edition.	64 bit <input checked="" type="radio"/> 32 bit <input type="radio"/>	Free tier eligible
Microsoft Windows Server 2008 R2 Base Microsoft Windows 2008 R2 SP1 Datacenter edition and 64-bit architecture.	64 bit <input checked="" type="radio"/> 32 bit <input type="radio"/>	Free tier eligible
Microsoft Windows Server 2008 R2 with SQL Server Express and IIS Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQLServer 2008 Express, Internet Information Services 7, ASP.NET 3.5.	64 bit <input checked="" type="radio"/> 32 bit <input type="radio"/>	Free tier eligible
Microsoft Windows Server 2008 R2 with SQL Server Web Microsoft Windows Server 2008 R2 SP1 Datacenter, 64-bit architecture, Microsoft SQL Server 2008 R2 Web Edition.	64 bit <input checked="" type="radio"/> 32 bit <input type="radio"/>	
Cluster Instances HVM SUSE Linux Enterprise 11 SUSE Linux Enterprise Server 11 Service Pack 2, 64-bit architecture, 64 bit <input checked="" type="radio"/> 32 bit <input type="radio"/> and HVM based virtualization for use with Amazon EC2 Cluster Compute and Cluster GPU		

Note: You can customize your settings in the next step. Continue >

[Submit Feedback](#) [Getting Started Guide](#)

8. Click **Continue** to view and customize the settings for your instance.

Amazon Elastic Compute Cloud Microsoft Windows Guide Launch a Windows Instance

- Under **Security Details**, in **Security Group**, you see the security group that is selected for you by the wizard.

A security group defines firewall rules for your instances. These rules specify which incoming network traffic is delivered to your instance. All other traffic is ignored.

If you're new to Amazon EC2 and haven't set up any security groups yet, AWS defines a default security group for you. The name and description for the group is quicklaunch-x where x is a number associated with your quicklaunch group. The first security group you create using the Quick Launch Wizard is named quicklaunch-1. You can change the name and description using the **Edit details** button. The group already has basic firewall rules that enable you to connect to the type of instance you choose. For a Windows instance, you connect through Remote Desktop Protocol (RDP) on port 3389. The quicklaunch-x security group automatically allows RDP traffic on port 3389.

If you have used Amazon EC2 before, the wizard looks for an existing security group for the type of instance you're creating.

Caution

The quicklaunch-x security group authorizes all IP addresses to access your instance over the specified ports (for example, RDP). This is acceptable for the short exercise in this tutorial, but it's unsafe for production environments. In production, you'll authorize only a specific IP address or range of IP addresses to access your instance.

Create a New Instance Cancel X

Microsoft Windows Server 2008 Base (ami-c941efa0)
Platform: Windows Microsoft Windows 2008 R1 SP2 Datacenter edition.
Architecture: x86_64

Please review your settings and click **Launch** to finish or **Edit details** to make changes.

Instance Details

Name: GSG Tutorial	Type: t1.micro
Detailed Monitoring: No	Availability Zone: No preference
Shutdown Behaviour: Stop	Termination Protection: No
Launch into a VPC: No	

Security Details

Key Pair: GSG_Keypair	Security Group: quicklaunch-1
-----------------------	-------------------------------

Advanced Details

Kernel ID: Default	Ramdisk ID: Default
User Data:	IAM Role:

[Go Back](#) [Edit details](#) [Launch](#)

- Review your settings, and click **Launch** to launch the instance.
- A confirmation page lets you know that your instance is launching. Click **Close** to close the confirmation page and return to the Amazon EC2 console.
- In the **Navigation** pane, click **Instances** to view the status of your instance. It takes a short time for an instance to launch. The instance's status is `pending` while it's launching.



After the instance is launched, its status changes to `running`.



Amazon Elastic Compute Cloud Microsoft Windows Guide Connect to Your Windows Instance

13. Record the public DNS name for your instance because you'll need it for the next step. You can get this information by selecting the instance, which displays its details (including the public DNS name) in the lower pane. You can also click **Show/Hide** in the top right corner of the console and select **Public DNS** to display the **Public DNS** column shown in the previous step.
14. (Optional) After your instance is launched, you can view the quicklaunch-x security group rules.
 - a. On the Amazon EC2 console, under **Network and Security**, click **Security Groups**.
 - b. Click the quicklaunch-1 security group to view the security rules created by the Quick Launch Wizard.

The screenshot shows the Amazon EC2 console interface for the security group 'quicklaunch-1'. The 'Inbound' tab is selected, and a 'Create a new rule' dialog is open. The rule is a 'Custom TCP rule' with a port range of 3389 (RDP) and a source of 0.0.0.0/0. An 'Add Rule' button is visible in the dialog. To the right, a table lists the existing rules:

TCP	Port (Service)	Source
	3389 (RDP)	0.0.0.0/0

As you can see, the security group contains one rule that authorizes RDP traffic from any IP source to port 3389. If you launch a Windows instance running IIS and SQL, the Quick Launch Wizard creates a security group that authorizes traffic to port 80 for HTTP (for IIS) and port 1433 for MS SQL, as shown in the following figure.

The screenshot shows the Amazon EC2 console interface for the security group 'quicklaunch-1'. The 'Inbound' tab is selected, and a 'Create a new rule' dialog is open. The rule is a 'Custom TCP rule' with a port range of 0.0.0.0/0 and a source of 0.0.0.0/0. An 'Add Rule' button is visible in the dialog. To the right, a table lists the existing rules:

TCP	Port (Service)	Source
	3389 (RDP)	0.0.0.0/0
	1433 (MS SQL)	0.0.0.0/0
	80 (HTTP)	0.0.0.0/0

Connect to Your Windows Instance

To connect to a Windows instance, you must retrieve the initial administrator password, and then specify this password with Remote Desktop. You'll need the private key file that you created when you launched the instance (for example, `GSG_Keypair.pem`).

To connect to your Windows instance

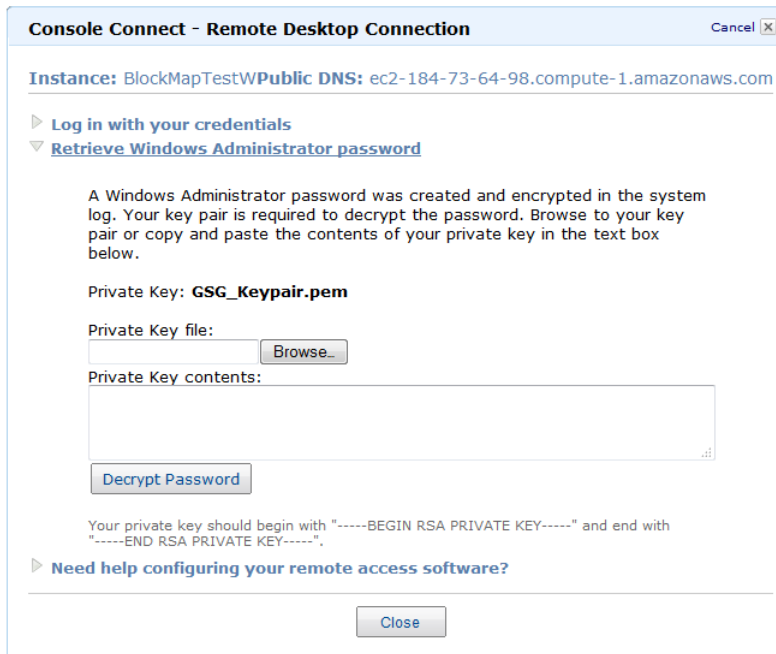
1. Before you try to connect, ensure that your Amazon EC2 instance accepts incoming RDP traffic (usually on port 3389). For more information, see [Authorizing Network Access to Your Instances](#).

Amazon Elastic Compute Cloud Microsoft Windows Guide Connect to Your Windows Instance

- Windows computers include an RDP client by default. You can check for an RDP client by typing **mstsc** at the Command Prompt window. If your computer doesn't recognize this command, go to the [Microsoft Windows home page](#) and search for the download for Remote Desktop Connection. For Mac OS X, you can use [Microsoft's Remote Desktop Client](#). For Linux/UNIX, you can use [rdesktop](#).
- In the Amazon EC2 console, right-click the instance that you created and click **Connect**.
- Click **Retrieve Password** (it might take a few minutes after the instance is launched before the password is available).



- Click **Browse** and navigate to the private key file you created when you launched the instance. Select the file and click **OK** to copy the entire contents of the file into the **Private Key contents** box.



Amazon Elastic Compute Cloud Microsoft Windows Guide Create an Elastic IP Address

6. Click **Decrypt Password**. The console displays the default administrator password for the instance in the **Console Connect** dialog box, replacing the link to **Retrieve Password** shown previously with the actual password.
7. Record the default administrator password, or copy it to the clipboard. You need this password to connect to the instance.
8. Click **Download shortcut file**. Your browser prompts you to either open or save the .rdp file. Either option is fine. When you have finished, you can click **Close** to dismiss the **Console Connect** dialog box.
9. If you opened the .rdp file, you'll see the **Remote Desktop Connection** dialog box. If you saved the .rdp file, navigate to your downloads directory, and double-click the .rdp file to display the dialog box. You may get a warning that the publisher of the remote connection is unknown. Click **Connect** to connect to your instance. You may get a warning that the security certificate could not be authenticated. Click **Yes** to continue.
10. Log in to the instance as prompted, using `Administrator` as the user name and the default administrator password that you recorded or copied in step 7.

We recommend that you do the following:

- Change the Administrator password from the default value. You change the password while logged on to the instance itself, just as you would on any other Windows Server.
- Create another user account with administrator privileges on the instance. Another account with administrator privileges is a safeguard if you forget the Administrator password or have a problem with the Administrator account.

You can work with your instance the same way you would work with any Windows server. For example, you can transfer files between an Amazon EC2 Windows instance and your local Windows computer using 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.

Create an Elastic IP Address

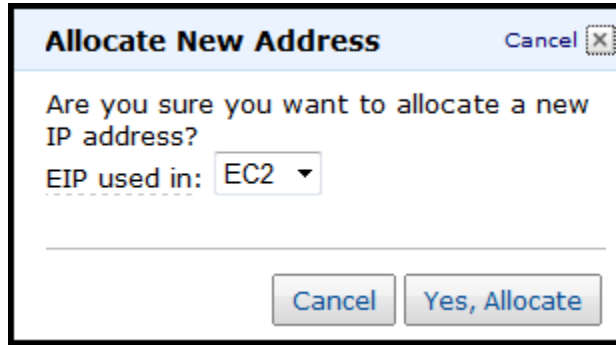
By default, all Amazon EC2 instances are assigned two IP addresses at launch: a private (RFC 1918) address and a public address that is mapped to the private IP address through network address translation (NAT).

To connect to your instance, you use the public DNS name associated with the public IP address. However, this name is not static and can change, for example when an instance reboots. If you want a persistent address to connect to, use an Elastic IP address.

Elastic IP addresses are static IP addresses designed for dynamic cloud computing. Additionally, 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.

To connect to your Windows instance

1. In the Amazon EC2 console navigation pane, click **Elastic IPs**.
2. Click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, click **Yes, Allocate**.



4. Select the Elastic IP address you created, and then click **Associate Address**.
5. In the **Associate Address** dialog box, in the **Instance** drop-down list, select your instance and then click **Yes, Associate**.

Create a CloudWatch Alarm to Monitor Your Instance

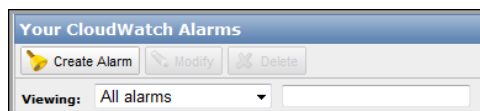
With Amazon CloudWatch, you can monitor various aspects of your instance and set up alarms based on criteria you choose. For example, you could configure an alarm to send you an email when an instance's CPU exceeds 70 percent.

Because you've just launched your instance, it is unlikely that the CPU will exceed this threshold, so instead, set a CloudWatch alarm to send you an e-mail when your instance's CPU is *lower than* 70 percent for five minutes.

The **Create Alarm Wizard** steps you through the process of creating an alarm.

To open the Create Alarm Wizard

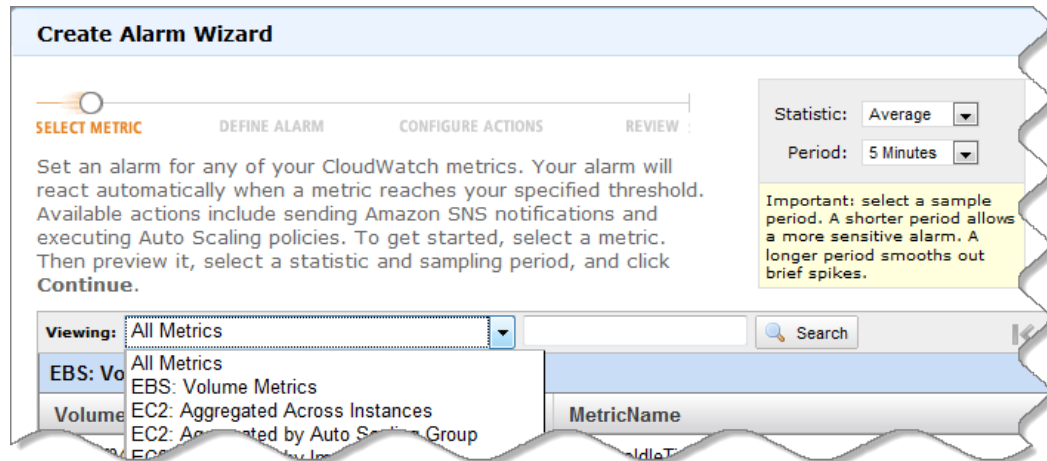
1. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. Click **Alarms** in the **Navigation** pane.
3. On the **CloudWatch Alarms** page, click **Create Alarm**.



4. The **SELECT METRIC** page of the **Create Alarm Wizard** opens.

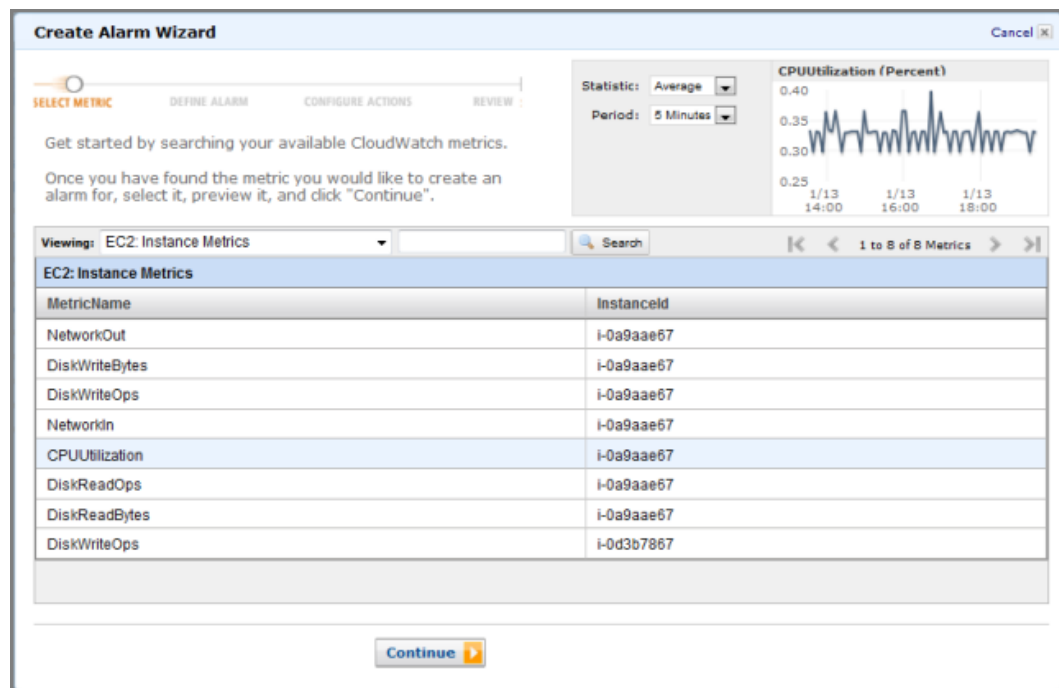
Amazon Elastic Compute Cloud Microsoft Windows Guide

Create a CloudWatch Alarm to Monitor Your Instance



To select a metric for your alarm

1. In the **SELECT METRIC** page of the **Create Alarm Wizard**, select **EC2: Instance Metrics** from the **Viewing** drop-down list.
The metrics available for individual instances appear in the **EC2 Instance Metrics** pane.
2. Select a row that contains **CPUUtilization** for a specific instance ID.
A graph showing average CPUUtilization for a single instance appears in the at the upper-right in the **SELECT METRICS** page.
3. Select **Average** from the **Statistic** drop-down list.
4. Select a period from the **Period** drop-down list, for example: **5 minutes**.
5. Click **Continue**.



6. The **DEFINE ALARM** page of the **Create Alarm Wizard** opens.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Create a CloudWatch Alarm to Monitor Your Instance**

To define the alarm name, description, and threshold

1. On the **DEFINE ALARM** page, in the **Name** field, enter the name of the alarm, for example: **myTestAlarm**.
2. In the **Description** field, enter a description of the alarm, for example: **CPU usage is lower than 70 percent**.
3. Select **<** in the **Define Alarm Threshold** drop-down list.
4. Enter **70** in the first **Define Alarm Threshold** field and **5** in the second field.
A graphical representation of the threshold appears on the page.
5. Click **Continue**.

Create Alarm Wizard Cancel X

SELECT METRIC DEFINE ALARM CONFIGURE ACTIONS REVIEW

Provide the details and threshold for your alarm. Use the graph below to help set the appropriate threshold.

Identify Your Alarm
Assign your alarm a name and description.

Name:

Description:

Define Alarm Threshold
Alarms have three states: ALARM, OK, and INSUFFICIENT DATA. The state of your alarm changes according to a threshold you specify. First, define the criterion for entering the ALARM state. Later, you can specify an action to be taken when your alarm enters any of the three states.

This alarm will enter the ALARM state when CPUUtilization is **<** for minutes.

CPU Utilization (Percent)

Time	CPU Utilization (%)
12/10 19:00	~10
12/10 20:00	~10
12/10 21:00	~10
12/10 22:00	~10
12/10 23:00	~10
12/11 00:00	~10

6. The **CONFIGURE ACTIONS** page of the **Create Alarm Wizard** opens.

Create Alarm Wizard Cancel X

SELECT METRIC DEFINE ALARM CONFIGURE ACTIONS REVIEW

Define what actions are taken when your 'myHighCpuAlarm' alarm changes.

Define Your Actions
Actions define what steps you want to automate when the alarm state changes. For example, you can send a message using email via the [Simple Notification Service \(SNS\)](#). You can also execute an [Auto Scaling Policy](#), if you have one configured ([learn about policies](#)).

Alarm State	Action Type	Action
ALARM	Send Notification	Topic: <input type="text" value="Select or create email topic"/> <input type="button" value="ADD ACTION"/>

To configure an email action for an alarm

1. On the **CONFIGURE ACTIONS** page, select **ALARM** from the **Alarm State** drop-down list.
2. Select **Create Email Topic** from the **Topic** drop-down list.
Two new fields named **Topic** and **Emails** replace the **Topic** drop-down list.
3. In the **Topic** field, enter a descriptive name for the Amazon Simple Notification Service (Amazon SNS) topic, for example: **myTestAlarm**.
4. In the **Emails** field, enter a comma-separated list of email addresses to be notified when the alarm changes to the **ALARM** state.

Alarm State	Action Type	Action
ALARM	Send Notification	Topic: <input type="text"/> Emails: <input type="text"/> <input type="button" value="ADD ACTION"/>
<small>A topic is a communication channel that can be reused across Send Notification actions. Please enter a list of comma-separated email addresses for the topic.</small>		

5. Click **ADD ACTION**.
The action is saved and the **ADD ACTION** button becomes a **REMOVE** button.
6. Click **Continue**.
7. The **REVIEW** page of the **Create Alarm Wizard** opens.

Now that you have defined the alarm and configured the alarm's actions, you are ready to review the settings and create the alarm.

To review the alarm settings and create the alarm

1. Review the alarm settings presented in the **REVIEW** page of the **Create Alarm Wizard**.
You can make changes to the settings with the **Edit Definition**, **Edit Metric**, or **Edit Actions** links.
2. Click **Create Alarm** to complete the alarm creation process.
A confirmation window opens.
3. Click **Close**.
Your alarm is created. A notification email is sent to the email address you provided with a link to an opt-in confirmation page for your notification. After you opt in, you will receive an email when your instance has been running for more than 5 minutes at less than 70 percent CPU utilization.

Clean Up

Now that you've completed these steps you can do any of the following:

- Keep using the instance and customize it to your needs.

Important

Remember, as soon as your instance starts to boot, you're billed for each hour or partial hour that you keep the instance running (even if the instance is idle).

- Try creating a WordPress blog.
- Clean up and terminate the instance.

When you've decided that you no longer need the instance, you need to do three things:

1. Delete the Amazon CloudWatch alarm.
2. Dissociate the Elastic IP address from your instance and release it (if you created an Elastic IP address)

Important

If you don't release the Elastic IP address, you are charged for not using it.

3. Terminate your instance.

To delete your CloudWatch alarm

1. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the **Navigation** pane, click **Alarms**.
3. Select the alarm you created, right-click, and then click **Delete**.

To disassociate and release an Elastic IP address

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the **Navigation** pane, click **Elastic IPs**.
3. Select your Elastic IP address, right-click and then click **Disassociate**.
4. Click **Yes, Disassociate**.
5. Right-click your Elastic IP address again, and then click **Release**.

Important

If you don't release the Elastic IP address, you are charged for not using it.

6. Click **Yes, Release**.

To terminate an instance

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Locate the instance you want to terminate in your list of instances on the **Instances** page.
3. Right-click the instance, and then click **Terminate**.
4. Click **Yes, Terminate** when prompted for confirmation.
Amazon EC2 begins terminating the instance. If you launched an instance in the Free Usage Tier, there are no charges. If you launched an instance that is not within the [Free Usage Tier](#), you'll stop incurring charges for that instance as soon as the instance status changes to `shutting down` or `terminated`.

Deploying a WordPress Blog on Your Amazon EC2 Instance

This section walks you through the process of creating and deploying a WordPress website on your Amazon EC2 Windows instance.

Process for Deploying a WordPress Blog

Task 1: Installing the Microsoft Web Deployment Tool (p. 20)
Task 2: Installing WordPress (p. 24)
Task 3: Configuring Your WordPress Site (p. 32)

Prerequisites

Before you get started, be sure you've done the following:

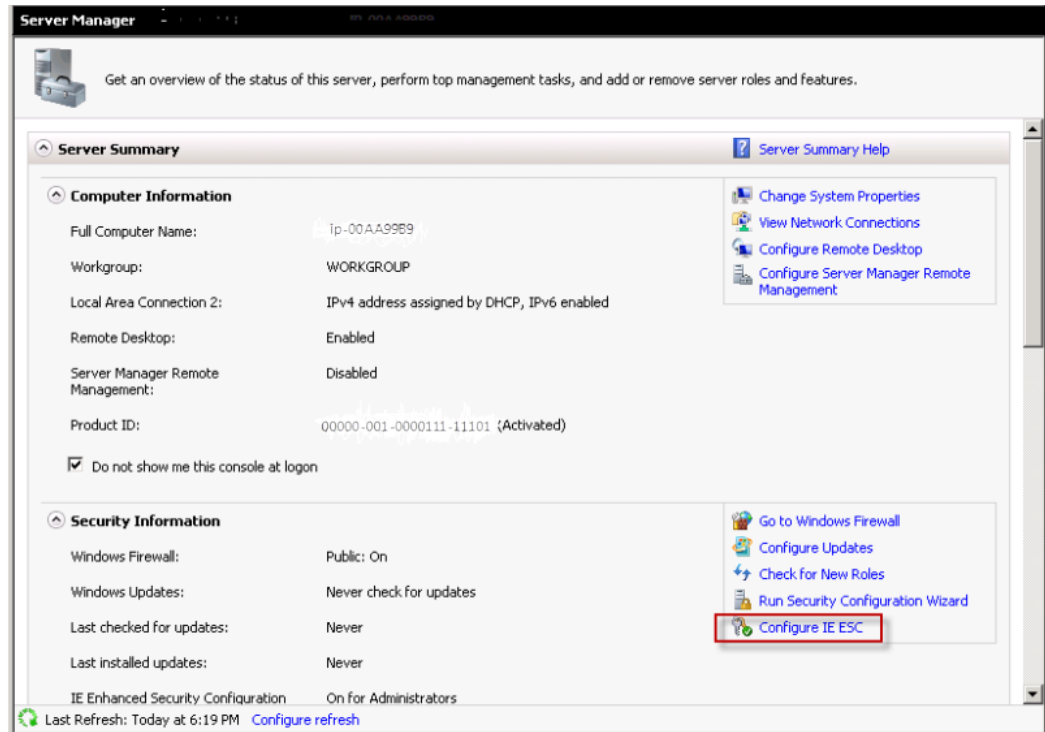
- Launched an Amazon EC2 instance from an AMI that has Windows Server 2008 R2 and Internet Information Services (IIS) pre-installed. For information on launching an Amazon EC2 instance, see [Getting Started with Amazon EC2 Windows Instances \(p. 8\)](#).
- Checked to ensure that the security group in which you're launching your Amazon EC2 instance has port 80 open for inbound traffic. If port 80 is not open, the WordPress site can't be accessed from outside the instance.
- Connected to your Amazon EC2 instance.

Task 1: Installing the Microsoft Web Deployment Tool

1. Verify you've met the conditions in [Prerequisites \(p. 20\)](#).
2. Configure Server Manager Settings.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide**
Task 1: Installing the Microsoft Web Deployment Tool

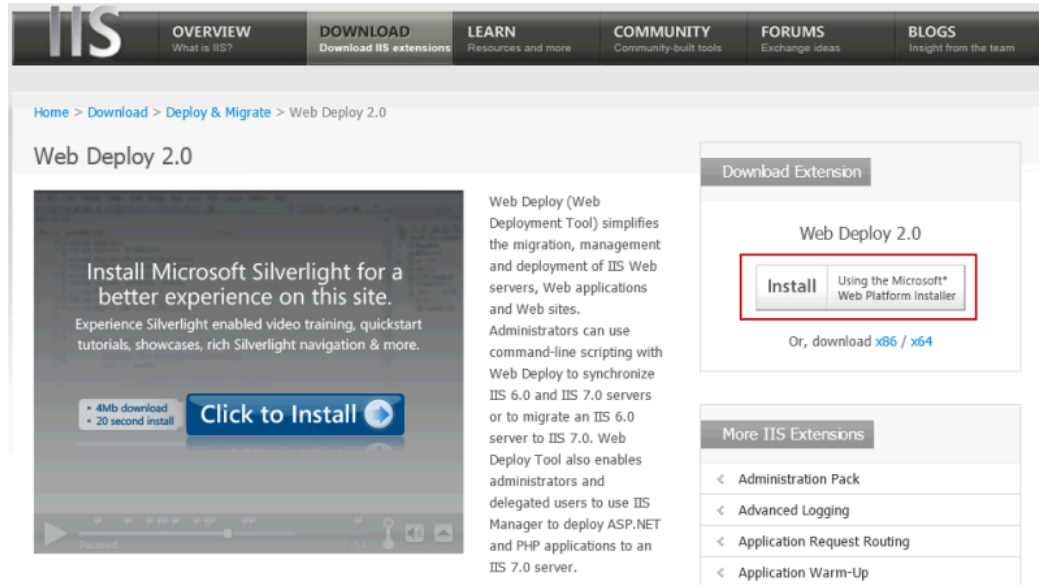
- a. In your Amazon EC2 instance, click **Start**.
- b. Click **Administrative Tools**.
- c. Select **Server Manager**.
- d. In the **Server Manager** box, in the **Security Information** panel, click **Configure IE ESC**.



- e. In the **Internet Explorer Enhanced Security Configuration** box, under **Administrators**, select the **Off** button.
 - f. Click **OK** to close the **Server Manager** box.
3. In the EC2 instance, in the **Internet Explorer** web address field type **http://www.iis.net/download/webdeploy**.
 4. In the **Download Extension** box, click **Install** to install the latest version of Web Deploy.

Amazon Elastic Compute Cloud Microsoft Windows Guide

Task 1: Installing the Microsoft Web Deployment Tool



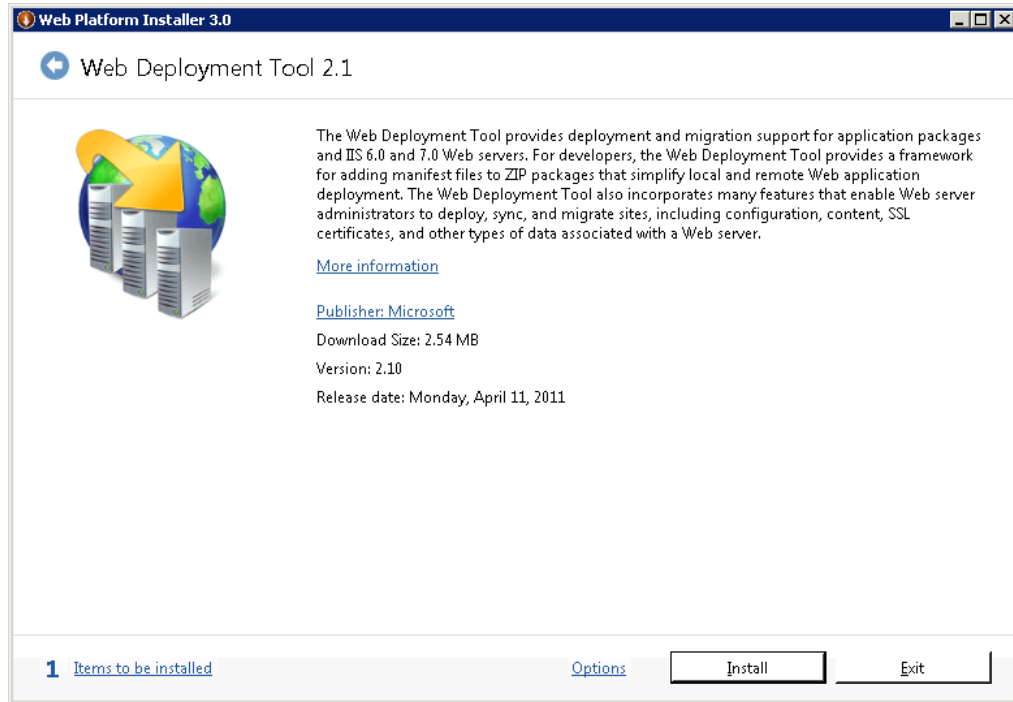
5. Click **license terms** next to the green **Install Now** button to read the license terms.



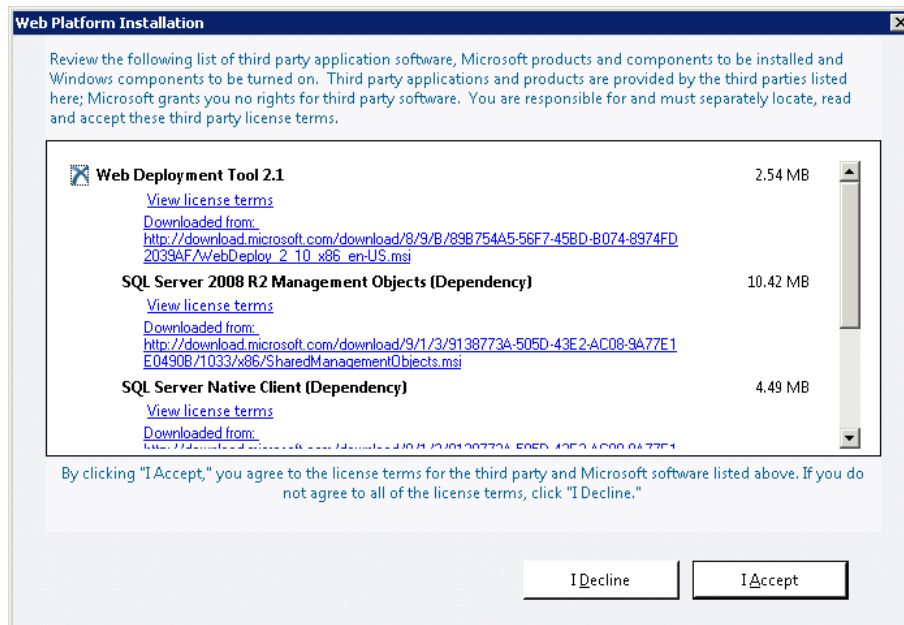
6. After you read and decide to agree to the license terms, click the green **Install Now** button.
7. When prompted, click **Run** to install the Microsoft Web Deployment Tool. The **Web Platform Installer 3.0** wizard appears.

Amazon Elastic Compute Cloud Microsoft Windows Guide

Task 1: Installing the Microsoft Web Deployment Tool

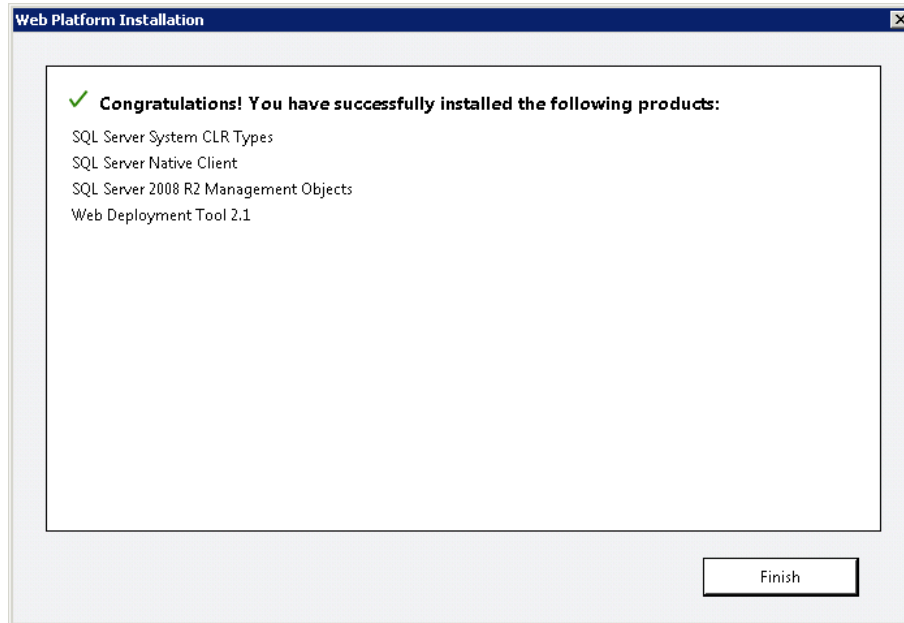


8. Click **Install**. A list of third-party application software, Microsoft products, and components appears.



9. Click **I Accept**. The Web Platform Installer begins to install the software. The wizard indicates when it's finished installing the software.

Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress



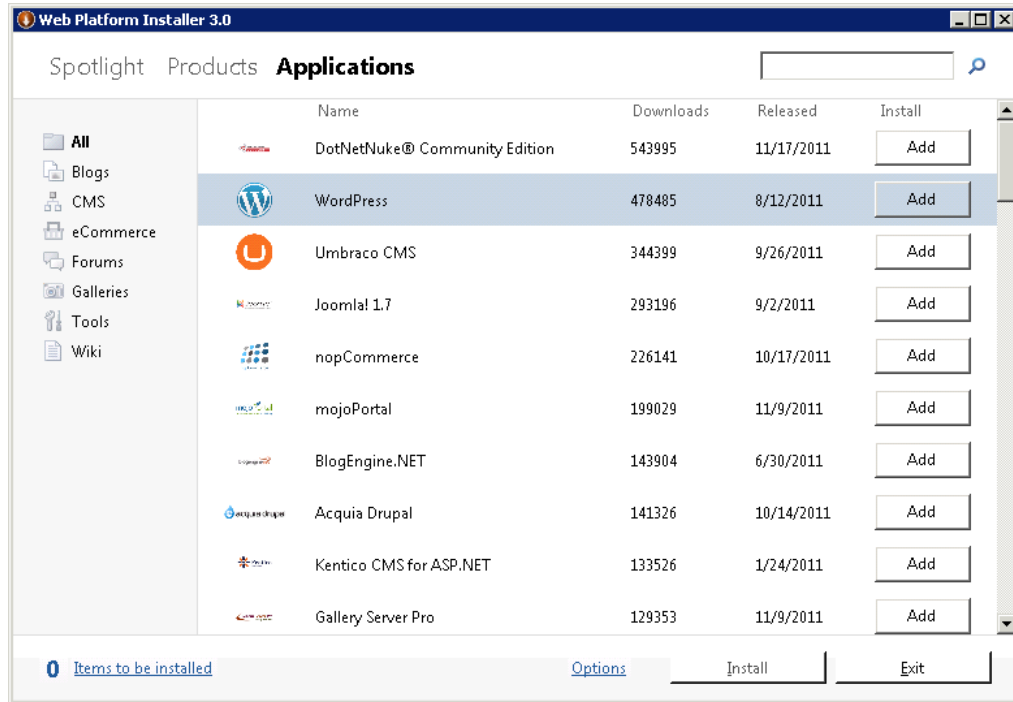
10. Click **Finish**. The **Web Platform Installer** launches so you can install WordPress.

Task 2: Installing WordPress

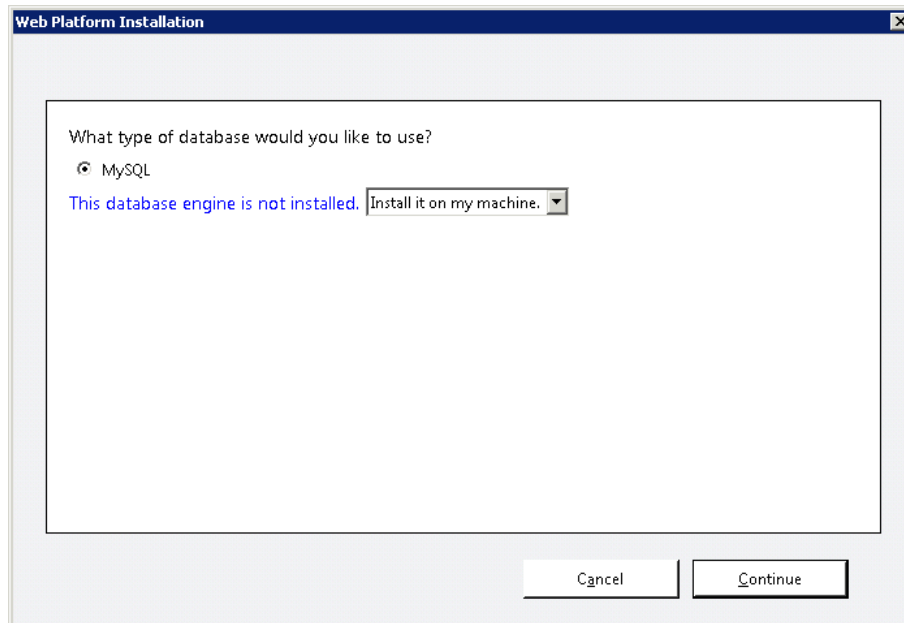
To Install WordPress

1. At the top of the **Web Platform Installer 3.0** window, click **Applications**.
2. Navigate to the **WordPress** row and click **Add**.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**

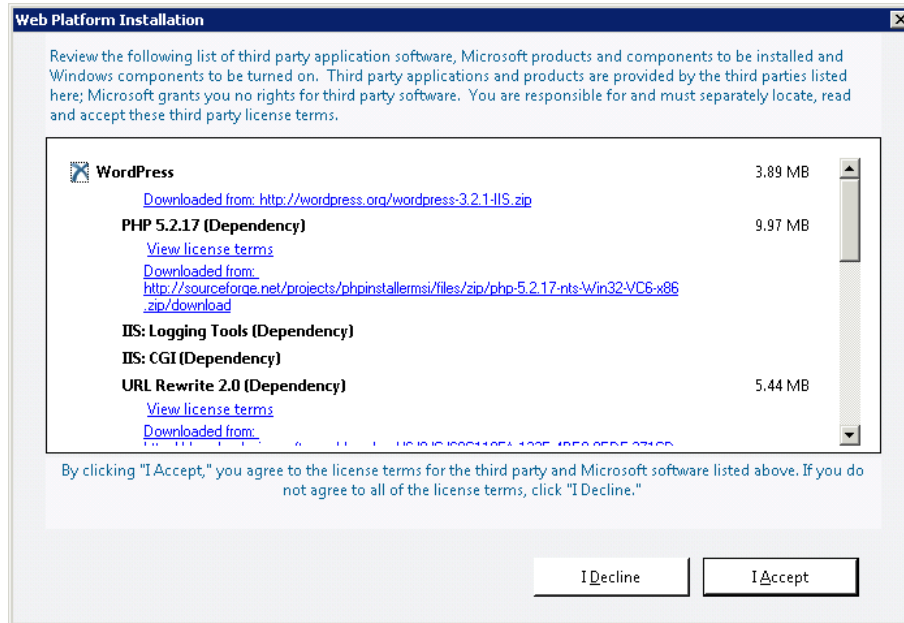


3. Click **Install**.
4. Select **MySQL** for the database you want to install. In the **This database engine is not installed** box, select **Install it on my machine**.

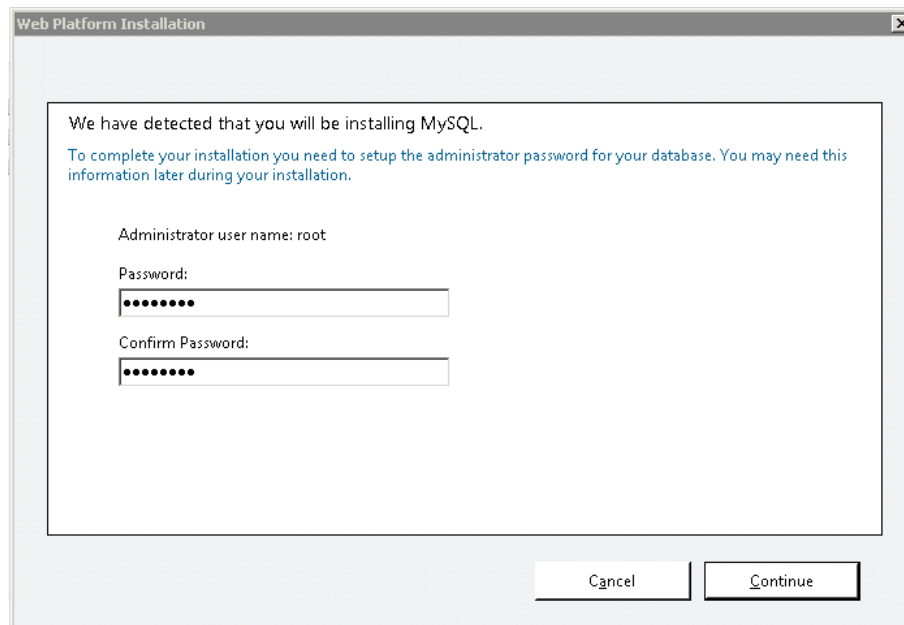


5. Click **Continue**. A list of third-party application software, Microsoft products, and components appears.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**

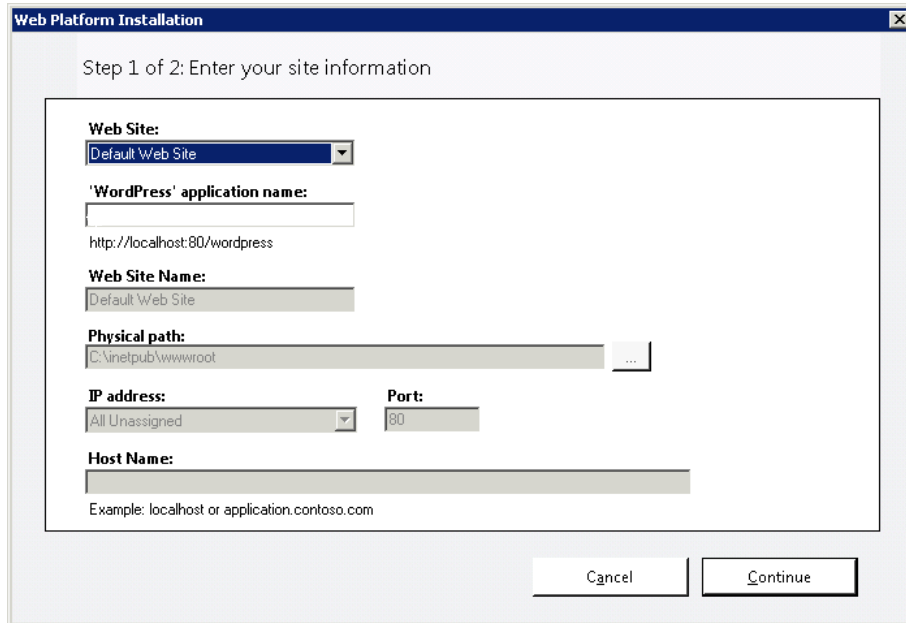


6. Click **I Accept**. The page where you enter your MySQL passwords appears.

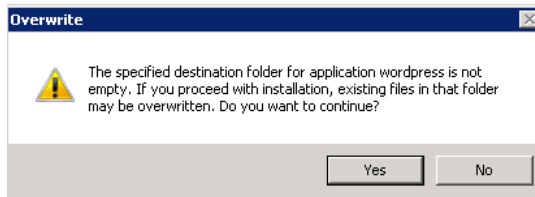


7. Type an administrator password for your MySQL database in the **Password** box and the **Confirm Password** box.
After the Web Platform Installer finishes installing the software, you are prompted to configure your new site.
8. For **Step 1 of Enter your site information:**
 - a. Clear the default application name in the '**WordPress**' **application name:** field and leave it blank.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**

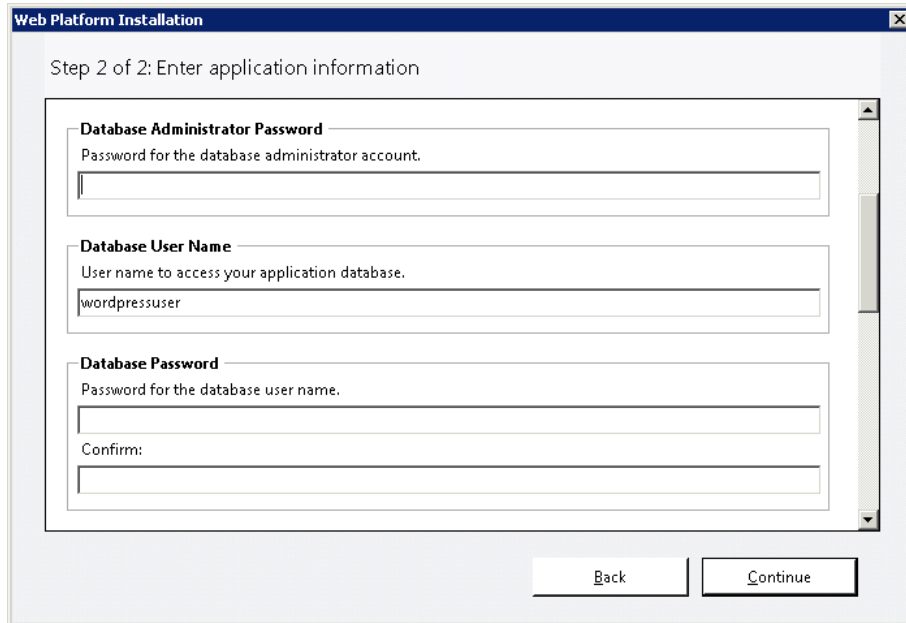


- b. Leave the default information in the other fields and click **Continue**.
- c. In the **Overwrite** box, click **Yes**.

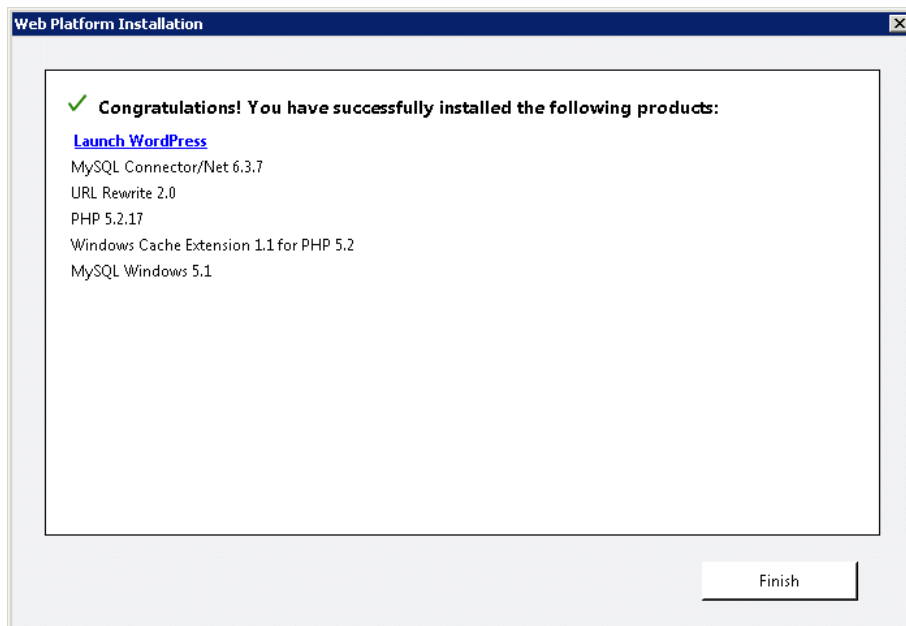


9. For **Step 2 of Enter application information**, enter the following information.
 - a. In the **Database Administrator Password** field, type the administrator password that you specified in step 7.
 - b. Leave the **Database User Name** field set to the default value.
 - c. Under **Database Password**, type a user password in the **Password for the database user name** and **Confirm** fields.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**



- d. Click **Continue**. The Web Platform Installation wizard installs the software and displays the completion page.



10. Click **Launch WordPress**. The WordPress **Welcome** page appears.
11. On the **Welcome** page, enter the following information.
- In the **Site Title** field, type your site title.
 - In the **Username** field, leave the default set to admin.
 - In the **Password, twice** fields, type your password twice.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**

- d. In the **Your E-mail** field, type your email address.

Welcome

Welcome to the famous five minute WordPress installation process! You may want to browse the [ReadMe documentation](#) at your leisure. Otherwise, just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title	<input type="text" value="MySite"/>
Username	<input type="text" value="admin"/> <small>Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods and the @ symbol.</small>
Password, twice	<input type="password" value="....."/> <input type="password" value="....."/> Strong <small>A password will be automatically generated for you if you leave this blank. Hint: The password should be at least seven characters long. To make it stronger, use upper and lower case letters, numbers and symbols like ! " ? \$ % ^ &).</small>
Your E-mail	<input type="text" value="janedoe@example.com"/> <small>Double-check your email address before continuing.</small>

Allow my site to appear in search engines like Google and Technorati.

- e. Click **Install WordPress** to install WordPress. The WordPress **Success** page appears.



Success!

WordPress has been installed. Were you expecting more steps? Sorry to disappoint.

Username	admin
Password	Your chosen password.

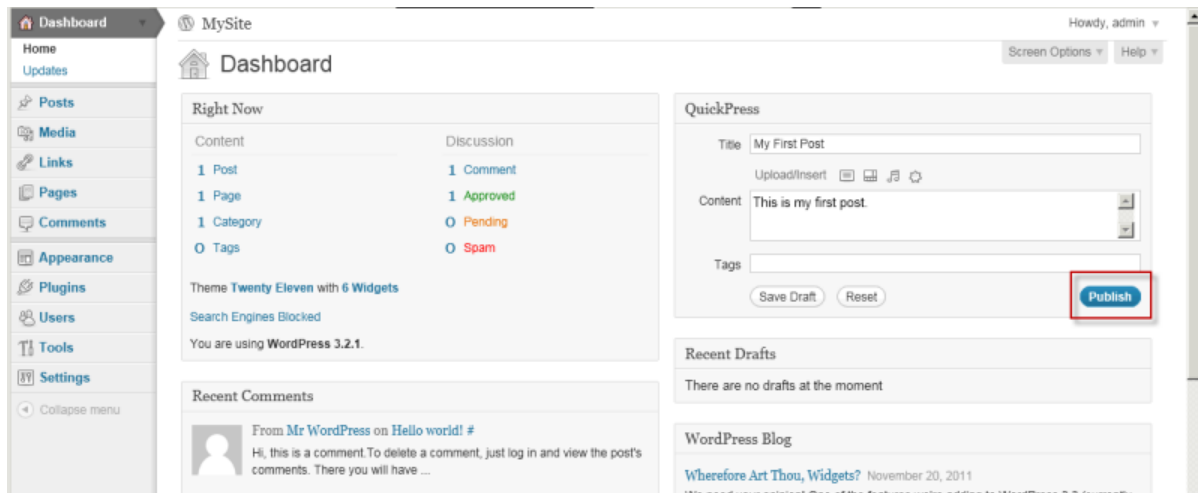
12. Click **Log In**. The **Log In** page appears.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Task 2: Installing WordPress**

13. On the **Log In** page, enter the following information.
 - a. In the **Username** field, type admin.
 - b. In the **Password** field, type the admin password you specified in step 11c.



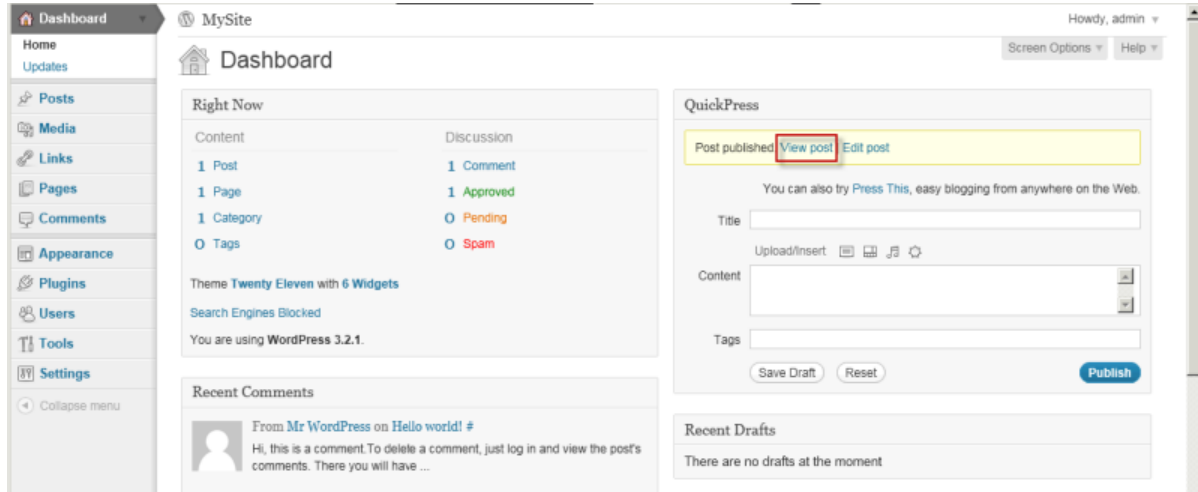
- c. Click **Log In**. The WordPress **Dashboard** appears.
14. In the **QuickPress** box, enter the following information.
 - a. In the **Title** field, enter a title for you blog.
 - b. In the **Content** field, enter content for your first blog post.



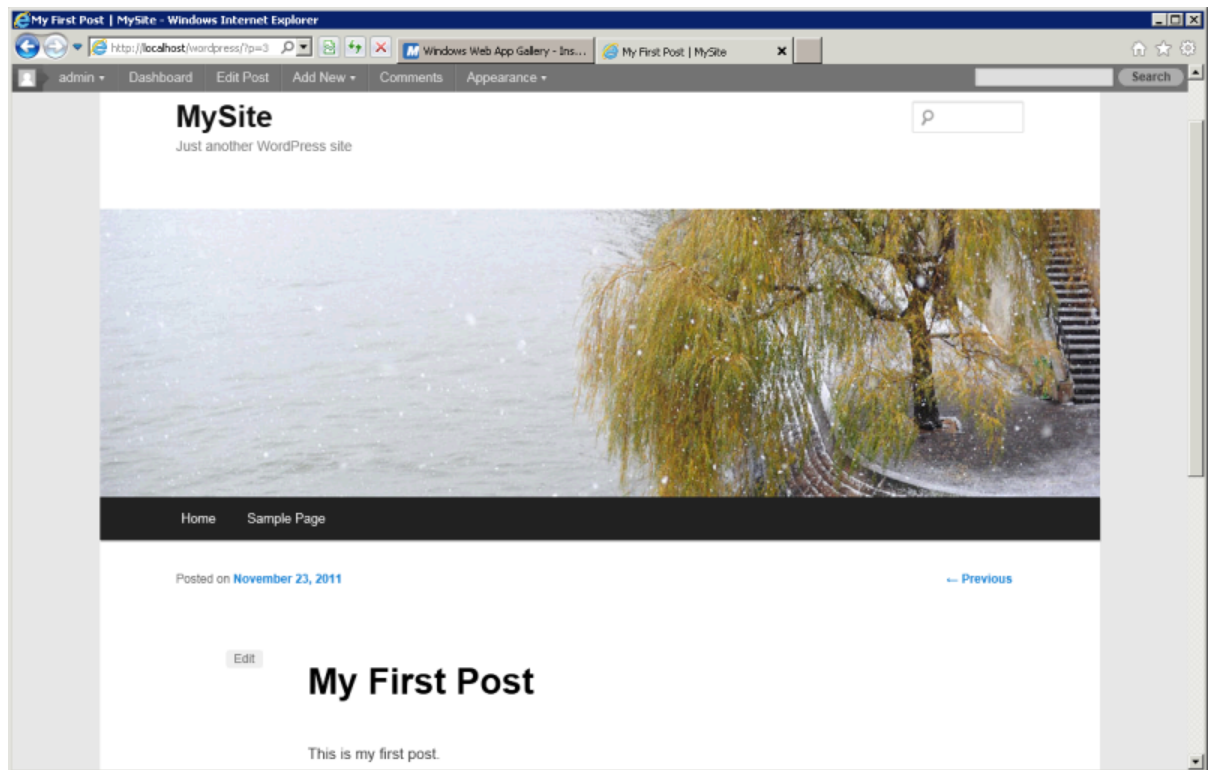
- c. Click **Publish** to publish your blog to your localhost. A notification appears in which you can choose to view or edit the post.

Amazon Elastic Compute Cloud Microsoft Windows Guide

Task 2: Installing WordPress



15. Click **View post**. Your post appears.

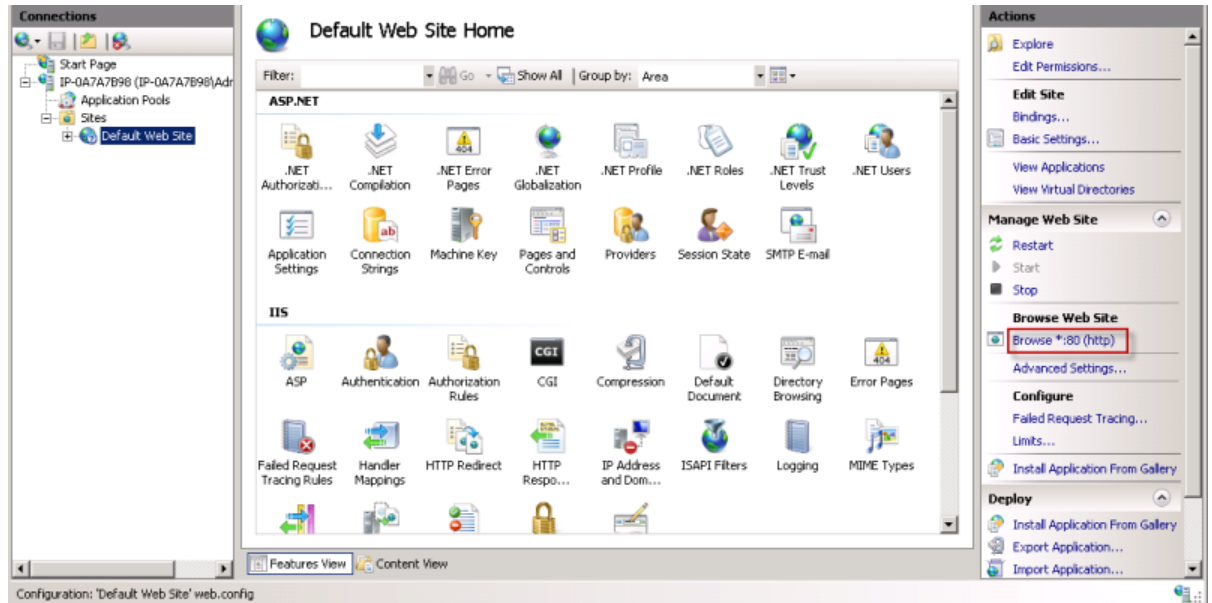


Now that you can see your WordPress blog on your localhost, you might want to publish this website as the default site on your Amazon EC2 instance so that other people can see it. The next task walks you through the process of modifying your WordPress settings to point to your Amazon EC2 instance instead of your localhost.

Task 3: Configuring Your WordPress Site

To configure the default settings for your WordPress site

1. Go back to Server Manager. Expand the **Sites** node of **Internet Information Services (IIS) Manager** and select **Default Web Site**. In the **Actions** pane, under **Browse Web Site**, click **Browse *:80 (http)**.



2. The website appears at <http://localhost/>.

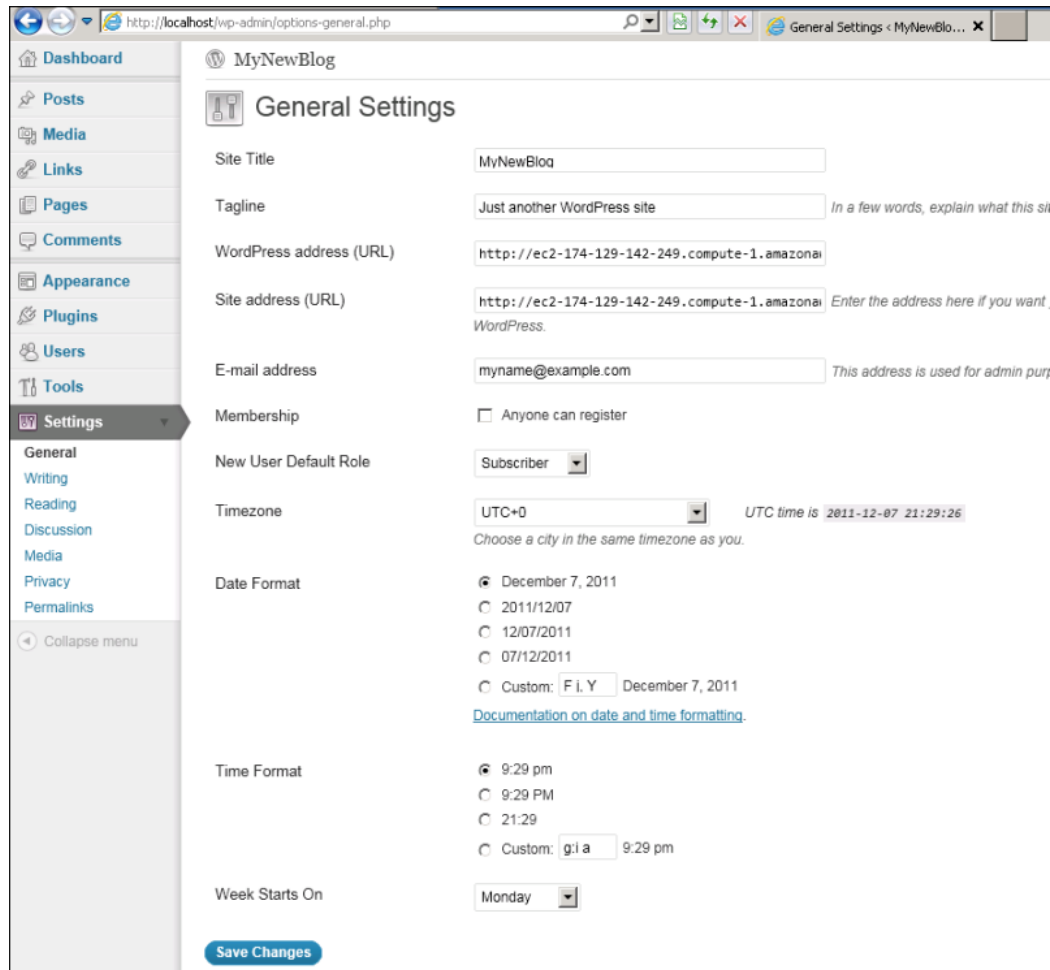
Note

If you do not see your WordPress site, click the **Refresh** button.

3. Go back to the WordPress **Dashboard**.
4. In the **Dashboard** pane, click **Settings**. The **General Settings** page appears.
5. On the **General Settings** page, enter the following information.
 - a. In the **WordPress address (URL)** box, type the public DNS address of your Amazon EC2 instance. For example, your URL may look something like this: <http://ec2-174-129-142-249.compute-1.amazonaws.com>. For information about getting your public DNS address of your Amazon EC2 instance, see the “Review your settings and launch the instance” step in the [Amazon Elastic Compute Cloud Getting Started Guide](#).
 - b. In the **Site address (URL)** box, type the same DNS address of your Amazon EC2 instance that you typed in the previous step.

Amazon Elastic Compute Cloud Microsoft Windows Guide

Task 3: Configuring Your WordPress Site



c. Click **Save Changes** at the bottom of the page.

6. Open a browser on a computer other than the EC2 instance hosting WordPress, and, in the web address field, type the public DNS address of your Amazon EC2 instance. Your WordPress site appears.

Congratulations! You have just deployed a sample WordPress site on an Amazon EC2 instance.

Amazon EC2 Infrastructure

As you get started with Amazon EC2, you should understand the Amazon EC2 infrastructure components and how they are similar to or different from your own data centers. This section provides a brief description of the main components of Amazon EC2.

Topics

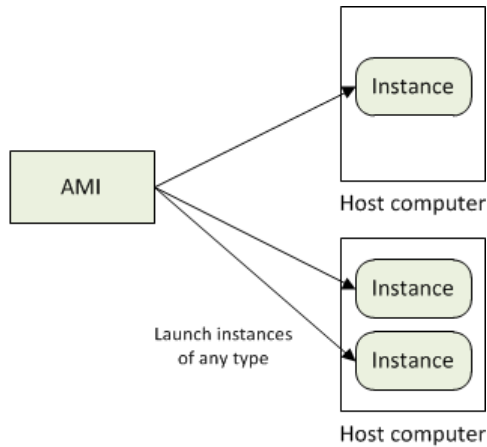
- [Amazon Machine Images and Instances \(p. 34\)](#)
- [Regions and Availability Zones \(p. 35\)](#)
- [Storage \(p. 36\)](#)
- [Networking and Security \(p. 38\)](#)
- [Monitoring, Auto Scaling, and Load Balancing \(p. 38\)](#)
- [AWS Identity and Access Management \(p. 38\)](#)
- [Available EC2 Interfaces \(p. 39\)](#)

Amazon Machine Images and Instances

An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications). From an AMI, you launch *instances*, which are copies of the AMI running as virtual servers in the cloud.

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. You can also create your own custom AMI or AMIs; doing so enables you to quickly and easily start new instances that have everything you need. For example, if your application is a web site or web service, your AMI could include a web server, the associated static content, and the code for the dynamic pages. As a result, after you launch an instance from this AMI, your web server starts, and your application is ready to accept requests.

You can launch different types of instances from a single AMI. An *instance type* essentially determines the hardware of the host computer used for your instance. Each instance type offers different compute and memory facilities. Select an instance type based on the amount of memory and computing power that you need for the applications or software that you plan to run on the instance. For more information, see [Available Instance Types](#). You can launch multiple instances from an AMI, as shown in the following figure.



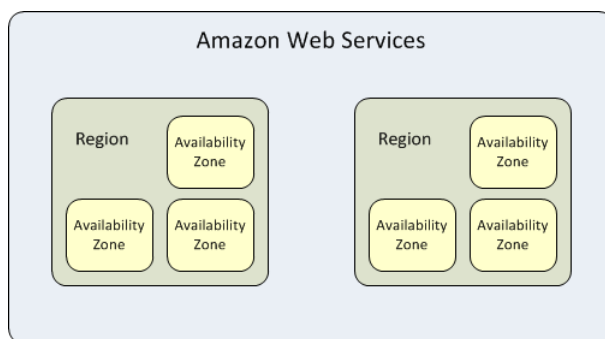
Your Windows instances keep running until you stop or terminate them, or until they fail. If an instance fails, you can launch a new one from the AMI.

For more information about Windows AMIs and instances, see [Windows Amazon Machine Images \(AMI\)](#) (p. 43) and [Windows Instance Types](#).

Regions and Availability Zones

Amazon has data centers in different areas of the world (for example, North America, Europe, and Asia). Correspondingly, Amazon 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](#)).

Each region contains multiple distinct locations called *Availability Zones*. 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.



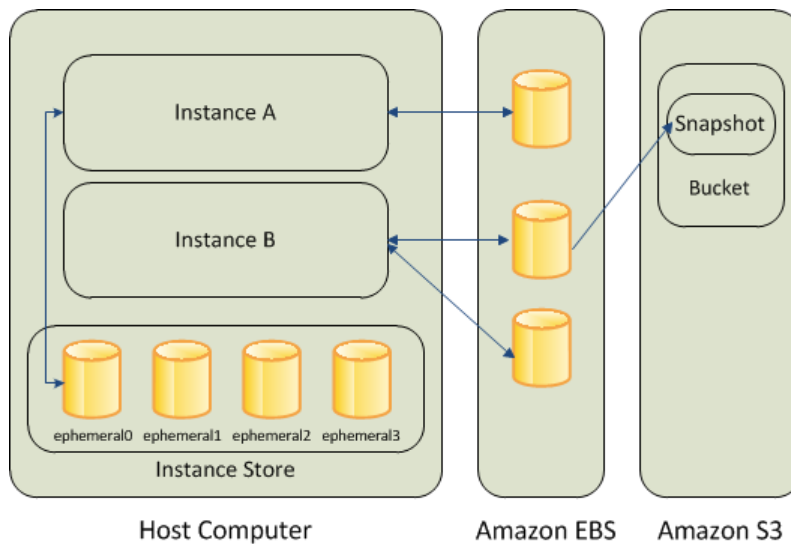
For more information about the available regions and Availability Zones, see [Using Regions and Availability Zones](#).

Storage

When using Amazon EC2, you may have data that you need to store. Amazon EC2 offers the following storage options:

- [Amazon Elastic Block Store \(Amazon EBS\)](#)
- [Amazon EC2 Instance Store](#)
- [Amazon Simple Storage Service \(Amazon S3\)](#)

The following figure shows the relationship between these types of storage.

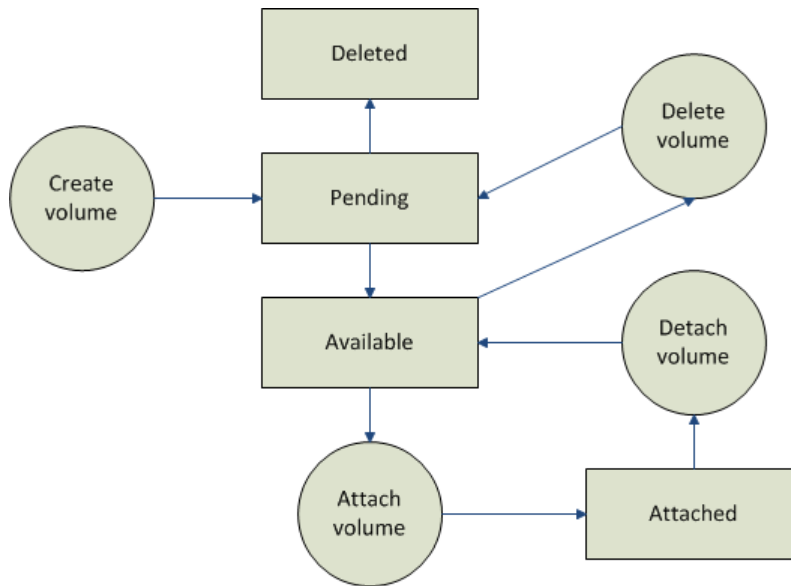


Amazon EBS Volumes

Amazon EBS volumes are the recommended storage option for the majority of use cases. 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.

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

As illustrated in the previous figure, you can attach multiple volumes to an instance. Also, to keep a back-up copy of your data, you can create a *snapshot* of an EBS volume, which is stored in Amazon S3. You can create a new Amazon EBS volume from a snapshot, and attach it to another instance. You can also detach a volume from an instance and attach it to a different instance. The following figure illustrates the life cycle of an EBS volume.



For more information about Amazon EBS volumes, see [Amazon Elastic Block Store](#).

Instance Store

All instance types, with the exception of Micro instances, offer *instance store*, which provides your instances with temporary, block-level storage. This is storage that is physically attached to the host computer. The data on an instance store volume doesn't persist when the associated instance is stopped or terminated. For more information about instance store volumes, see [Amazon EC2 Instance Store](#).

Instance store is an option for inexpensive temporary storage. You can use instance store volumes if you don't require data persistence.

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, see the [Amazon S3 product page](#).

Root Device Storage

When you launch an Amazon EC2 instance, the root device contains the image used to boot the instance.

All AMIs are categorized as either *backed by Amazon EBS*, which means that the root device for an instance launched from the AMI is an Amazon EBS volume, or *backed by instance store*, which means that the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3.

The description of an AMI indicates the type of root device (either `ebs` or `instance store`). This is important because there are significant differences in what you can do with each type of AMI. For more information about these differences, see [Root Device Storage on Windows AMIs \(p. 45\)](#).

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 has a different public IP address than the original. However, if your application needs a static IP address, Amazon EC2 offers *elastic IP addresses*. For more information, see [Using Instance IP Addresses](#).

You can use *security groups* to control who can access your instances. These are analogous to an inbound network firewall that enables 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 to reach the instance. You can configure a security group so that only specific IP addresses or specific security groups have access to the instance. For more information about security groups, see [How Security Groups Work](#) (p. 40).

Monitoring, Auto Scaling, and Load Balancing

AWS provides features that enable you to do the tasks described in the following table.

Task	Relevant Guide
Monitor basic statistics for your instances and Amazon EBS volumes.	CloudWatch Developer Guide
Automatically scale your Amazon EC2 capacity up or down according to the conditions that you define.	Auto Scaling Developer Guide
Automatically distribute incoming application traffic across multiple Amazon EC2 instances.	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 EC2 Interfaces

AWS provides different interfaces to access EC2.

AWS Management Console

The AWS Management Console is a simple web-based GUI. To get started using the console, see [Getting Started with Amazon EC2 Windows Instances](#) (p. 8).

Command Line Tools (API Tools)

EC2 provides a Java-based command-line client that wraps the EC2 API. For more information, see [Installing the Amazon EC2 Command Line Tools on Windows](#) (p. 81) and [Amazon Elastic Compute Cloud Command Line Reference](#).

Programmatic Interface

The following table lists the resources that you can use to access EC2 programmatically.

Resource	Description
AWS SDKs	AWS SDKs include sample code, libraries, tools, documentation, and templates. To download the AWS SDKs, go to AWS Software Development Kits (SDKs) .
Libraries	Developers can provide their own libraries, which you can find at the following AWS developer centers: <ul style="list-style-type: none">• Java Developer Center• Mobile Developer Center• PHP Developer Center• Python Developer Center• Ruby Developer Center• Windows and .NET Developer Center
EC2 API	If you prefer, you can code directly to the EC2 API. For more information, see Making API Requests , and go to Amazon Elastic Compute Cloud API Reference .

Controlling Access: Security Groups and Credentials

This section describes some key concepts you need to understand to successfully use Amazon EC2.

- [How Security Groups Work](#) (p. 40)
- [Using Credentials with Amazon EC2](#) (p. 42)

How Security Groups Work

A *security group* acts as a virtual firewall to control traffic allowed into a group of instances. All outbound traffic from your instances is allowed automatically. You can't change outbound behavior.

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.

For each rule, you define two settings:

- A port or a port range with the protocol allowed.
- The source that defines the access to the port. The source can be one of the following:
 - An IP address or IP address range.
 - An EC2 security group. This security group can be:
 - The current security group. If you specify the security group as the source, and you add other instances to the group, each instance within the group allows inbound traffic from other instances in the group using the ports and protocols specified.
 - Another security group in your AWS account (e.g., sg-edcd9784).
 - A security group in another AWS account. 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/ sg-edcd9784).

Restricting Access to an IP Address or IP Address Range

When you create a security group rule, the source defaults to `0.0.0.0/0`. The syntax is used is Classless InterDomain Routing (CIDR) notation. The default value allows any IP address to connect to your instance. You might want to use this setting for a web server so that anyone can see your web pages. However, for RDP access, you need to control who can access your instance, so you should use that security group rule to restrict access to a specific IP address or range of IP addresses.

If you type a specific IP address as the source, the AWS Management Console rewrites the IP address in CIDR notation. It does not change the IP address. If you want to specify only certain IP addresses, that's all you need to know about CIDR notation. If you want to specify a range of adjacent IP addresses, using CIDR blocks is useful. For more information, see [Using Security Groups](#) in the *Amazon Elastic Compute Cloud User's Guide*.

Restricting Access to a Specific Security Group

You can also define a security group as the source for a security group rule. For example, let's assume your application will use two Amazon EC2 instances:

- A web server running IIS
- A database server running SQL Server

The only source you want to be able to connect to your database server is the web server, which was launched in security group **sg-edcd9784**.

When you create the security group for your database server instance, add a rule opening port 1433 (MS SQL) and specify the source as **sg-edcd9784**. The database server will only accept MS SQL traffic from members of the sg-edcd9784 security group. In this example, only the Amazon EC2 instance running your web server can connect to your database instance on this port.

For our database server, assume that 203.0.113.19 is the static IP address of the only client computer that you want to allow to connect to the database server using RDP. You can specify the IP address as 203.0.113.19. When you enter this IP address in the AWS Management Console, the console automatically creates the CIDR notation for this address. Because the suffix 32 uses the entire IPv4 address, it allows in only a single host.

The screenshot shows the 'Security Group: MyDatabaseServerGroup' configuration page. The 'Inbound' tab is active. On the left, there is a 'Create a new rule' section with a dropdown set to 'Custom TCP rule'. Below it, the 'Port range' is set to '1433 (MS SQL)' and '3389 (RDP)'. The 'Source' is set to '0.0.0.0'. An 'Add Rule' button is visible. On the right, a table lists the rules:

Port (Service)	Source	Action
1433 (MS SQL)	sg-edcd9784	Delete
3389 (RDP)	203.0.113.19/32	Delete

Changing Security Groups Used by an Instance

When you add or modify rules in a security group, the updated rules are automatically applied to all running instances. You can also add an additional security group to an instance. If an operation is in progress when you change a security group, the operation uses the security group settings in place when the operation started. Any subsequent operations use the new security group settings applied to the instance.

After you launch an instance with a security group, you cannot remove a security group associated with the instance. If you launch an instance with the wrong security group, you can do one of the following:

- Modify the security group associated with the instance.
- Add another security group to the instance.
- Terminate the instance and launch a new one with the correct security group.

Using Credentials with Amazon EC2

Like other AWS products, Amazon EC2 requires you to use access keys and X.509 certificates to connect to your instances and interact with other services. The following table lists which credential to use in different scenarios.

Note

It is important not to confuse the secret access key (the private key portion of your SSH key pair used to retrieve an initial administrator password) with the EC2 key in the X.509 certificate, which is used to execute command line operations. Remember this distinction between secret access keys and X.509 certificates when you use the command line tools.

For more information about using your credentials with Amazon EC2, see [Using Security Groups](#) in the *Amazon Elastic Compute Cloud User's Guide*.

If you want to..	Use this credential..
Launch and connect to an instance	Amazon EC2 Key Pairs (commonly called <i>SSH key pairs</i>) Administrator password
Use the command line tools	X.509 Certificates
Have your instance talk to other AWS products (e.g., Amazon S3, etc.)	Access Keys or X.509 Certificates Put the credentials on the instance itself; the set of credentials that you use depends on the requirements of the service you are connecting to.
Bundle a Windows AMI	Access Keys Used for both bundling and uploading the AMI to Amazon S3
Use the REST API	Access Keys
Use the SOAP API	X.509 Certificates
Share an AMI or EBS snapshot	AWS Account ID of the account to share with (without the hyphens)

Windows Amazon Machine Images (AMI)

A Windows Amazon Machine Image (AMI) is a template with all the information necessary to boot an Amazon EC2 Windows instance. It is similar to a snapshot of the boot partition that contains Windows Server and other required software to run on your server. You specify an AMI when you launch your Windows instances, which are virtual servers running in the cloud.

For more information on Amazon Windows AMIs, see [Amazon Windows AMI Basics \(p. 43\)](#).

You can use the [AWS Management Console](#) to search for Windows AMIs that meet your specific criteria. For example, you can view the Windows AMIs provided by Amazon, or the Windows AMIs provided by the EC2 community. For more information on choosing a Windows AMI, see [Choosing a Windows AMI \(p. 46\)](#).

You might find public AMIs that suit your needs. You can customize a public AMI and then save that customized AMI for your own use and create a new AMI. For more information see [Creating Your Own Windows AMI \(p. 57\)](#).

After you create a new AMI, you can keep it private so that only you can use it, or you can share it with other AWS accounts that you specify. You can also 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 create and use shared AMIs, see [Shared Windows AMIs \(p. 62\)](#).

Paid AMIs are AMIs that you purchase from third parties or AMIs that come with service contracts from organizations such as Red Hat. If you're interested in selling an AMI to other developers, see [Amazon DevPay](#). You can also create your AMIs and sell it to other EC2 users. For more information on selling or using paid AMIs, see [Paid Windows AMIs \(p. 67\)](#).

To help categorize and manage your AMIs, you can assign custom *tags* to them. For more information, see [Using Tags](#).

Amazon Windows AMI Basics

Amazon Web Services (AWS) provides a set of publicly available AMIs that contain software configurations specific to the Windows platform, so that you can quickly start building and deploying your applications using Amazon EC2. First choose the AMI that meets your specific requirements, then launch an EC2

Amazon Elastic Compute Cloud Microsoft Windows Guide

Amazon Windows AMI Basics

instance (virtual server) using that AMI. You connect to the instance using Remote Desktop Connection, just as you would with any other virtual server, and then use the instance just as you would use any other Windows server.

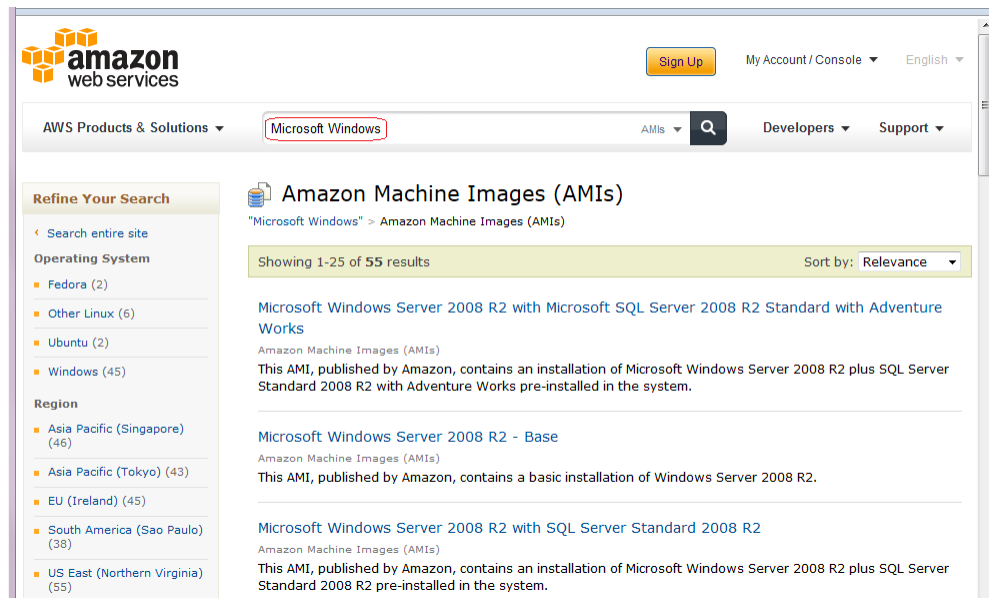
Amazon AWS currently provides AMIs based on the following versions of Windows:

- 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)
- Microsoft Windows Server 2012 (64-bit)

AWS also provides a set of publicly available AMIs that include SQL Server, SQL Server Express, Internet Information Services (IIS), and ASP.NET to help you get started quickly. You can use one or more of these AMIs to deploy your applications. For example, you can use an Amazon Windows AMI with SQL Server Express, IIS, and ASP.NET to launch an instance that runs web and ASP.NET applications. Launching an instance from an Amazon Windows AMI with SQL Server offers you the flexibility to run the instance as a database server. Or, you can launch an instance from one of the basic Windows AMIs, customize the instance by installing the software and applications of your choice, and then register the customized instance as an AMI. You can then use this customized AMI to launch additional instances that include your chosen software and applications.

In addition to the public AMIs provided by AWS, there are AMIs published by the AWS developer community available for your use. We highly recommend that you use only those Windows AMIs that Amazon or other reputable sources provide.

For a list of Amazon-approved Microsoft Windows AMIs, go to [Amazon Machine Images \(AMIs\)](#) on the AWS website. You can refine your search by selecting the platform name *Microsoft Windows* in the search box as shown in the following screen shot.



Click any AMI in the list to see all the relevant information about it.

Root Device Storage on Windows AMIs

An Amazon EC2 Windows instance can be launched from an AMI backed either by instance store or by Amazon Elastic Block Store (Amazon EBS). This section describes the differences between these two types of AMIs. It is important to consider these differences before you choose an AMI.

Instances launched from an AMI backed by instance store use an instance store volume as the root device. The image of the root device volume of an instance store-backed AMI is initially stored in Amazon S3. When an instance is launched using an instance store-backed AMI, the image of its root device is copied from Amazon S3 to the root partition of the instance. The root device volume is then used to boot the instance.

Instances launched from an AMI backed by Amazon EBS use an Amazon EBS volume as the root device. The root device volume of an Amazon EBS-backed AMI is an Amazon EBS snapshot. When an instance is launched using an Amazon EBS-backed AMI, a root EBS volume is created from the EBS snapshot and attached to the instance. The root device volume is then used to boot the instance.

When you select **AMIs** in the **Navigation** pane of the EC2 console, the **Root Device** column indicates whether the AMI is backed by instance store (*instance-store*) or Amazon EBS (*ebs*).

The following table provides a summary of the differences between instance store-backed AMIs and Amazon EBS-backed AMIs.

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	Amazon EBS volume	Instance store volume
Data Persistence	Data on EBS volumes persists on instance failure and can persist on instance termination	Data on instance store volumes persists for the life of the instance
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 (AMI storage)	Instance usage and Amazon S3 (AMI storage)
Stopped State	Can be placed in the stopped state (the instance is not running, but is persisted in Amazon EBS)	Cannot be placed in the stopped state

Configuration of an Amazon Windows AMI

The Amazon-provided Windows AMIs are, as much as possible, configured the same way as the Windows Server you install from Microsoft-issued media. There are however, a few differences in the installation defaults. An Amazon EC2 Windows AMI comes with an additional service installed, the EC2Config Service.

The EC2Config Service runs in the local system account and is primarily used during the initial setup. EC2Config performs the following tasks when launching your instance:

- Sets the hostname to the private DNS name
- Generates and sets a random initial password on the administrator's account
- Initializes and formats all the drives attached to the instance
- Generates and installs the host certificate for Remote Desktop
- Syncs the instance clock with a time server

After you launch your Windows instance with its initial configuration, you can use the EC2Config Service to change the configuration settings as part of the process of customizing and creating your own AMIs. Instances launched from your customized AMI are launched with the new configuration. The binaries for the EC2Config Service, as well as additional tools needed to configure the new Windows AMI, are contained in the `%ProgramFiles%\Amazon` (32-bit instances) or `%ProgramFiles(x86)\Amazon` (64-bit instances) directory. For more information, see [Creating Your Own Windows AMI](#) (p. 57).

Xen Drivers

Amazon Windows AMIs contain a set of drivers to permit access to Xen virtualized hardware. These drivers are used by Amazon EC2 to map the instance store and Amazon Elastic Block Store (Amazon EBS) volumes to the devices. The source files for the drivers are in the `%ProgramFiles%\RedHat` (32-bit instances) or `%ProgramFiles(x86)\RedHat` (64-bit instances) directory. The two drivers are `rhelnet`, the RedHat Paravirtualized network driver, and `rhelscsi`, the RedHat SCSI miniport driver.

Keeping Your Instances Updated

At their initial launch, your Windows instances contain all the latest security updates. However, after you launch an instance, you are responsible for managing future updates, including the updates issued after you built the AMI. You can use the Windows Update service, or the Automatic Updates tool available on your instance to deploy the Microsoft updates. Any third-party software you deploy must also be kept up-to-date using whatever mechanisms are appropriate for that software. We recommend that you run the Windows Update service first thing with every Windows instance that you launch.

Note

You can reboot an Amazon EC2 Windows instance after the updates take place. Rebooting works the same way for both instance store-backed instances and Amazon EBS-backed instances. For more information, see [Root Device Storage on Windows AMIs](#) (p. 45).

Support

Support for installation and use of the base Amazon Windows 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 about using Amazon Windows AMIs to the [Amazon EC2 forums](#).

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

Choosing a Windows AMI

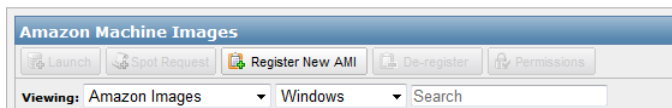
Amazon Machine Images (AMIs) are the basic building block of Amazon EC2. Before you accomplish anything with Amazon EC2, you must first choose an AMI. The AMI can be provided by Amazon or the Amazon EC2 community, or you can create your own AMIs. However, to create your own AMI, you must start by using one of the base AMIs provided.

After finding and selecting an AMI, record its AMI ID. You'll use the AMI ID when you launch your instance and then connect to it. For information about launching your instance, see [Launch a Windows Instance](#) (p. 9). For information about connecting to your Windows instance, see [Connect to Your Windows Instance](#) (p. 12).

Using the AWS Management Console

To view a list of available AMIs

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Click **AMIs** in the **Navigation** pane.
The console displays a list of all available AMIs.
3. [Optional] Use the **Viewing** options to narrow the list of displayed AMIs. For example, to see a list of all Windows AMIs provided by Amazon, select **Amazon Images** from the first drop-down list and **Windows** from the second list, as shown here.



4. Click an AMI to view its properties. The AMI properties appear in the **Description** tab in the lower pane.

As you are selecting an AMI, it's important to note whether the AMI is backed by instance store or by Amazon EBS. Select the type of AMI that meets your needs. For more information, see [Root Device Storage on Windows AMIs](#) (p. 45).

Using Command Line Tools

Amazon EC2 provides a Java-based command-line client that wraps the EC2 Query API. You must install the command line tools before you can try the example commands in this section. For information about installing the command line tools, see [Installing the Amazon EC2 Command Line Tools on Windows](#) (p. 81).

To find a suitable AMI

- Use the `ec2-describe-images` command to list the AMIs that you're interested in.

The following command lists all Amazon-owned Windows AMIs. The example output shown here consists of a few entries from the list of all Amazon Windows AMIs.

```
C:\> ec2-describe-images -o amazon --filter "platform=windows"

IMAGE ami-c941efa0 amazon/Windows_Server-2008-SP2-English-64Bit-Base-
2012.07.11 amazon available public x86_64
machine windows ebs hvm xen
BLOCKDEVICEMAPPING /dev/sda1 snap-b81a74c9 30 standard
IMAGE ami-2b41ef42 amazon/Windows_Server-2008-R2_SP1-English-64Bit-Base-
2012.07.11 amazon available public x86_64
machine windows ebs hvm xen
BLOCKDEVICEMAPPING /dev/sda1 snap-f00e6081 30 standard
IMAGE ami-b340eeda amazon/Windows_Server-2008-R2_SP1-English-64Bit-
SQL_2008_R2_SP1_Express-2012.07.11 amazon available public x86_64
```

```
machine windows ebs hvm xen
BLOCKDEVICEMAPPING /dev/sda1 snap-0e2d437f 30 standard
IMAGE ami-a8e705c1 ec2-paid-ibm-images/ibm-infosphere-is-winclient.manifest.xml amazon available public [devpay: EC129708]
i386 machine windows instance-store hvm xen
IMAGE ami-df20c3b6 ec2-public-windows-images/Server2003r2-i386-Win-v1.07.manifest.xml amazon available public i386
machine windows instance-store hvm xen
IMAGE ami-dd20c3b4 ec2-public-windows-images/Server2003r2-x86_64-Win-v1.07.manifest.xml amazon available public x86_64
machine windows instance-store hvm xen
```

Tip

You can filter the 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*.

Configuring a Windows Instance Using EC2ConfigService

Amazon Windows AMIs contain an additional service installed by Amazon Web Services—the EC2Config service. Although optional, this service provides access to advanced features that aren't otherwise available. This service runs in the LocalSystem account and performs tasks on the instance. Its binaries and additional files are contained in the `%ProgramFiles%\Amazon\EC2ConfigService` directory.

The EC2Config service is started when the instance is booted. It performs tasks during initial instance startup and each time you stop and start the instance. It can also perform tasks on demand. Some of these tasks are automatically enabled, while others must be enabled. EC2ConfigService uses settings files to control its operation. You can update these settings files using either a graphical tool or by directly editing XML files.

EC2ConfigService runs Sysprep, a Microsoft tool that enables you to create a customized Windows image that can be reused. For more information about Sysprep, see [Sysprep Technical Reference](#).

When EC2ConfigService calls Sysprep, it uses the settings files in `EC2ConfigService\Settings` to determine which operations to perform. You can edit these files indirectly using the **Ec2 Service Properties** dialog box, or directly using an XML editor or a text editor. However, there are some advanced settings that aren't exposed in the **Ec2 Service Properties** dialog box, so you must edit those entries directly.

If you create an AMI from an instance after updating its settings, the new settings are applied to any instance that's launched from the new AMI. For information about creating an AMI after using EC2ConfigService, see [Creating an Amazon EBS-Backed Windows AMI \(p. 57\)](#).

Topics

- [EC2ConfigService Tasks \(p. 49\)](#)
- [Ec2 Service Properties \(p. 49\)](#)
- [EC2ConfigService Settings Files \(p. 52\)](#)
- [Installing the Latest Version of EC2ConfigService \(p. 55\)](#)
- [Stopping, Deleting, or Uninstalling EC2ConfigService \(p. 55\)](#)
- [EC2ConfigService Release Notes \(p. 56\)](#)

EC2ConfigService Tasks

Initial startup tasks are run when the instance is first started and are then disabled. To run these tasks again, you must explicitly enable them prior to shutting down the instance, or by running Sysprep manually. These tasks are as follows:

- Set the computer name (to match the private DNS name).
- Set a random, encrypted password for the Administrator account.
- Generate and install the host certificate used for Remote Desktop Connection.
- Dynamically extend the operating system partition.
- Execute the specified User Data (and CloudInit, if it's installed).
- Configure the Key Management Server (KMS) and activate Windows.

The following tasks are performed every time the instance starts:

- Check for activation status and activate Windows as necessary.
- Format and mount any EBS volumes and instance store volumes, and map volume names to drive letters.
- Synchronize the instance clock with a time server.
- Write event log entries to the console to help with troubleshooting.
- Write to the console that Windows is ready.
- Display wallpaper information to the desktop background.
- Add a custom route to the primary network adapter to enable the following IP addresses when multiple NICs are attached: 169.254.169.250, 169.254.169.251, and 169.254.169.254. These addresses are used by Windows Activation and when you access instance metadata.

While the instance is running, you can request that EC2ConfigService perform the following task on demand:

- Run Sysprep and shut down the instance so that you can create an AMI from it. (For more information, see [Creating an Amazon EBS-Backed Windows AMI \(p. 57\)](#).)

Ec2 Service Properties

The following procedure describes how to use the **Ec2 Service Properties** dialog box to enable or disable settings.

To change settings using the EC2Config Service tool

1. Launch and connect to your Windows instance.
2. From the **Start** menu, click **All Programs**, and then click **EC2ConfigService Settings**. Your Windows instance displays the **Ec2 Service Properties** dialog box.
3. On the **General** tab, you can enable or disable the following settings.

Set Computer Name

Sets the hostname of the instance to a unique name based on the IP address of the instance and reboots once after booting. To set your own hostname, or to prevent your existing hostname from being modified, don't enable this setting.

User Data

Creates and executes scripts created by the user on the first launch of an instance after Sysprep is run.

Amazon Elastic Compute Cloud Microsoft Windows Guide Ec2 Service Properties

Commands wrapped in `script` tags are saved to a batch file, and commands wrapped in `powershell` tags are saved to a .ps1 file.

Event Log

Enables the display of event log entries on the console during boot for easy monitoring and debugging.

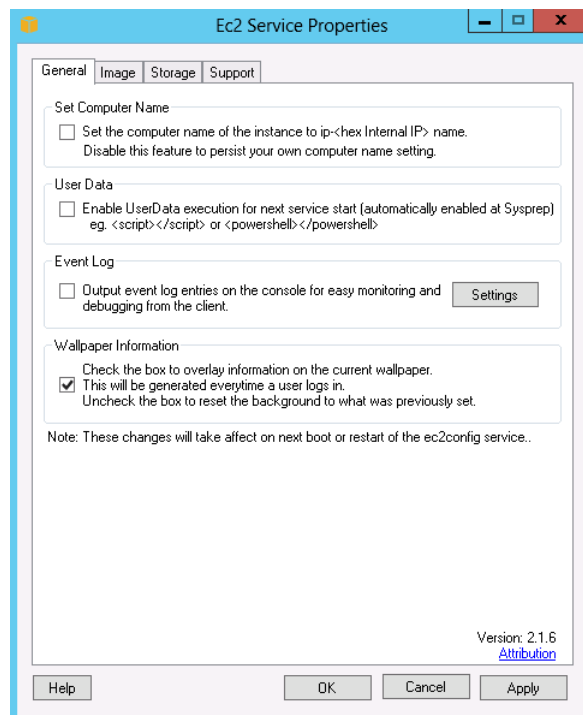
Click the **Settings** button to specify filters for the log entries sent to the console. By default, the three most recent error entries from the System event log are sent to the console.

Wallpaper Information

Enables the display of system information on the desktop background. The information displayed on the desktop background is controlled by the settings file `EC2ConfigService\Settings\WallpaperSettings.xml`.

The following is an example of the information displayed on the desktop background.

```
Instance ID : i-ad019dd6
IP Address  : 23.22.254.170
Availability Zone : us-east-1c
Instance Size : m1.small
Architecture : AMD64
Total Memory : 1.7 GB
Processing Power : 1 ECU
I/O Performance : Moderate
```



4. Click the **Storage** tab. You can enable or disable the following settings.

Root Volume

Dynamically extends Disk 0/Volume 0 to include any unpartitioned space. This can be useful when the instance is booted from a root device volume that has a custom size.

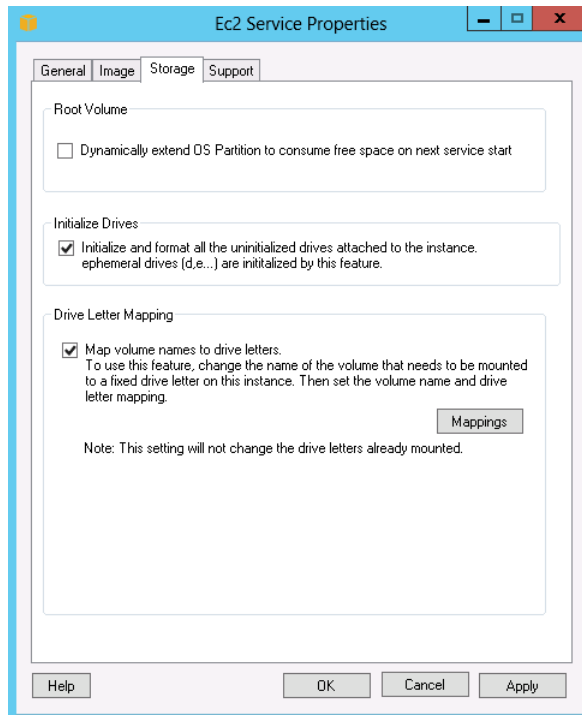
Initialize Drives

Formats and mounts all instance store volumes attached to the instance during startup.

Drive Letter Mapping

By default, drives are mapped to letters by the system. To specify your own mappings, click the **Mappings** button. In the **DriveLetterSetting** dialog box, specify the **Volume Name** and **Drive Letter** for each mapping, and then click **OK**.

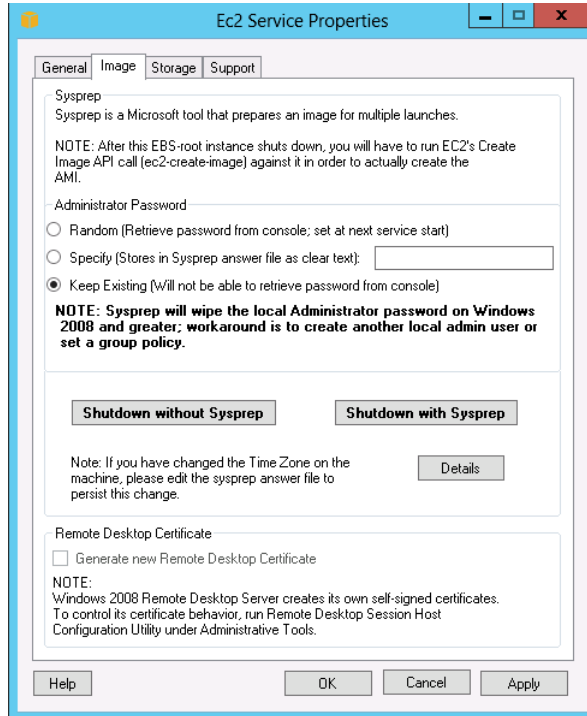
These settings take effect immediately on volumes you attach after following this procedure. These settings don't change the drive letters of volumes that are already mounted. However, the mapping fails if the drive letter is already in use. To minimize this problem, we recommend that you pick drive letters from the end of the alphabet.



5. To save your settings and continue working on them later, click **OK** to close the **Ec2 Service Properties** dialog box.

Otherwise, if you have finished customizing your instance and are ready to create your AMI from this instance, click the **Image** tab. Select an option for the Administrator password, and then click **Shutdown with Sysprep** or **Shutdown without Sysprep**. EC2ConfigService edits the settings files based on the password option that you selected.

Amazon Elastic Compute Cloud Microsoft Windows Guide EC2ConfigService Settings Files



When you are asked to confirm that you want to run Sysprep and shut down the instance, click **Yes**. You'll notice that EC2ConfigService runs Sysprep. Next, you are logged off the instance, and the instance is shut down. If you check the **Instances** page in the EC2 console, the instance state changes from *running* to *stopping*, and then finally to *stopped*. At this point, it's safe to create an AMI from this instance.

You can manually invoke the Sysprep tool from the command line using the following command:

```
%ProgramFiles%\Amazon\Ec2ConfigService\ec2config.exe -sysprep
```

However, you must be very careful that the XML file options specified in the `Ec2ConfigService\Settings` folder are correct, otherwise, you might not be able to connect to the instance. For more information about the settings files, see [EC2ConfigService Settings Files](#) (p. 52). For an example of configuring and then running Sysprep from the command line, see `Ec2ConfigService\Scripts\InstallUpdates.ps1`.

EC2ConfigService Settings Files

You can modify the following settings files located in the `Ec2ConfigService\Settings` directory:

- `ActivationSettings.xml`—Controls product activation using a Key Management Server (KMS).
- `BundleConfig.xml`—Controls how EC2ConfigService prepares an instance for AMI creation.
- `Config.xml`—Controls the primary settings.
- `DriveLetterConfig.xml`—Controls drive letter mappings.
- `EventLogConfig.xml`—Controls the event log information that's displayed on the console while the instance is booting.
- `WallpaperSettings.xml`—Controls the information that's displayed on the desktop background.

ActivationSettings.xml

- `SetAutodiscover`—Indicates whether to automatically detect a KMS.
- `TargetKMSServer`—The private IP address of a KMS. The KMS must be in the same region as your instance.
- `DiscoverFromZone`—Discovers the KMS server from the specified DNS zone.
- `ReadFromUserData`—Gets the KMS server from `UserData`.
- `LegacySearchZones`—Discovers the KMS server from the specified DNS zone.
- `DoActivate`—Attempt activation using the specified settings in the section. This value can be `true` or `false`.
- `LogResultToConsole`—Displays the result to the console.

BundleConfig.xml

- `AutoSysprep`—Indicates whether to automatically use Sysprep. Change the value to `Yes` if you want to use Sysprep.
- `SetRDPCertificate`—Sets a self-signed certificate to the Remote Desktop server running on a Windows 2003 instance. This enables you to securely RDP into the instances. Change the value to `Yes` if you want the new instances to have the certificate.
This setting is not used with Windows Server 2008 or Windows Server 2012 instances because they can generate their own certificates.
- `SetPasswordAfterSysprep`—Sets a random password on a newly launched instance, encrypts it with the user launch key, and outputs the encrypted password to the console. Change the value of this setting to `No` if you do not want the new instances to set a random encrypted password.

Config.xml

- `Ec2SetPassword`—Generates a random encrypted password each time you launch an instance. This feature is disabled by default after the first launch so that reboots of this instance don't change a password set by the user. Change this setting to `Enabled` to continue to generate passwords each time you launch an instance.

This setting is important if you are planning to create an AMI from your instance.

- `Ec2SetComputerName`—Sets the hostname of the instance to a unique name based on the IP address of the instance and reboots the instance. To set your own hostname, or prevent your existing hostname from being modified, you must disable this setting.
- `Ec2InitializeDrives`—Initializes and formats all instance store volumes during startup. This feature is enabled by default, and initializes and mounts the instance store volumes as drives D:/, E:/, and so on. For more information about instance store volumes, see [Amazon EC2 Instance Store](#) in the Amazon EC2 User Guide.
- `Ec2EventLog`—Displays event log entries in the console. By default, the three most recent error entries from the system event log are displayed. To specify the event log entries to display, edit the `EventLogConfig.xml` file located in the `EC2ConfigService\Settings` directory. For information about the settings in this file, see [Eventlog Key](#) in the MSDN Library.
- `Ec2ConfigureRDP`—Sets up a self-signed certificate on the instance, so users can securely access the instance using Remote Desktop. This feature is disabled on Windows Server 2008 and Windows Server 2012 instances because they can generate their own certificates.
- `Ec2OutputRDPcert`—Displays the Remote Desktop certificate information to the console so that the user can verify it against the thumbprint.
- `Ec2SetDriveLetter`—Sets the drive letters of the mounted volumes based on user-defined settings. By default, when an Amazon EBS volume is attached to an instance, it can be mounted using the drive letter on the instance. To specify your drive letter mappings, edit the `DriveLetterConfig.xml` file located in the `EC2ConfigService\Settings` directory.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
EC2ConfigService Settings Files**

- **Ec2WindowsActivate**—Indicates whether to search through the DNS Suffix List for appropriate KMS entries. When the appropriate KMS entries are found, the plug-in sets your activation server to the first server to respond to the request successfully. Starting with Windows Server 2008 R2, Windows Server is able to search the suffix list automatically. With Windows Server 2008 R2 and Windows Server 2012, the plug-in performs this search manually.

To modify the KMS settings, edit the `ActivationSettings.xml` file located in the `EC2ConfigService\Settings` directory.

- **Ec2DynamicBootVolumeSize**—Extends Disk 0/Volume 0 to include any unpartitioned space.
- **Ec2HandleUserData**—Creates and executes scripts created by the user on the first launch of an instance after Sysprep is run. Commands wrapped in script tags are saved to a batch file, and commands wrapped in PowerShell tags are saved to a .ps1 file.

DriveLetterConfig.xml

- **DriveLetterMapping**—Sets the drive letter mappings. Construct the following XML to create drive letter mappings.

```
<?xml version="1.0" standalone="yes"?>
<DriveLetterMapping>
  <Mapping>
    <VolumeName></VolumeName>
    <DriveLetter></DriveLetter>
  </Mapping>
  . . .
  <Mapping>
    <VolumeName></VolumeName>
    <DriveLetter></DriveLetter>
  </Mapping>
</DriveLetterMapping>
```

- **VolumeName**—The volume name exposed to EC2. For example, xvdf.
- **DriveLetter**—The drive letter. For example, X:.

EventLogConfig.xml

- **Category**—The event log key to monitor.
- **ErrorType**—The event type (for example, Error, Warning, Information.)
- **NumEntries**—The number of events stored for this category.
- **LastMessageTime**—To prevent the same message from being pushed repeatedly, the service updates this value every time it pushes a message.
- **AppName**—The event source or application that logged the event.

WallpaperSettings.xml

- **Instance ID**—Displays the ID of the instance.
- **Public IP Address**—Displays the public IP address of the instance.
- **Private IP Address**—Displays the private IP address of the instance.
- **Availability Zone**—Displays the Availability Zone in which the instance is running.
- **Instance Size**—Displays the type of instance.
- **Architecture**—Displays the setting of the `PROCESSOR_ARCHITECTURE` environment variable.
- **AddMemory**—Displays the system memory, in GB.

- `AddECU`—Displays the processing power, in ECU.
- `AddIO`—Displays the I/O performance.

Installing the Latest Version of EC2ConfigService

By default, EC2ConfigService is included in each Amazon Windows AMI. When Amazon releases an updated version of EC2ConfigService, we update all Amazon Windows AMIs with the latest version. However, you'll need to update your own Windows AMIs and instances with the latest version of EC2ConfigService.

To find notifications of updates to EC2ConfigService, go to the [Amazon Elastic Compute Cloud](#) forum.

To verify the version of EC2ConfigService that is included with your Windows AMI, launch an instance from your AMI and connect to it. From Control Panel, select **Programs and Features**. Look for EC2ConfigService in the list of installed programs. Its version number appears in the **Version** column.

To install version 2.1.6 of EC2ConfigService

1. Download [EC2Install.zip](#) and unzip the file.
2. Run `EC2Install.exe`.
The setup program stops the service, uninstalls it, and reinstalls the new version.
3. Reboot your instance.
4. Connect to your instance, run the Services administrative tool, and verify that the status of `EC2ConfigService` is `Started`.

For more information about the changes in each version of EC2ConfigService, see [EC2ConfigService Release Notes \(p. 56\)](#).

Stopping, Deleting, or Uninstalling EC2ConfigService

You can manage EC2ConfigService just as you would any other service.

If you want to apply updated settings to your instance, you can stop and restart the service. If you're manually installing EC2ConfigService, you must stop the service first.

To stop EC2ConfigService

1. Launch and connect to your Windows instance.
2. On the **Start** menu, point to **Administrative Tools**, and then click **Services**.
3. In the list of services, right-click EC2Config, and select **Stop**.

If you don't need to update the configuration settings or create your own AMI, you can delete the service. Deleting a service removes its registry subkey.

To delete EC2ConfigService

1. Start a command prompt window.
2. Run the following command:

```
sc delete ec2config
```

If you don't need to update the configuration settings or create your own AMI, you can uninstall the service. Uninstalling a service removes the files, the registry subkey, and any shortcuts to the service.

To uninstall EC2ConfigService

1. Launch and connect to your Windows instance.
2. On the **Start** menu, click **Control Panel**.
3. Double-click **Programs and Features**.
4. On the list of programs, select EC2ConfigService, and click **Uninstall**.

EC2ConfigService Release Notes

The following are the release notes for the released versions of EC2ConfigService.

Changes in 2.1.6

- Added version information to the **General** tab.
- Renamed the **Bundle** tab to **Image**.
- Simplified the process of specifying passwords and moved the password-related UI from the **General** tab to the **Image** tab.
- Renamed the **Disk Settings** tab to **Storage**.
- Set the Windows 2003 `Sysprep.ini` to extend the OS partition by default.
- Added the private IP address to the wallpaper.
- Add a **Support** tab with common tools for troubleshooting.

Changes in 2.1.2

- Changed the console time stamps to UTC.
- The `Sysprep.xml` for Windows 2003 defaults to UTC instead of Pacific.
- Added a check box to enable the execution of user scripts.
- Added a feature to dynamically expand the root volume on first boot.
- Selecting **Set a random password on next boot** automatically selects **Enable SetPassword feature after sysprep**.
- Renamed the **Drive Mapping** tab to **Disk Settings**.
- Moved **Initialize Drives** from the **General** tab to the **Disk Settings** tab.
- Fixed an issue where clicking the **Help** button didn't open the help file.
- Added `InstallUpdates.ps1` to the `Scripts` folder for automating patches and cleanup prior to running Sysprep.
- Randomized activation servers.

Changes in 2.1.0

- Desktop wallpaper displays instance information by default upon first logging on.

- PowerShell can be executed from the user data scripts by surrounding the code with the following:

```
<powershell></powershell>
```

Changes in 1.5.2.0

- For quicker launch times, the computer name is no longer updated on launch by default.
- Added support for user data scripts.
- Added documentation as an HTML file.
- Fixed an issue where an instance couldn't connect to metadata with 2 NICs.

Creating Your Own Windows AMI

When you are connected to your Amazon Windows EC2 instance, you can use it just like you use any Windows Server. There are several ways you can use your Windows instance:

- Use the instance as is for specific tasks and duration, and stop or terminate the instance when your task is done.
- Customize the instance by installing software, applications, and additional storage for specific tasks and duration. For example, you can use an Amazon Windows AMI as the base, install Visual Studio Team Foundation Server, and attach Amazon EBS volumes for additional storage. (Note that you can reboot both instance store-backed and Amazon EBS-backed instances after installing software and applications.)
- Create your own AMI from your customized instance. This new customized AMI can then be used as a base to launch multiple instances.

For information about launching, connecting, and using your Windows instance, see [Amazon EC2 Instances](#).

Before you create your own AMI, you can configure your base customized instance. The new configuration applies to all the instances that are launched from the new AMI. Your Amazon Windows instance comes with a configuration tool, EC2ConfigService. You can use this tool to configure your instance. For information on using the EC2Config Service, see [Configuring a Windows Instance Using EC2ConfigService \(p. 48\)](#)

The root storage device that you selected for the AMI determines the process you follow to create the AMI. The AMI is 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, storage, and persistence of data. For information about the differences between these AMIs, see [Root Device Storage on Windows AMIs \(p. 45\)](#).

For detailed instructions for creating an Amazon EBS-backed Windows AMI, see [Creating an Amazon EBS-Backed Windows AMI \(p. 57\)](#). For detailed instructions for creating an instance store-backed Windows AMI, see [Creating an Instance Store-Backed Windows AMI \(p. 59\)](#).

Creating an Amazon EBS-Backed Windows AMI

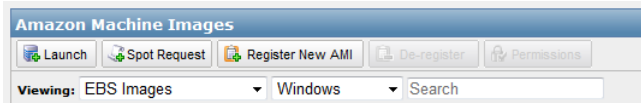
The process for creating an Amazon EBS-backed Windows AMI is simple. First you launch and customize an instance, then you create the AMI.

Amazon Elastic Compute Cloud Microsoft Windows Guide Creating an Amazon EBS-Backed Windows AMI

The process for creating an instance store-backed AMI is different. For more information, see [Creating an Instance Store-Backed Windows AMI \(p. 59\)](#).

To prepare to create an Amazon EBS-backed AMI

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the **Navigation** pane, click **AMIs**. Select an Amazon EBS-backed AMI that is similar to the AMI that you want to create. To view the Amazon EBS-backed Windows AMIs, select the following options from the **Viewing** list boxes.



You can select any public AMI that uses the version of Windows Server that you want to use for your AMI. However, you must select an Amazon EBS-backed AMI; don't start with an instance store-backed AMI.

3. Click **Launch** to launch an instance of the Amazon EBS-backed AMI that you've selected. Accept the default values as you step through the wizard.

For more information about launching a Windows instance using the AWS Management Console, see [Launch a Windows Instance \(p. 9\)](#).

4. While the instance is running, connect to it and customize it however you want. For example, you can perform any of the following actions on your instance:
 - a. Install software and applications.
 - b. Copy data.
 - c. Reduce startup time by deleting temporary files, defragmenting your hard drive, and zeroing out free space.
 - d. Create a new user account and add it to the Administrators group.
 - e. Configure settings using EC2ConfigService. For more information, see [Configuring a Windows Instance Using EC2ConfigService \(p. 48\)](#).

For information about connecting to a Windows instance using the AWS Management Console, see [Connect to Your Windows Instance \(p. 12\)](#).

5. When the instance is set up the way you want it, it is best to stop the instance before you create the AMI to ensure data integrity. If you didn't use EC2ConfigService to stop the instance already, use the following steps to stop the instance.
 - a. Right-click your running instance and select **Stop Instance**.
 - b. In the confirmation dialog box, click **Yes, Stop Instance**.

Now that you've customized your instance, you can create a Windows AMI. The following procedure describes how to create your AMI using the AWS Management Console. For information about creating your AMI by using the command line tools instead, see [ec2-create-image](#).

To create an Amazon EBS-backed AMI

1. On the **Instances** page of the EC2 console, right-click your instance and select **Create Image (EBS AMI)**.

The **Create Image** dialog box opens.

2. Enter a unique name and an optional description for the image (up to 255 characters).

3. To add an EBS volume, click **EBS Volumes**. Fill in the required information for each volume and click **Add**.

When you launch an instance from your new AMI, these additional volumes are automatically attached to the instance. Empty volumes must be formatted and mounted. Volumes based on a snapshot must be mounted.

4. To add an instance store volume, click **Instance Store Volumes**. Select the instance store volume and the device name and click **Add**.

When you launch an instance from your new AMI, these additional volumes are automatically initialized and mounted. These volumes don't contain data from the instance store volumes of the running instance from which you based your AMI.

5. Click **Yes, Create** to start creating the AMI.
6. Go to the **AMIs** page and view the status of your AMI. While your AMI is being created, its status is `pending`.

It takes a few minutes to complete the AMI creation process. When the process has completed, the status of your AMI is `available`.

7. Go to the **Snapshots** page and view the snapshot that was created for your new AMI. Any instance that you launch from your new AMI uses this snapshot for its root device volume.

Now you have created a new AMI and a snapshot. Both continue to incur charges to your AWS account until you delete them. When you are ready to delete your AMI and snapshot, you can do so using the console as follows.

To delete an AMI and a snapshot

1. Go to the **AMIs** page. Right-click the AMI and select **De-register Image**. When asked for confirmation, click **Yes, De-register**.
2. Go to the **Snapshots** page. Right-click the snapshot and select **Delete Snapshot**. When asked for confirmation, click **Yes, Delete**.

Alternatively, you can use the `ec2-deregister` command to delete an AMI and the `ec2-delete-snapshot` command to delete a snapshot.

Creating an Instance Store-Backed Windows AMI

This topic describes the process for creating an instance store-backed Windows AMI. First you launch and customize an instance, then you bundle the image, and finally you register the image.

The process for creating an Amazon EBS-backed Windows AMI is different. For more information, see [Creating an Amazon EBS-Backed Windows AMI \(p. 57\)](#).

Topics

- [Overview of Instance Store-Backed Windows AMIs \(p. 59\)](#)
- [Preparing to Create an Instance Store-Backed Windows AMI \(p. 60\)](#)
- [Bundling an Instance Store-Backed Windows AMI \(p. 61\)](#)
- [Registering an Instance Store-Backed Windows AMI \(p. 62\)](#)

Overview of Instance Store-Backed Windows AMIs

Instances launched from an AMI backed by instance store use an instance store volume as the root device volume. The image of the root device volume of an instance store-backed AMI is initially stored

in Amazon S3. When an instance is launched using an instance store-backed AMI, the image of its root device volume is copied from Amazon S3 to the root partition of the instance. The root device volume is then used to boot the instance.

When you create an instance store-backed AMI, it must be uploaded to 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.

Bundling Process

The bundling process comprises the following tasks:

- Compress the image to minimize bandwidth usage and storage requirements.
- Encrypt and sign the compressed image to ensure confidentiality and authenticate the image against its creator.
- Split the encrypted image into manageable parts for upload.
- Run `Sysprep` to strip computer-specific information (for example, the MAC address and computer name) from the Windows image to prepare it for virtualization.
- Create a manifest file that contains a list of the image parts with their checksums.
- Put all components of the AMI in the Amazon S3 bucket that you specified when making the bundle request.

Storage Volumes

It is important to remember the following details about the storage for your instance when you create an instance store-backed AMI:

- The root device volume (C:) is automatically attached when a new instance is launched from your new AMI. The data on any other instance store volumes is deleted when the instance is bundled.
- The instance store volumes other than the root device volume (for example, D:) are temporary and should be used only for short-term storage.
- You can add Amazon EBS volumes to your instance store-based instance. Amazon EBS volumes are stored within Amazon S3 buckets and remain intact when the instance is bundled. Therefore, we recommend that you store all the data that must persist on Amazon EBS volumes, not instance store volumes.

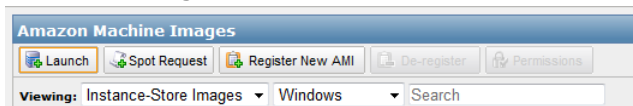
For more information about Amazon EC2 storage options, see [Storage](#).

Preparing to Create an Instance Store-Backed Windows AMI

When you create an AMI, you start by basing it on an instance. You can customize the instance to include the data and software that you need. As a result, any instance that you launch from your AMI has everything that you need.

To prepare to create an instance store-backed Windows AMI

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the **Navigation** pane, click **AMIs**. Select an instance store-backed AMI that is similar to the AMI that you want to create. To view the instance store-backed Windows AMIs, select the following options from the **Viewing** list boxes.



You can select any public AMI that uses the version of Windows Server that you want to use for your AMI. However, you must select an instance store-backed AMI; don't start with an Amazon EBS-backed AMI.

3. Click **Launch** to launch an instance of the instance store-backed AMI you've selected. Accept the default values as you step through the wizard.

For more information about launching a Windows instance using the AWS Management Console, see [Launch a Windows Instance \(p. 9\)](#).

4. While the instance is running, connect to it and customize it however you want. For example, you can perform any of the following on your instance:
 - a. Install software and applications.
 - b. Copy data.
 - c. Reduce startup time by deleting temporary files, defragmenting your hard drive, and zeroing out free space.
 - d. Create a new user account and add it to the Administrators group.
 - e. Configure settings using EC2ConfigService. For more information, see [Configuring a Windows Instance Using EC2ConfigService \(p. 48\)](#).

For information about connecting to a Windows instance using the AWS Management Console, see [Connect to Your Windows Instance \(p. 12\)](#).

Bundling an Instance Store-Backed Windows AMI

Now that you've customized your instance, you can bundle the instance to create an AMI. The following procedure describes how to bundle your AMI using the AWS Management Console. For information about bundling your AMI by using the command line tools instead, see [ec2-bundle-instance](#).

To bundle an Amazon EC2 instance store-backed AMIs

1. Determine whether you'll use an existing Amazon S3 bucket for your new AMI or create a new one. To create a new Amazon S3 bucket, use the following steps:
 - a. Open the Amazon S3 console at <https://console.aws.amazon.com/s3>.
 - b. Click **Create Bucket**.
 - c. Specify a name for the bucket and click **Create**.
2. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
3. Right-click the instance and select **Bundle Instance (instance store AMI)**.

The **Bundle Instance** dialog box opens.

4. Fill in the requested information, and then click **Bundle**.
 - a. Specify the name of an S3 bucket that you own in **Amazon S3 Bucket Name**.
 - b. Specify a prefix for the files to be generated by the bundle process in **Amazon S3 Key Name**.

The **Bundle Instance** dialog box displays a message letting you know that the request to bundle the instance succeeded, and also provides the ID of the bundle task.

Amazon EC2 shuts down the instance, bundles it, and puts the new image in the Amazon S3 bucket that you specified.

5. To view the status of the bundle task, click **View Bundling Tasks** in the **Bundle Instance** dialog box. Click **Close** to close the dialog box.

The bundle task progresses through several states, including `waiting-for-shutdown`, `bundling`, and `storing`. If the bundle task can't be completed successfully, the status is `failed`.

Registering an Instance Store-Backed Windows AMI

Finally, you must register your bundled image so that Amazon EC2 can locate it and launch instances from it.

The following procedure describes how to register your AMI using the AWS Management Console. For information about registering your AMI by using the command line tools instead, see [ec2-register](#).

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the **Navigation** pane, click **AMIs**. To view your AMIs, select **Owned By Me** from the first drop-down list for **Viewing**.
3. Right-click your newly-bundled AMI, and then click **Register New AMI**.
4. In the **Register Image** dialog box, provide the **AMI Manifest Path** and click **Register**.

Now you have created a new AMI stored in Amazon S3. You'll continue to incur charges to your AWS account until you deregister and delete the AMI.

If you make any changes to the source image stored in Amazon S3, you must deregister and register the image before the changes take effect.

Shared Windows AMIs

Shared Windows AMIs are the Windows AMIs that developers build and make available for other AWS developers to use. You can either use an available shared AMI or create your own AMI for sharing. Creating safe, secure, usable Windows AMIs for public consumption is a fairly straightforward process.

Creating Windows AMIs for Sharing

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

To create a Windows AMI for sharing, follow these guidelines:

1. Follow the instructions to launch and connect to a Windows instance.
2. Customize the instance by installing the software and applications you want to share. Do the following to make your AMI safe and secure for sharing:
 - Always delete the shell history before bundling. The shell history may contain sensitive information.
 - If you have saved your instance credentials, such as your key pair, remove them or move them to a location that is not going to be included in the AMI.
 - Ensure that the Administrator password, and passwords on any other accounts, is set to an appropriate value for sharing. These passwords will be available for anyone who launches your shared AMI.
 - Remove any saved passwords.
 - Make sure to test your AMI before you release to the public.

3. Run Sysprep to prepare the instance and enable the new password generation on new instance launch. The instance will shut down.
4. Create an image of the instance.

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 guidelines for sharing AMIs in the [Creating Windows AMIs for Sharing \(p. 62\)](#).

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 (i.e., 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 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`.

```
C:\> ec2-modify-image-attribute <ami_id> --launch-permission -a all
```

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

This example makes the `ami-2bb65342` AMI public.

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

To check the launch permissions of an AMI

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

```
C:\> ec2-describe-image-attribute <ami_id> -l
```

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

Amazon Elastic Compute Cloud Microsoft Windows Guide Creating Windows AMIs for Sharing

```
C:\> ec2-describe-image-attribute ami-2bb65342 -l
launchPermission      ami-2bb65342      group    all
```

To make an AMI private again

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

```
C:\> 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.

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

```
C:\> 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:

```
C:\> 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.

The following example grants launch permissions to the AWS account with ID `111122223333` for the `ami-2bb65342` AMI:

```
C:\> ec2-modify-image-attribute ami-2bb65342 -l -a 111122223333
launchPermission      ami-2bb65342      ADD     userId   111122223333
```

To remove launch permissions for an account

- Enter the following command:

```
C:\> 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.

The following example removes launch permissions from the AWS account with ID `111122223333` for the `ami-2bb65342` AMI:

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Creating Windows AMIs for Sharing**

```
C:\> ec2-modify-image-attribute ami-2bb65342 -l -r 111122223333  
launchPermission      ami-2bb65342      REMOVE      userId 111122223333
```

To remove all launch permissions

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

```
C:\> ec2-reset-image-attribute <ami_id> -l
```

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

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

```
C:\> ec2-reset-image-attribute ami-2bb65342 -l  
launchPermission      ami-2bb65342      RESET
```

Note

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

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 your AMI 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 feature
 - 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 ID: </strong>[ami-id]<br />  
<strong>AMI Manifest: </strong>[myawsbucket/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 />
```


- The following command displays a list of AMIs for which you have explicit launch permissions. AMIs that you own are excluded from the list.

```
C:\> ec2dim -x self
```

- The following command displays a list of AMIs owned by Amazon.

```
C:\> ec2dim -o amazon
```

- The following command displays a list of AMIs owned by a particular AWS account.

```
C:\> ec2dim -o <target_uid>
```

The `<target_uid>` is the account ID that owns the AMIs you're looking for.

For more information about the flags and how to use flags 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 (such as a website or another EC2 user that you trust). If you do not know the source of an AMI, we recommend that you search the AWS 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.

For information on launching, connecting, and using the Windows instances, see [Using Instances](#).

Paid Windows AMIs

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.

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](#).

Note

All paid AMIs from Amazon DevPay are backed by Amazon instance store. At this time, AWS Marketplace does not support paid Windows AMIs.

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:

```
C:\> 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.

This example shows an `ec2-describe-images` call describing a paid AMI. The product code is ACD42B6F.

```
C:\> 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, go to [On-Demand and Reserved Instances](#).

Launch 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.

To launch a paid AMI

- Enter the following command:

```
C:\> ec2-run-instances <ami_id>
```

The `<ami_id>` is the AMI ID.

This example shows the command used to launch the ami-2bb65342 AMI.

```
C:\> ec2-run-instances ami-2bb65342
RESERVATION r-a034c7c9 111122223333 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
```

Note

The owner of a paid AMI will be able to confirm if a particular instance was launched using their paid AMI.

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, go to [On-Demand and Reserved Instances](#).

To associate the product code with your AMI

- Enter the `ec2-modify-image-attribute` command:

```
C:\> 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.

The following command launches the *ami-2bb65342* paid AMI.

```
C:\> ec2-run-instances ami-2bb65342  
RESERVATION r-a034c7c9 111122223333 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.

For information on the usage information for your paid and supported AMIs , go to [amazonpayments](#)) sign in page.

Setting Up a Windows HPC Cluster on Amazon EC2

This section steps you through how to launch a scalable Microsoft Windows High Performance Computing (HPC) cluster using only Amazon Elastic Compute Cloud (Amazon EC2) instances. A Windows HPC cluster requires an Active Directory domain controller and a DNS server, a head node, and one or more compute nodes. By following the steps in this section, you can assemble each of these components and launch a Windows HPC cluster. For more information on High Performance Computing, go to [High Performance Computing \(HPC\) on AWS](#).

Process for Setting Up a Windows HPC Cluster on Amazon EC2

Task 1: Set Up Your Active Directory Domain Controller (p. 72)
Task 2: Configure Your Head Node (p. 73)
Task 3: Set Up the Compute Node (p. 75)
Task 4: Scale Your HPC Compute Nodes (Optional) (p. 77)

Prerequisites

Before you begin to configure the instances for your Windows HPC cluster, make sure that the following requirements are met:

- Open an Amazon EC2 account, if you haven't already, and set up permissions to launch new EC2 instances. For more information, see [Sign Up for AWS](#).
- Before you begin the configuration in a specific region, check the [Amazon EC2 pricing page](#) and select the drop-down list for that region to see if Cluster Compute Instances are available in that region.
- Install the Amazon EC2 command line tools. For more information, go to [Installing the Amazon EC2 Command Line Tools on Windows \(p. 81\)](#).
- Optionally, you can download the [HPC Pack 2008 R2](#). You can also download HPC Pack 2008 R2 Express directly to your AMI instance later.

Task 1: Set Up Your Active Directory Domain Controller

The Active Directory domain controller provides authentication and centralized resource management of the HPC environment and is required for the installation. Setting up your Active Directory involves three steps:

1. Creating security groups for Active Directory.
2. Launching an instance for your domain controller.
3. Configuring your domain controller for your HPC cluster.

Setting Up Security Groups for Active Directory

Run the security group script `create-AD-sec-groups.bat` to create the rules for the domain controller and domain members. If you have not installed the command line tools, manually create a security group with the port requirements for Windows Server 2008/Windows Server 2008 R2. For more information, go to [How to configure a firewall for domains and trusts](#) on the Microsoft website.

To create the required security groups for Active Directory

1. Using a text editor, copy the contents of the [create_AD_security.bat \(p. 78\)](#), and save the file with the name `create-AD-sec-groups.bat` to a computer configured with the Amazon EC2 command line tools from which you connect to Amazon Web Services.
2. Run the file as a local administrator.
3. Log in to the AWS Management Console and verify that the following security groups appear: SG - Domain Controller and SG - Domain Member.

Launch an Instance for Your Domain Controller

Configure your domain controller by launching an instance from AWS and then configuring the instance as a domain controller for your HPC cluster.

To launch an instance for your domain controller

1. Launch an `m1.large` Amazon EC2 instance type from Microsoft Windows Server 2008 R2 Base (you could use another instance type depending on your anticipated usage) with the name **Domain Controller** and assign it to the **SG - Domain Controller** security group.
2. Create an Elastic IP address and associated this IP address with the Domain Controller instance.
 - a. In the navigation pane, click **Elastic IPs**.
 - b. Click **Allocate New Address**.
 - c. In the **Allocate New Address** dialog box, click **Yes Allocate**.
 - d. Select the Elastic IP address you created, and then click **Associate Address**.
 - e. In the **Associate Address** dialog box, in the **Instance** drop-down list, select the domain controller instance and then click **Yes Associate**.

Configure Your Domain Controller for Your HPC Cluster

Next, log in to the instance you created and configure the server as a domain controller for the HPC cluster.

To configure your instance as a domain controller

1. Connect to your instance.
2. Open **Server Manager**, and add the Active Directory Domain Services role.
3. Promote the server to a domain controller using Server Manager or by running **DCPromo.exe**.
4. Create a new domain in a new forest.
5. Enter hpc.local as the fully qualified domain name (FQDN).
6. Select Forest Functional Level as **Windows Server 2008 R2**.
7. Ensure that the DNS Server option is selected, and then click **Next**.
8. Select **Yes, the computer will use an IP address automatically assigned by a DHCP server (not recommended)**.
9. In the warning box, click **Yes** to continue.
10. Complete the wizard and then select **Reboot on Completion**.
11. Log in to the instance as hpc.local\administrator.
12. Create a domain user hpc.local\hpcuser.

Task 2: Configure Your Head Node

HPC clients all connect to the head node. The head node facilitates the scheduled jobs. You configure your head node by:

1. Creating security groups for your HPC cluster.
2. Launching an instance for your head node.
3. Installing the HPC Pack.
4. Configuring your cluster.

Creating Security Groups for Your HPC Cluster

Run the security group script `create-HPC-sec-group.bat` to create a security group named **SG - Windows HPC Cluster** with the rules for the HPC cluster nodes. If you have not installed the command line tools, manually create a security group configure with the port requirements for HPC cluster members to communicate only within this security group. For more information, go to [Windows Firewall](#) on the Microsoft website.

To create the required security groups for your HPC cluster

1. Using a text editor, copy the contents of the [create-HPC-sec-group.bat](#) (p. 79), and save the file with the name `create-HPC-sec-group.bat` to a computer configured with the EC2 command line tools from which you connect to Amazon Web Services.
2. Run the file as a local administrator.

3. Log in to AWS Management Console and verify that the security group SG - Windows HPC Cluster appears.

Launch an Instance for the HPC Head Node

Configure your head node by launching a cluster instance from AWS and then configuring the instance as a domain member of the hpc.local and with the necessary user accounts.

To configure an instance for your head node

1. Launch an instance from **Microsoft Windows 2008 R2 64-bit for Cluster Instances** with the name **HPC-Head** and assign the instance to both the **SG - Windows HPC Cluster** and **SG - Domain Member** security groups.
2. Log in to the instance and get the existing DNS server address from **HPC-Head** using **IPConfig /all**.
3. Update the TCP/IPv4 properties of the **HPC-Head** NIC to include the **Domain Controller** Elastic IP address as the primary DNS and then add the additional DNS IP address from the previous step.
4. Join the machine to the hpc.local domain using hpc.local\administrator credentials (the domain administrator account).
5. Add hpc.local\hpcuser as the local administrator. When prompted for credentials, use hpc.local\administrator, and then restart.
6. Log back in to **HPC-Head** as hpc.local\hpcuser.

Install the HPC Pack

This section explains how to download and install the HPC Pack.

To install the HPC Pack

1. Connect to your **HPC-Head** instance using the hpc.local\hpcuser account.
2. Using **Server Manager**, turn off Internet Explorer Enhanced Security Configuration (IE ESC) for Administrators.
 - a. In **Server Manager**, under **Security Information**, click **Configure IE ESC**.
 - b. Turn off IE ESC for administrators.
3. Install the HPC Pack 2008 R2 Express on **HPC-Head**.
 - a. Download HPC Pack 2008 R2 Express onto **HPC-Head** from <http://go.microsoft.com/fwlink/?LinkID=198084>.
 - b. Extract the files to a folder, open the folder, and double-click **setup.exe**.
 - c. Select **HPC Pack 2008 R2 Express**, and then click **Next**.
 - d. Accept the licensing agreement if you agree, and then click **Next**.
 - e. On the Installation page, select **Create a new HPC cluster by creating a head node**, and then click **Next**.
 - f. Accept the default settings to install all the databases on the Head Node, and then click **Next**.
 - g. Complete the wizard.

Configure Your HPC Cluster on the Head Node

This section explains how to configure your HPC cluster on the head node.

To configure your HPC cluster on the head node

1. Start **HPC Cluster Manager**.
2. In the **Deployment To-Do List**, select **Configure your network**.
 - a. In the wizard, select the default option (5), and then click **Next**.
 - b. Complete the wizard accepting default values on all screens, and choose how you want to update the server and participate in customer feedback.
 - c. Click **Configure**.
3. Select **Provide Network Credentials**, then supply the hpc.local\hpcuser credentials.
4. Select **Configure the naming of new nodes**, and then click **OK**.
5. Select **Create a node template**.
 - a. Select the **Compute node template**, and then click **Next**.
 - b. Select **Without operating system**, then continue with the defaults.
 - c. Click **Create**.

Task 3: Set Up the Compute Node

Setting up the compute node involves the following steps:

1. Launching an instance for your compute node.
2. Installing the HPC Pack on the instance.
3. Adding the compute node to your cluster.

Launch an Instance for the HPC Compute Node

Configure your compute node by launching a cluster instance from AWS, and then configuring the instance as a domain member of hpc.local with the necessary user accounts.

To configure an instance for your compute node

1. Launch an instance from **Microsoft Windows 2008 R2 64-bit for Cluster Instances** with the name **HPC-Compute** and assign the instance to both **SG - Windows HPC Cluster** and **SG - Domain Member** security groups.
2. Log in to the instance and get the existing DNS server address from **HPC-Compute** using **IPConfig /all**.
3. Update the TCP/IPv4 properties of the **HPC-Compute** NIC to include the Domain Controller Elastic IP address as the primary DNS and then add the additional DNS IP address from the previous step.
4. Join the machine to the hpc.local domain using hpc.local\administrator credentials (the domain administrator account).

5. Add hpc.local\hpcuser as the local administrator. When prompted for credentials, use hpc.local\administrator, and then restart.
6. Log back in to **HPC-Compute** as hpc.local\hpcuser.

Install the HPC Pack on the Compute Node

This section explains how to download and install the HPC Pack on the compute node for your HPC cluster.

To install the HPC Pack on the compute node

1. Connect to your **HPC-Compute** instance using the hpc.local\hpcuser account.
2. Using **Server Manager**, turn off Internet Explorer Enhanced Security Configuration (IE ESC) for Administrators.
 - a. In **Server Manager**, under **Security Information**, click **Configure IE ESC**.
 - b. Turn off IE ESC for administrators.
3. Install the HPC Pack 2008 R2 Express on **HPC-Compute**.
 - a. Download HPC Pack 2008 R2 Express onto **HPC-Compute** from <http://go.microsoft.com/fwlink/?LinkID=198084>.
 - b. Extract the files to a folder, open the folder, and double-click **setup.exe**.
 - c. Select **HPC Pack 2008 R2 Express**, and then click **Next**.
 - d. Accept the licensing agreement if you agree, and then click **Next**.
 - e. On the Installation page, select **Join an existing HPC cluster by creating a new compute node**, and then click **Next**.
 - f. Specify the machine name FQDN of the **HPC-Head** instance, and then choose the defaults.
 - g. Complete the wizard.

Add the Compute Node to Your HPC Cluster

To complete your cluster configuration, from the head node, add the compute node to your cluster.

To add the compute node to your cluster

1. Log in to the **HPC-Head** as hpc.local\hpcuser.
2. On **HPC-Head**, open **HPC Cluster Manager**.
3. Select **Node Management** in the bottom-left pane.
4. If the compute node displays in the **Unapproved** bucket, then right-click the node that is listed and select **Add Node**.
 - a. Select **Add compute nodes or broker nodes that have already been configured**.
 - b. Select the check box next to the node and click **Add**.
5. Right-click the node and click **Bring Online**.

Task 4: Scale Your HPC Compute Nodes (Optional)

To scale your compute nodes

1. Log in to **HPC-Compute** as `hpc.local\hpcuser`.
2. Delete any files you downloaded locally from the HP Pack 2008 R2 Express installation package. (You have already run setup and created these files on your image so they do not need to be cloned for an AMI.)
3. From `C:\Program Files\Amazon\Ec2ConfigService` open the file, `sysprep2008.xml`.
4. At the bottom of `<settings pass="specialize">`, add the following section – make sure to replace `hpc.local`, `password` and `hpcuser` to match your environment.

```
<component name="Microsoft-Windows-UnattendedJoin" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.microsoft.com/WMICConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Identification>
    <UnsecureJoin>>false</UnsecureJoin>
    <Credentials>
      <Domain>hpc.local</Domain>
      <Password>Password</Password>
      <Username>hpcuser</Username>
    </Credentials>
    <JoinDomain>hpc.local</JoinDomain>
  </Identification>
</component>
```

5. Save `sysprep2008.xml`.
6. Click **Start**, point to **All Programs**, and then click **EC2ConfigService Settings**.
 - a. Click the **General** tab, and clear the **Set Computer Name** check box.
 - b. Click the **Bundle** tab, and then click **Run Sysprep and Shutdown Now**.
7. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
8. In **Navigation**, click **Instances**.
9. Wait for the instance status to show **Stopped**.
10. Right-click the instance, and select **Create Image (EBS AMI)**.
11. Specify an image name and image description, and then click **Create This Image** to create an AMI from the instance.
12. Start the original **HPC-Compute** node that was shut down.
13. Connect to the head node using the `hpc.local\hpcuser` account.
14. From **HPC Cluster Manager**, delete the old node that now appears in an error state.
15. In the AWS Management Console, in **Navigation**, click **AMIs**.
16. Use the AMI you created to add additional nodes to the cluster.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
Running the Lizard Performance Measurement
Application**

Any number of additional compute nodes can now be launched from the AMI that was created. The nodes are automatically joined to the domain, but you must add them to the cluster as already configured nodes in **HPC Cluster Manager** using the head node and then bring them online.

Running the Lizard Performance Measurement Application

If you choose, you can run the Lizard application, which measures the computational performance and efficiency that can be achieved by your HPC cluster. Go to <http://www.microsoft.com/download/en/details.aspx?id=8433>, download the lizard_x64.msi installer and run it directly on your head node as hpc.localhpcuser.

create_AD_security.bat

The following .bat file creates two security groups for your Active Directory environment: one group for Active Directory domain controllers and one for Active Directory domain member servers.

```
set DC="SG - Domain Controller"
set DM="SG - Domain Member"

:: =====
:: Creates Security groups Prior to Adding Rules
:: =====

call ec2addgrp %DM% -d "Active Directory Domain Member"
call ec2addgrp %DC% -d "Active Directory Domain Controller"

:: =====
:: Security group for Domain Controller
:: =====

:: For LDAP and related services. Details at link below
:: http://support.microsoft.com/kb/179442
call ec2auth %DC% -o %DM% -P UDP -p 123
call ec2auth %DC% -o %DM% -P TCP -p 135
call ec2auth %DC% -o %DM% -P UDP -p 138
call ec2auth %DC% -o %DM% -P TCP -p "49152-65535"
call ec2auth %DC% -o %DM% -P TCP -p 389
call ec2auth %DC% -o %DM% -P UDP -p 389
call ec2auth %DC% -o %DM% -P TCP -p 636
call ec2auth %DC% -o %DM% -P TCP -p 3268
call ec2auth %DC% -o %DM% -P TCP -p 3269
call ec2auth %DC% -o %DM% -P TCP -p 53
call ec2auth %DC% -o %DM% -P UDP -p 53
call ec2auth %DC% -o %DM% -P TCP -p 88
call ec2auth %DC% -o %DM% -P UDP -p 88
call ec2auth %DC% -o %DM% -P TCP -p 445
call ec2auth %DC% -o %DM% -P UDP -p 445
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
create-HPC-sec-group.bat**

```
:: For ICMP as required by Active Directory
call ec2auth %DC% -P ICMP -t -1:-1

:: For Elastic IP to communicate with DNS
call ec2auth %DC% -s 0.0.0.0/0 -P UDP -p 53

:: For RDP for connecting to desktop remotely
call ec2auth %DC% -P TCP -p 3389

:: =====
:: Security group for Domain Member
:: =====

:: For LDAP and related services. Details at link below
:: http://support.microsoft.com/kb/179442

call ec2auth %DM% -o %DC% -P TCP -p "49152-65535"
call ec2auth %DM% -o %DC% -P UDP -p "49152-65535"
call ec2auth %DM% -o %DC% -P TCP -p 53
call ec2auth %DM% -o %DC% -P UDP -p 53
```

create-HPC-sec-group.bat

The following .bat file creates a security group for your HPC cluster nodes. Run this bat file from the client computer from which you are connecting to Amazon Web Services.

```
set HPC="SG - Windows HPC Cluster"

:: =====
:: Creates Security groups Prior to Adding Rules
:: =====

call ec2addgrp %HPC% -d "Windows HPC Server 2008 R2 Cluster Nodes"

:: =====
:: Security group for Windows HPC Cluster
:: =====

:: For HPC related services. Details at link below
:: http://technet.microsoft.com/en-us/library/ff919486(WS.10).aspx#BKMK_Firewall
call ec2auth %HPC% -o %HPC% -P TCP -p 80
call ec2auth %HPC% -o %HPC% -P TCP -p 443
call ec2auth %HPC% -o %HPC% -P TCP -p 1856
call ec2auth %HPC% -o %HPC% -P TCP -p 5800
call ec2auth %HPC% -o %HPC% -P TCP -p 5801
call ec2auth %HPC% -o %HPC% -P TCP -p 5969
call ec2auth %HPC% -o %HPC% -P TCP -p 5970
call ec2auth %HPC% -o %HPC% -P TCP -p 5974
call ec2auth %HPC% -o %HPC% -P TCP -p 5999
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
create-HPC-sec-group.bat**

```
call ec2auth %HPC% -o %HPC% -P TCP -p 6729
call ec2auth %HPC% -o %HPC% -P TCP -p 6730
call ec2auth %HPC% -o %HPC% -P TCP -p 7997
call ec2auth %HPC% -o %HPC% -P TCP -p 8677
call ec2auth %HPC% -o %HPC% -P TCP -p 9087
call ec2auth %HPC% -o %HPC% -P TCP -p 9090
call ec2auth %HPC% -o %HPC% -P TCP -p 9091
call ec2auth %HPC% -o %HPC% -P TCP -p 9092
call ec2auth %HPC% -o %HPC% -P TCP -p "9100-9163"
call ec2auth %HPC% -o %HPC% -P TCP -p "9200-9263"
call ec2auth %HPC% -o %HPC% -P TCP -p 9794
call ec2auth %HPC% -o %HPC% -P TCP -p 9892
call ec2auth %HPC% -o %HPC% -P TCP -p 9893
call ec2auth %HPC% -o %HPC% -P UDP -p 9893

:: For HPC related services, these are NOT in the first table but are there in
the third table at link below
:: http://technet.microsoft.com/en-us/library/ff919486\(WS.10\).aspx#BKMK\_Firewall
call ec2auth %HPC% -o %HPC% -P TCP -p 6498
call ec2auth %HPC% -o %HPC% -P TCP -p 7998
call ec2auth %HPC% -o %HPC% -P TCP -p 8050
call ec2auth %HPC% -o %HPC% -P TCP -p 5051

:: For RDP for connecting to desktop remotely
call ec2auth %HPC% -P TCP -p 3389
```

Installing the Amazon EC2 Command Line Tools on Windows

These instructions describe how to install the Amazon EC2 command line tools, a set of tools that you can run from a Windows Command Prompt window that closely wrap the Amazon EC2 API actions.

Process for Installing the Command Line API Tools

Task 1: Download the Command Line Tools (API Tools) (p. 81)
Task 2: Set the JAVA_HOME Environment Variable (p. 82)
Task 3: Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables (p. 83)
Task 4: Set the EC2_HOME Environment Variable (p. 84)
Task 5: Set the Region (p. 85)
Task 6: Download Remote Desktop (p. 85)

Note

These instructions are written for a Windows 7 client. What you need to do to complete some steps may vary if you're using a different version of Windows.

Task 1: Download the Command Line Tools (API Tools)

The command line tools are available as a ZIP file on this site: [Amazon EC2 API Tools](#). The 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. You can simply download the file and unzip it.

Some additional setup is required before you can use the tools. These steps are discussed next.

Task 2: Set the JAVA_HOME Environment Variable

The Amazon EC2 command line tools read an environment variable (JAVA_HOME) computer to locate the Java runtime.

To set the JAVA_HOME environment variable

1. If you don't have Java 1.6 or later installed, download and install Java. Either a JRE or JDK installation is acceptable. To view and download JREs for a range of platforms, go to <http://java.oracle.com/>.
2. Set JAVA_HOME to the full path of the Java home directory. This is the directory that contains the bin directory that contains the Java executable you installed (java.exe). For example, if your Java executable is in C:\jdk\bin, set JAVA_HOME to C:\jdk.
 - a. On the computer you'll use to connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.
 - b. Click **Advanced system settings**.
 - c. Click **Environment Variables**.
 - d. Under **System variables**, click **New**.
 - e. In **Variable name**, type JAVA_HOME.
 - f. In **Variable value**, type the path to your Java home directory (enclosing the path in quotation marks if the path contains spaces). For example, "C:\Program Files (x86)\Java\jre7"
3. Open a new Command Prompt window and verify your JAVA_HOME setting using this command.

```
C:\> %JAVA_HOME%\bin\java -version
```

If you've set the environment variable correctly, the output looks something like this.

```
java version "1.7.0_05"  
Java(TM) SE Runtime Environment (build 1.7.0_05-b05)  
Java HotSpot(TM) Client VM (build 23.1-b03, mixed mode, sharing)
```

Otherwise, check the setting of JAVA_HOME, fix any errors, open a new Command Prompt window, and try the command again.

- a. In **System variables**, select **Path**, and then click **Edit**.
 - b. In **Variable values**, before any other versions of Java add ;%JAVA_HOME%\bin.
5. Open a new Command Prompt window and verify your update to the Path environment variable using this command.

```
C:\> java -version
```

You should see the same output as before. Otherwise, check the setting of `Path`, fix any errors, open a new Command Prompt Window, and try the command again.

Task 3: Set the `AWS_ACCESS_KEY` and `AWS_SECRET_KEY` Environment Variables

When you sign up for an AWS account, we create access credentials for you so that you can make secure requests to AWS. You must provide these credentials to the Amazon EC2 command line tools so that they know that the commands that you issue come from your account.

The following procedure describes how you can view your access credentials.

To view your AWS access credentials

1. Go to the Amazon Web Services website at <http://aws.amazon.com>.
2. Click **My Account/Console**, and then click **Security Credentials**.
3. Under **Your Account**, click **Security Credentials**.
4. In the spaces provided, type your user name and password, and then click **Sign in using our secure server**.
5. Under **Access Credentials**, on the **Access Keys** tab, your access key ID is displayed. To view your secret key, under **Secret Access Key**, click **Show**.

You can specify these credentials with the `--aws-access-key` and `--aws-secret-key` (or `-O` and `-W`) options every time you issue a command. However, it's easier to specify your access credentials using the following environment variables:

- `AWS_ACCESS_KEY`—Your access key ID
- `AWS_SECRET_KEY`—Your secret access key

If these environment variables are set properly, their values serve as the default values for these required options, so you can omit them from the command line.

Note

Although we don't encourage it, for a limited time you can still use `EC2_PRIVATE_KEY` and `EC2_CERT` instead of `AWS_ACCESS_KEY` and `AWS_SECRET_KEY`. For more information, see *Deprecated Options* in [Common Options for API Tools](#) in the *Amazon Elastic Compute Cloud CLI Reference*. If you specify both sets of credentials, the command line tools use the access key ID and secret access key.

The following procedure describes how to create environment variables that specify your access credentials.

To set up your environment variables

1. On the computer you'll use to connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.
2. Click **Advanced system settings**.
3. Click **Environment Variables**.
4. Under **System variables**, click **New**.

5. In **Variable name**, type `AWS_ACCESS_KEY`.
6. In **Variable value**, specify your access key ID.
7. Under **System variables**, click **New**.
8. In **Variable name**, type `AWS_SECRET_KEY`.
9. In **Variable value**, specify your secret access key.

Task 4: Set the `EC2_HOME` Environment Variable

The Amazon EC2 command line tools read an environment variable (`EC2_HOME`) to locate supporting libraries. You'll need to set this environment variable before you can use the tools.

To set the `EC2_HOME` environment variable

1. Set `EC2_HOME` to the path of the directory into which you unzipped the command line tool. This directory is named `ec2-api-tools-w.x.y.z` (where `w`, `x`, `y`, and `z` are components of the version number). It contains sub-directories named `bin` and `lib`.
 - a. On the computer you'll use to connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.
 - b. Click **Advanced system settings**.
 - c. Click **Environment Variables**.
 - d. Under **System variables**, click **New**.
 - e. In **Variable name**, type `EC2_HOME`.
 - f. In **Variable value**, type the path to the directory where you installed the command line tools. For example, `C:\AWS\EC2\ec2-api-tools-1.5.4.0`.
2. Open a new Command Prompt window and verify your `EC2_HOME` setting using this command.

```
C:\> dir %EC2_HOME%
```

If you've set the environment variable correctly, you'll see output for the directory listing. If you get a File Not Found error, check the setting of `EC2_HOME`, fix any errors, open a new Command Prompt window, and try the command again.

3. Add the `bin` directory for the tools to your system `Path` environment variable. The rest of this guide assumes that you've done this.

You can update your `Path` as follows:

- a. In **System variables**, select **Path**, and then click **Edit**.
 - b. In **Variable values**, add `;%EC2_HOME%\bin`.
4. Open a new Command Prompt window and verify your update to the `Path` environment variable using this command.

```
C:\> ec2-describe-regions
```

If all your environment variables are set correctly, you'll see output that looks something like this.

REGION	eu-west-1	ec2.eu-west-1.amazonaws.com
REGION	sa-east-1	ec2.sa-east-1.amazonaws.com
REGION	us-east-1	ec2.us-east-1.amazonaws.com
REGION	ap-northeast-1	ec2.ap-northeast-1.amazonaws.com
REGION	us-west-2	ec2.us-west-2.amazonaws.com
REGION	us-west-1	ec2.us-west-1.amazonaws.com
REGION	ap-southeast-1	ec2.ap-southeast-1.amazonaws.com

If you get an error that this command is not recognized as an internal or external command, check the setting of `Path`, fix any errors, open a new Command Prompt window, and try the command again.

If you get an error that required option `-O` is missing, check the setting of `AWS_ACCESS_KEY`, fix any errors, open a new Command Prompt window, and try the command again.

If you get an error that required option `-W` is missing, check the setting of `AWS_SECRET_KEY`, fix any errors, open a new Command Prompt window, and try the command again.

Task 5: Set the Region

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. If your instances are in a different region, you must specify the region where your instances reside. For example, if your instances are in Europe, you must specify the `eu-west-1` Region by using the `--region eu-west-1` option or by setting the `EC2_URL` environment variable.

This section describes how to specify a different region by changing the service endpoint URL.

To specify a different region

1. To view available Regions, see [Regions and Endpoints](#) in the *Amazon Web Services General Reference*.
2. To change the service endpoint, set the `EC2_URL` environment variable.

The following example sets `EC2_URL` to the EU (Ireland) Region.

- a. On the computer you'll use to connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.
- b. Click **Advanced system settings**.
- c. Click **Environment Variables**.
- d. Under **System variables**, click **New**.
- e. In **Variable name**, type `EC2_URL`.
- f. In **Variable value**, type `https://ec2.eu-west-1.amazonaws.com`.

Task 6: Download Remote Desktop

Some of the examples in this guide require a Remote Desktop client. Most recent versions of Windows include a Remote Desktop client already. To check whether you have one, open a Command Prompt window and type `mstsc`. If this command displays the Remote Desktop Connection window, you're set.

Otherwise, go to the [Microsoft Windows home page](#) and search for the download for Remote Desktop Connection.

Now you're ready to start using Amazon EC2 from a Command Prompt window!

Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET

Amazon Web Services provides the AWS SDK for .NET as one of the options for communicating with the Amazon Elastic Compute Cloud (Amazon EC2) API. Windows PowerShell can use the .NET SDK as well. This increases your options for script development for AWS. In this topic you will learn how to configure Windows PowerShell to work with the .NET SDK and explore code samples that will help get you started. If you have not installed PowerShell, you will need to do so before you begin.

Configuring PowerShell to Work with the AWS SDK for .NET

In addition to the typical Windows PowerShell requirements, the following requirements will need to be met:

1. Install the AWS .NET SDK on the client computer that you will use to connect to AWS. Install the SDK from <http://aws.amazon.com/sdkfornet>.
2. Have your AWS account public access key ID and secret key available.

To view your AWS access credentials

1. Go to the Amazon Web Services website at <http://aws.amazon.com>.
2. Click **My Account/Console**, and then click **Security Credentials**.
3. Under **Your Account**, click **Security Credentials**.
4. In the spaces provided, type your user name and password, and then click **Sign in using our secure server**.
5. Under **Access Credentials**, on the **Access Keys** tab, your access key ID is displayed. To view your secret key, under **Secret Access Key**, click **Show**.

Code Samples

The following code samples give you an introduction to using PowerShell with the .NET SDK. To use the code samples copy the scripts to your local machine, save with a .ps1 extension, then run them in PowerShell.

Sample 1: Export a List of All Amazon AMIs

A limited number of Amazon Machine Images (AMIs) are displayed in the AWS Management Console. The following PowerShell script exports a complete list of AMIs from all regions into a .CSV file and saves the file in the directory where you saved the scripts. You can then open the .csv file in Excel and search or filter across the complete set of available AMIs to find a specific AMI.

To use the script

1. Copy the contents of the [EC2_ListOfAMIs.ps1 \(p. 90\)](#) into a text editor and save the file with .ps1 extension.
2. Open a PowerShell window and run `.\EC2_ListOfAMIs.ps1 -EC2AccessKey <Your Access Key ID> -EC2SecretKey <Your Secret Key>`. For example:

```
.\EC2_ListOfAMIs.ps1 -EC2AccessKey AKIAIOSFODNN7EXAMPLE -EC2SecretKey  
wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
```

3. Navigate to the directory where you saved the scripts and open the AMIList.csv file exported by the script.

Sample 2: Launch an Instance Using a Specified Amazon Machine Image (AMI)

When Amazon Machine Images are updated, the previous AMI is depreciated and a new AMI replaces the previous one. The AMI ID changes when the AMI is updated. To track AMIs, use the Name field. When an AMI is updated, the basic structure of the AMI name usually remains the same, but the name has a new date appended to the end.

You can use the PowerShell script [E2_CreateInstances.ps1 \(p. 91\)](#) to match the name you provide to find the corresponding AMI, confirm the AMI returned by the script, and then launch an Amazon EC2 instance using the AMI.

To use the script:

1. Create a key pair to associate with your instance. You'll need the private key file to retrieve your initial Windows password and connect to your instance later.
 - a. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. Under **Security**, click **Key Pairs**. You can use an existing key pair or create a new key pair. For this example, we'll create a new key pair.
 - c. Click **Create Key Pair**.
 - d. Enter a name for your key pair, and then click **Create**.
 - e. Download the .pem file and save it to a secure location. You'll need the contents of this key pair to retrieve the password to connect to your instance.

**Amazon Elastic Compute Cloud Microsoft Windows
Guide**
**Sample 2: Launch an Instance Using a Specified Amazon
Machine Image (AMI)**

2. Copy the contents of the [E2_CreateInstances.ps1 \(p. 91\)](#) into a text editor and save the file with a .ps1 extension.
3. Open a PowerShell window and run one of the following example scripts:

Example 1:

```
.\EC2_CreateInstances.ps1 -EC2AccessKey AKIAIOSFODNN7EXAMPLE -EC2SecretKey  
wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY -matchDescription 'Windows_Server-  
2008-SP2-English-64Bit-Base-2012' -RegionName us-east-1 -InstanceType  
t1.micro -KeyPairName MyKeyPair
```

Example 2:

```
.\EC2_CreateInstances.ps1 -EC2AccessKey AKIAIOSFODNN7EXAMPLE -EC2SecretKey  
wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY -matchDescription 'Windows_Server-  
2008-R2-English-64Bit-Base-2012' -RegionName us-east-1 -InstanceType t1.micro  
-InstancesToLaunch 1 -SecurityGroup Default -KeyPairName MyKeyPair -NameT  
oTagCreatedInstance "This is my tag"
```

Parameters

Name	Description
<i>EC2AccessKey</i>	The access key associated with your AWS account.
<i>EC2SecretKey</i>	The secret access key associated with your AWS account.
<i>matchDescription</i>	The description of the AMI that will be used to create an instance.
<i>RegionName</i>	The region in which to search for the AMIs.
<i>InstanceType</i>	The type of EC2 instance that you want to launch. By default the instance launches with as a t1.micro. For a list of valid instance type names, see Available Instance Types in the <i>Amazon Elastic Compute Cloud User Guide</i> .
<i>InstancesToLaunch</i> <i>[optional]</i>	The number of EC2 instances to launch for each of the matched AMIs. If you do not specify a number, the value defaults to 1 and a single instance is launched.
<i>SecurityGroup [optional]</i>	The security group that you want to launch the instance with. If you do not specify this parameter, the instance is launched with default security group in your region.
<i>KeyPairName</i>	Enter the name of the key pair you created previously.
<i>NameToTagCreatedInstance</i>	A tag used by the AWS Management Console to help identify the instance. For more information about tags, see Using Tags in the <i>Amazon Elastic Compute Cloud User Guide</i> .

EC2_ListOfAMIs.ps1

The following PowerShell script creates CSV file that lists of all Amazon Machine Images (AMIs) running the Windows operating system.

```
<#
.DESCRIPTION
    Outputs a list of all Images (AMI's) for each region to CSV into the current
    folder.

.NOTES
    PREREQUISITES:
        1) Download the SDK library from http://aws.amazon.com/sdkfornet/
        2) Obtain Secret and Access keys from https://aws-
portal.amazon.com/gp/aws/developer/account/index.html?action=access-key

    API Reference:http://docs.amazonwebservices.com/AWSEC2/latest/APIRefer
ence/query-apis.html

.EXAMPLE
    powershell.exe .\EC2_ListOfAMIs.ps1 -EC2AccessKey ThisIsMyAccessKey -
EC2SecretKey ThisIsMySecretKey
#>

##Incoming Parameters
Param(
    [parameter(mandatory=$true)][string]$EC2AccessKey,
    [parameter(mandatory=$true)][string]$EC2SecretKey
)

##Creates and defines the objects for the Account
$AccountInfo = New-Object PSObject -Property @{
    EC2AccessKey = $EC2AccessKey
    EC2SecretKey = $EC2SecretKey
}

##Removes unused variables
Remove-Variable EC2AccessKey
Remove-Variable EC2SecretKey

##Declare Variables
$ListOfAllAMIs = @() #Creates an array to store all the AMIs to -- this will
include the Region information

##Loads the SDK information into memory
$SDKLibraryLocation = dir C:\Windows\Assembly -Recurse -Filter "AWSSDK.dll"
$SDKLibraryLocation = $SDKLibraryLocation.FullName
Add-Type -Path $SDKLibraryLocation

#Sets the end-point for the different regions
$config = New-Object Amazon.EC2.AmazonEC2Config

#Sets the Client property for making calls -- queries across all regions (uses
default end-point)
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
E2_CreatelInstances.ps1**

```
$EC2Client=[Amazon.AWSClientFactory]::CreateAmazonEC2Client($AccountInfo.EC2AccessKey,$AccountInfo.EC2SecretKey,$config)

#Creates object for requesting list of regions -- value is null so all regions
will be returned
$DescribeRegionsRequest = New-Object Amazon.EC2.Model.DescribeRegionsRequest

#Queries for region information
$DescribeRegionsResponse = $EC2Client.DescribeRegions($DescribeRegionsRequest)

#Creates object for requesting list of regions -- value is null so all regions
will be returned
$DescribeImagesRequest = New-Object Amazon.EC2.Model.DescribeImagesRequest
$DescribeImagesRequest.Owner.add("amazon") #Queries against AMI's that were
created by Amazon

#Loops through all the regions and outputs AMI information to $ListOfAllAMIs
and includes the Region it was found in
Foreach ($Region in $DescribeRegionsResponse.DescribeRegionsResult.Region)
{
    Write-Output $Region.RegionName #Displays the Region name
    ##Sets the region
    $config.set_ServiceURL("https://"+$Region.Endpoint)
    $DescribeImagesResponse = $EC2Client.DescribeImages($DescribeImagesRequest)
    Write-Host "    Count " $DescribeImagesResponse.DescribeImagesResult.Image.Count
    #Displays the count of AMIs in each region

    #Loops through all the AMIs and copies information into $ListOfALLAMIs array
    Foreach ($Item in $DescribeImagesResponse.DescribeImagesResult.Image)
    {
        $object = New-Object psObject $Item
        Add-Member -InputObject $object -MemberType noteproperty -Name Region -Value
        $Region.RegionName
        $ListOfAllAMIs += $object
    }
}

##Exports to Excel
$File = "AMIList.csv"
$ListOfAllAMIs | Where{$_.Platform -eq "Windows"} | Export-CSV $File
Write-Host "Output to .\AMIList.csv"
```

E2_CreatelInstances.ps1

The following PowerShell script creates an Amazon EC2 instance from a specified AMI in your default region.

```
<#
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
E2_CreateInstances.ps1**

```
.DESCRIPTION
Creates an Instance from an AMI Description in user specified region where it
exists.
Useful to keep track of Amazon AMI ID's since they change every time updates
occur.

.NOTES
    PREREQUISITES:
    1) Download the SDK library from http://aws.amazon.com/sdkfornet/
    2) Have the AccessKey and SecretKey handy
    7) Know the name of the appropriate keyname for Region to $RunInstances
Request.KeyName

    API Reference: http://docs.amazonwebservices.com/AWSEC2/latest/APIRefer
ence/query-apis.html

.EXAMPLE
.\EC2_CreateInstance.ps1 -EC2AccessKey ThisIsMyAccessKey -EC2SecretKey This
IsMySecretKey -matchDescription 'Windows_Server-2008-R2-English-64Bit-2012' -
RegionName us-east-1 -InstanceType t1.micro -InstancesToLaunch 1 -SecurityGroup
Default -KeyPairName MyRegionalCertificateName -NameToTagCreatedInstance "This
is my tag"

.\EC2_CreateInstance.ps1 -EC2AccessKey ThisIsMyAccessKey -EC2SecretKey This
IsMySecretKey -matchDescription 'Windows_Server-2008-R2-English-64Bit-2012' -
RegionName us-east-1 -InstanceType t1.micro
#>

##Incoming Parameters
Param(
    [parameter(mandatory=$true)][string]$EC2AccessKey,
    [parameter(mandatory=$true)][string]$EC2SecretKey,
    [parameter(mandatory=$true)][string]$matchDescription,
    [parameter(mandatory=$true)][string]$RegionName,
    [parameter(mandatory=$true)][string]$InstanceType,
    [parameter(mandatory=$false)][string]$InstancesToLaunch = 1,
    [parameter(mandatory=$false)][string]$SecurityGroup = "Default",
    [parameter(mandatory=$true)][string]$KeyPairName,
    [parameter(mandatory=$false)][string]$NameToTagCreatedInstance = ""
)
##Creates and defines the objects for the Account
$AccountInfo = New-Object PSObject -Property @{
    EC2AccessKey = $EC2AccessKey
    EC2SecretKey = $EC2SecretKey
}

##Creates and defines the objects for the Instance Request
$Instance = New-Object PSObject -Property @{
    RegionName = $RegionName
    matchDescription = $matchDescription
    SecurityGroup = $SecurityGroup
    InstancesToLaunch = $InstancesToLaunch
    InstanceType = $InstanceType
    KeyPairName = $KeyPairName
    NameToTagCreatedInstance = $NameToTagCreatedInstance
}

##Defines Variables
```


**Amazon Elastic Compute Cloud Microsoft Windows
Guide
E2_CreateInstances.ps1**

```
$TagKey = "Name" #To add a custom tag, change this value
$ListOfQueriedAMIs = @() #Creates an array to store all the found AMI details
$ListOfCreatedInstances = @() #Creates an array to store all the created Instance
details

##Removes unused variables
Remove-Variable EC2AccessKey
Remove-Variable EC2SecretKey
Remove-Variable RegionName
Remove-Variable matchDescription
Remove-Variable SecurityGroup
Remove-Variable InstancesToLaunch
Remove-Variable InstanceType
Remove-Variable KeyPairName

##Loads the SDK information into memory
$SDKLibraryLocation = dir C:\Windows\Assembly -Recurse -Filter "AWSSDK.dll"
$SDKLibraryLocation = $SDKLibraryLocation.FullName
Add-Type -Path $SDKLibraryLocation

#Sets the end-point to the specified region
$config = New-Object Amazon.EC2.AmazonEC2Config
$config.set_ServiceURL("https://ec2."+$Instance.RegionName+".amazonaws.com")
#Sets the region

#Sets the Client property for making calls -- queries across all regions (uses
default end-point)
$EC2Client=[Amazon.AWSClientFactory]::CreateAmazonEC2Client($AccountInfo.EC2Ac
cessKey,$AccountInfo.EC2SecretKey,$config)

#####
##### FUNCTIONS #####

#Launches an Instance from an AMI
function createInstance
{ param([string]$amiid,[string]$instance
type,[string]$count,[string]$keypair,[string]$securitygroup)

#Required Information
$RunInstancesRequest = New-Object Amazon.EC2.Model.RunInstancesRequest
$RunInstancesRequest.ImageId = $amiid
$RunInstancesRequest.MaxCount = $count
$RunInstancesRequest.MinCount = $count
$RunInstancesRequest.SecurityGroup.Add($securitygroup)
if ($keypair -ne ""){$RunInstancesRequest.KeyName=$keypair} #If there is a
keypair value, uses it
$RunInstancesRequest.InstanceType = $instancetype

#Submits the request
$RunInstancesResponse = $EC2Client.RunInstances($RunInstancesRequest)

#Declares Array for return values
$runninginstances = @()

Foreach ($instance in $RunInstancesResponse.RunInstancesResult.Reservation.Run
ningInstance)
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
E2_CreateInstances.ps1**

```
{
    ##Adds each created instance to an Array to return
    $Object = New-Object psObject $instance.instanceID
    $runninginstances += $Object #Adds item to array with Region information
}
return $runninginstances
}

#Adds tag to the instance, which is visible via the AWS Console
function addTag
{ param([string]$instanceid,[string]$key,[string]$value)
  #Creates the tag objects
  $Tag = new-object amazon.EC2.Model.Tag
  $Tag.Key = $key #Tags the instance with a Name, which is visible in the AWS
  EC2 Console, or via API query
  $Tag.Value = $value #Records the AMI Name into the Tag Name value

  #Prepares the tag request
  $CreateTagRequest = new-object amazon.EC2.Model.CreateTagsRequest
  $CreateTagRequest.Tag.Add($Tag) #Bundles the Tag(s) into a single Tag Request

  $CreateTagRequest.ResourceId.Add($instanceid)

  #Submits the request
  $tagRequest = $EC2Client.CreateTags($CreateTagRequest) #Adds the Tag to the
  earlier created instance

  return $tagRequest
}

#####
#START OF SCRIPT
#####

Write-Output $Region.RegionName #Displays the Region name

#Creates filter to limit the return of objects
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "name"
$filter.Value.Add("*" + $Instance.matchDescription + "*") #Wildcard search for
Description

#Creates object for requesting list of AMIs
$DescribeImagesRequest = New-Object Amazon.EC2.Model.DescribeImagesRequest
$DescribeImagesRequest.Filter.Add($filter)
$DescribeImagesRequest.Owner.add("amazon") #Gets all AMI's that were created
by Amazon

#Submits the request to obtain list of AMIs
$DescribeImagesResult = $EC2Client.DescribeImages($DescribeImagesRequest)

#Checks to see if any AMIs were returned
If ($DescribeImagesResult.DescribeImagesResult.Image.Count -lt 1){return " No
results found for " + $Instance.matchDescription}

#Loops through all the AMIs and copies information into $ListOfALLAMIs array
#Foreach ($item in $DescribeImagesResult.DescribeImagesResult.Image | Where
{$_.Name -match $Instance.matchDescription -and $_.Visibility -like "Public"})
```

**Amazon Elastic Compute Cloud Microsoft Windows
Guide
E2_CreateInstances.ps1**

```
Foreach ($item in $DescribeImagesResult.DescribeImagesResult.Image)
{
    ##Adds each AMI found to an Array for later retrieval for launching instances

    $object = New-Object psObject $item
    Add-Member -InputObject $object -MemberType noteproperty -Name Region -Value
    $Instance.RegionName
    $ListOfQueriedAMIs += $object #Adds item to array with Region information
}

##Outputs details to console
Write-Host "Search Results for *"$Instance.matchDescription "*"
$ListOfQueriedAMIs | Format-Table -Property Region,name,ImageId,Architecture -
AutoSize #Displays the found AMIs in a table

#Calculates total number of instances that will be launched
$count = [int]$Instance.InstancesToLaunch * [int]$DescribeImagesResult.DescribeIm
agesResult.Image.Count

#Prompts user to create instance(s)
Write-Host "Total instance(s) to create " $count
$input = Read-Host "Enter 'y' to launch Instance(s) from the above AMI(s):"
if ($input -eq "y")
{
    Foreach ($AMI in $ListOfQueriedAMIs)
    {
        Write-Host "Creating Instance from AMI " $AMI.ImageId
        $ListOfCreatedInstances += $InstanceID = createInstance $AMI.ImageId $In
stance.InstanceType $Instance.InstancesToLaunch $Instance.KeyPairName $In
stance.SecurityGroup

        #Loops through all the instances that were created and adds the Name Tag in
formation; this is visible in the AWS EC2 Console
        Foreach ($createdInstance in $ListOfCreatedInstances)
        {
            #Runs if the optional tag value exists
            if ($Instance.NameToTagCreatedInstance -ne ""){$result = addTag $createdIn
stance $TagKey $Instance.NameToTagCreatedInstance}
        }
    }
}
##Exports to table for display
Write-Host
Write-Host "Instances Created"
$ListOfCreatedInstances.SyncRoot | Format-Table
}
```

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 S3-backed 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 (AMI) .
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 <i>Amazon DevPay Developer Guide</i> .
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.
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

The following table describes important additions to the *Amazon EC2 Microsoft Windows Guide*. We also update this guide to address the feedback that you send us.

Current API version: 2012-10-01

Change	Description	Release Date
Added content	The topic Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET (p. 87) explains how to configure Windows PowerShell to work with the .NET SDK and provides some code samples to help get you started.	March 2012
Added content	The topic Getting Started with Amazon EC2 Windows Instances (p. 8) helps you launch and connect to your first EC2 Windows instance. The topic Controlling Access: Security Groups and Credentials (p. 40) provides an overview of controlling access to your instances. The topic Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 20) shows how to create and deploy a WordPress blog on your Amazon EC2 instance.	December 2011
Added content	The topic Setting Up a Windows HPC Cluster on Amazon EC2 (p. 71) explains how to configure a Windows HPC Cluster on Amazon Elastic Compute Cloud.	November 2011
	This guide provides information about using Amazon EC2 Windows instances. For information about the basic infrastructure components of Windows instances, see What is Amazon EC2? (p. 3) . For information about using Windows AMIs, see Windows Amazon Machine Images (AMI) (p. 43) . For information about setting up your command line interface, see Installing the Amazon EC2 Command Line Tools on Windows (p. 81) .	September 2011

Index

A

AMIs

- paid, 67
- shared, 66
 - finding, 66
 - security, 67
- sharing, 62

B

batch processing, 3

G

glossary, 96

I

introduction, 3

O

overview, 3

P

Paid AMIs, 67

R

Regions, 85

S

scalable applications, 3

security, 40

service overview, 3

shared AMIs, 66

- finding, 66
- security, 67

sharing AMIs, 62

T

temporary events, 3

W

Windows user, 3