Amazon Elastic Compute Cloud

Microsoft Windows Guide

API Version 2012-04-01

Amazon Web Services

Amazon Elastic Compute Cloud: Microsoft Windows Guide

Amazon Web Services

Amazon Elastic Compute Cloud Microsoft Windows Guide

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizeable computing capacity literally server instances in Amazon's data centers that you use to build and host your software systems. 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. 75)

How to Use this Guide

The following table lists the complete contents of this guide.

I Want to..	Relevant Sections
Get a brief overview of Amazon EC2	Using Amazon EC2 (p. 3)
Get up and running right away with Amazon EC2	Getting Started (p. 8)
Control access to my Amazon EC2 instances	Controlling Access: Security Groups and Credentials (p. 20)
Learn the basic concepts for interacting with EC2	Amazon EC2 Infrastructure (p. 42)
Set up a WordPress blog on an Amazon EC2 instance	Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 23)

I Want to..	Relevant Sections
Get started with the command line tools	Installing the Amazon EC2 Command Line Tools on Windows (p. 37)
Get detailed information on how to use Windows AMIs	Using Windows AMIs (p. 46)
Use Windows PowerShell with Amazon EC2	Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET (p. 75)
Set up an HPC Cluster using Amazon EC2	Setting Up a Windows HPC Cluster on Amazon Elastic Compute Cloud (p. 84)

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

Using Amazon EC2

What Is Amazon EC2?

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 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 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 Using Windows AMIs (p. 46) and for information on using AMIs and Instances, see Using Amazon EC2.

Differences Between Windows Server and an Amazon EC2 Windows Instance

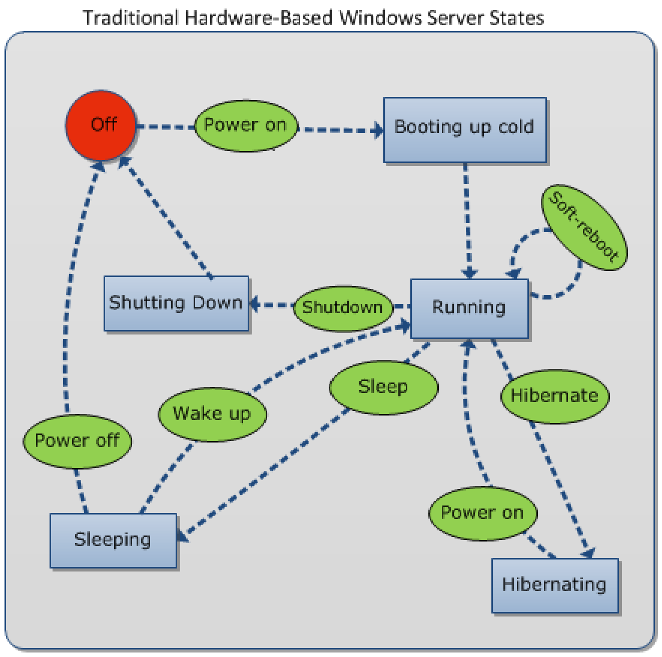
Amazon EC2 infrastructure is composed of virtual servers accessed via the Internet. These are commonly called *cloud servers*. By using Amazon EC2, you eliminate the need to buy and maintain expensive hardware. However, before you begin launching Amazon EC2 windows instances, you should be aware that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

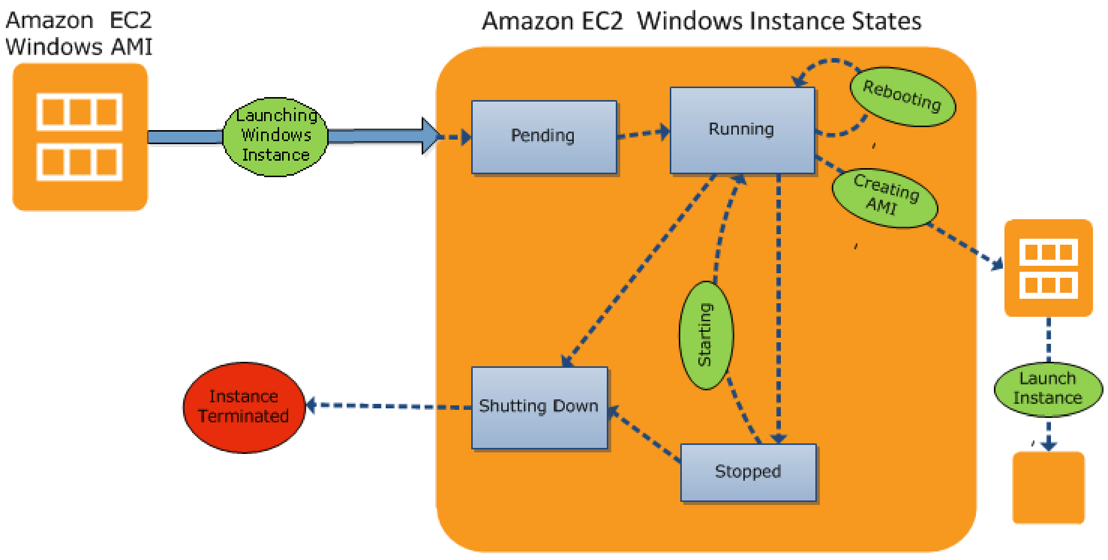
A traditional Windows Server goes through a number of phases from the time it is booted up through the time it is shut down, as the following diagram shows.



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, once 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.



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 will 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 will improve 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 once 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, ensure you 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

You can get started using Amazon Elastic Compute Cloud (Amazon EC2) Windows instances by stepping through the tasks 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.

Getting Started with Amazon EC2

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

Step 1: Sign Up for EC2

If you already have an AWS account, skip to the next procedure. If you don't already have an AWS account, use the following procedure to create one.

Note

When you create an account, AWS automatically signs up the account for all AWS services. You are charged only for the services you use.

To create an AWS account

1. Go to http://aws.amazon.com, and click **Sign Up**.
2. Follow the on-screen instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Step 2: Launch an Amazon EC2 Windows Instance

Now that you're signed up for Amazon EC2, you're ready to launch an instance using the AWS Management Console.

You can either leverage the Free Usage Tier to launch and use a free Amazon EC2 Windows Micro Instance for 12 months, or you can launch a regular Windows instance (not within the Free Usage Tier). For more information about the Free Usage Tier, go to the AWS Free Usage Tier product page and Getting Started with AWS Free Usage Tier.

If you want to launch a regular Windows instance (not within the Free Usage Tier), you will incur the standard Amazon EC2 usage fees for the instance until this tutorial shows you how to terminate it in the last task . The total charges will be minimal (typically less than a few dollars). For more information about Amazon EC2 usage rates, go to the Amazon EC2 product page.

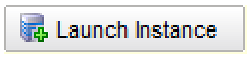
Important

If you launch an instance that is not within the Free Usage Tier, the usage fees are minimal, and you are billed once you launch the instance and charged for the time that the instance is running even if it remains idle.

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 you used when signing up for Amazon EC2.
2. From the Amazon EC2 console dashboard, click **Launch Instance**.

   The **Create a New Instance** page provides two ways to launch an instance:

   - The **Classic Wizard** offers you more granular control and advanced settings for configuring the type of instance you want to launch.
   - The **Quick Launch Wizard** simplifies the process for you and automatically configures many selections for you so you can started quickly with an instance. This tutorial guides you through the Quick Launch Wizard.

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

   Important

   Do not select the **None** option. If you launch an instance without a key pair, you will not be able to connect to your instance.

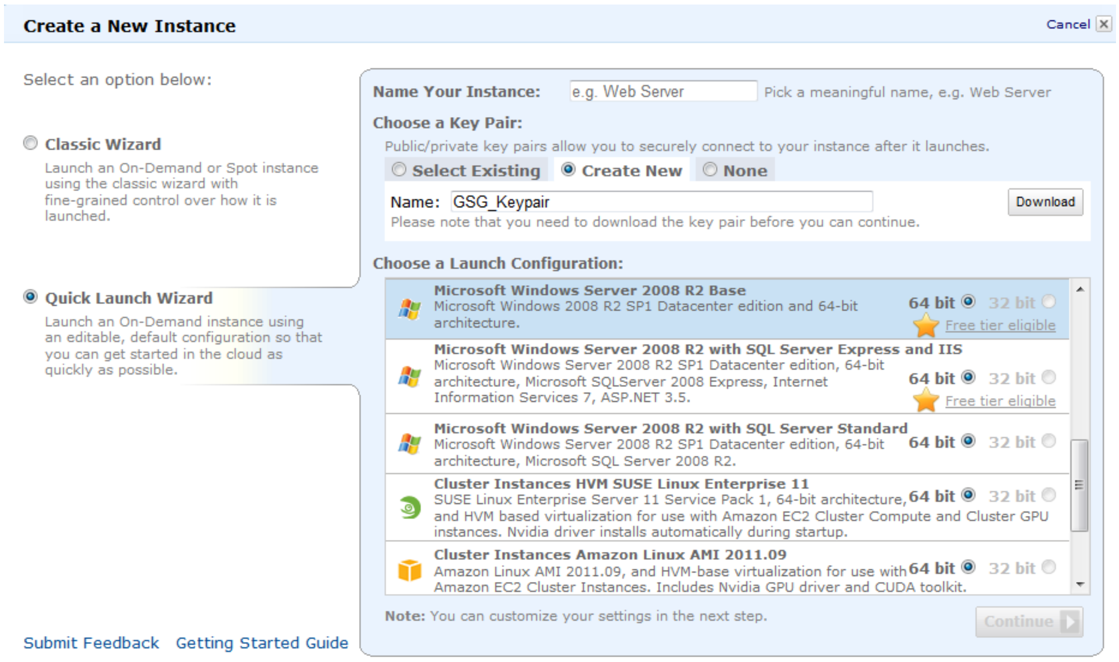
   a. Click **Create new Key Pair**.

b. Type a name for your key pair and then click **Download**. You will need the contents of the private key to connect to your instance once it is launched. Amazon Web Services does not keep the private portion of key pairs.

c. Save the private key in a safe place on your system. Note the location because you'll need the key to connect to the instance.

6. Under **Choose a Launch Configuration**, choose the operating system and software configuration for your instance. In this example, we'll use Microsoft Windows Server 2008 with a 64-bit operating system. The star by this choice indicates that it is within the Free Usage Tier.

   The Quick Launch Wizard displays a list of basic configurations called Amazon Machine Images (AMIs) that you can choose from to launch an instance. An Amazon Machine Image (AMI) contains all the information needed to create a new instance of a server. For example, an AMI might contain all the software to act as a web server (e.g., Windows and IIS), or all the software to act as a Windows database server (e.g., Windows and SQL Server). To keep things simple, AWS marks the AMIs that are available in the Free Usage Tier with a star.



Create a New Instance

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.

Name Your Instance: e.g. Web Server — 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: GSG_Keypair

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

Choose a Launch Configuration:

Microsoft Windows Server 2008 R2 Base Microsoft Windows 2008 R2 SP1 Datacenter edition and 64-bit architecture.	[✓ 64 bit] 32 bit	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] 32 bit	Free tier eligible
Microsoft Windows Server 2008 R2 with SQL Server Standard Microsoft Windows Server 2008 R2 SP1 Datacenter edition, 64-bit architecture, Microsoft SQL Server 2008 R2.	[✓ 64 bit] 32 bit	
Cluster Instances HVM SUSE Linux Enterprise 11 SUSE Linux Enterprise Server 11 Service Pack 1, 64-bit architecture, and HVM based virtualization for use with Amazon EC2 Cluster Compute and Cluster GPU instances. Nvidia driver installs automatically during startup.	[✓ 64 bit] 32 bit	
Cluster Instances Amazon Linux AMI 2011.09 Amazon Linux AMI 2011.09, and HVM-base virtualization for use with Amazon EC2 Cluster Instances. Includes Nvidia GPU driver and CUDA toolkit.	[✓ 64 bit] 32 bit	

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

7. Click **Continue** to view the settings that your instance will launch with.

8. Under **Security Details**, in **Security Group**, the wizard automatically makes a security group selection for you.

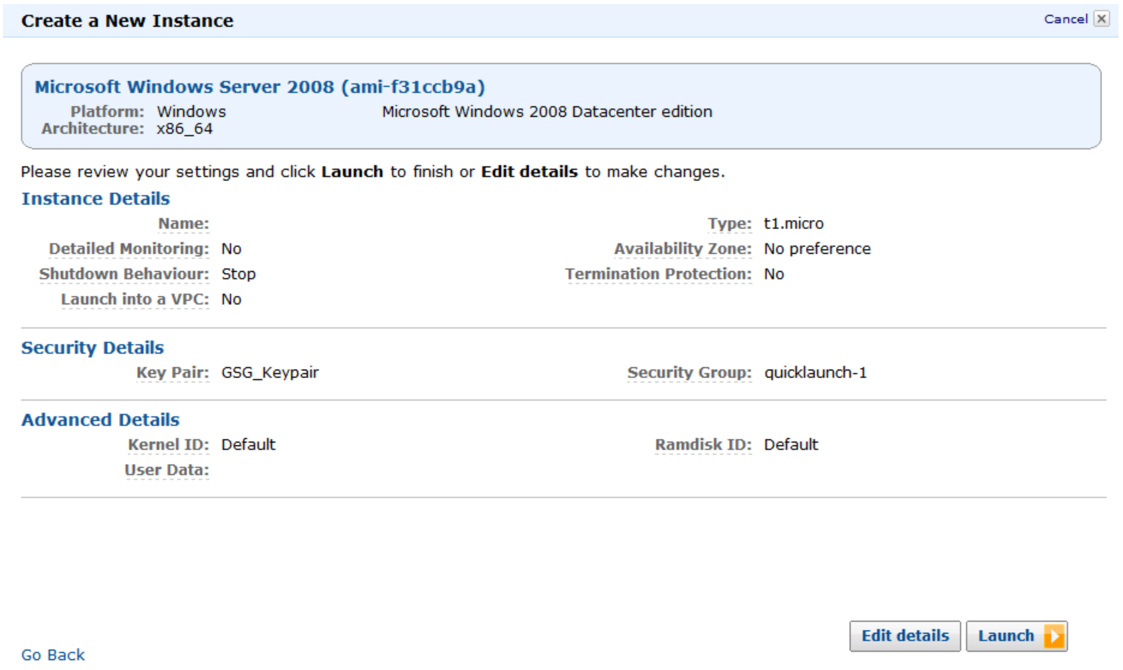
   A security group defines firewall rules for your instances. These rules specify which incoming network traffic will be delivered to your instance. All other traffic is ignored.

   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 quick-launch-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 enables all IP addresses to access your instance over the specified ports (e.g., RPD for Windows). 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 addresses to access your instance.



Microsoft Windows Server 2008 (ami-f31ccb9a)
Platform: Windows — Microsoft Windows 2008 Datacenter edition
Architecture: x86_64

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

Instance Details

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

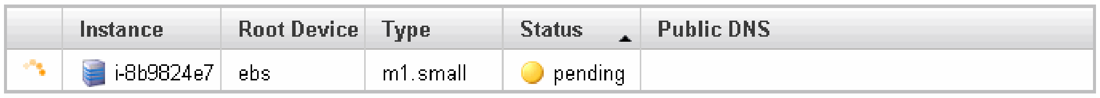
Security Details

Key Pair:	GSG_Keypair	Security Group:	quicklaunch-1

Advanced Details

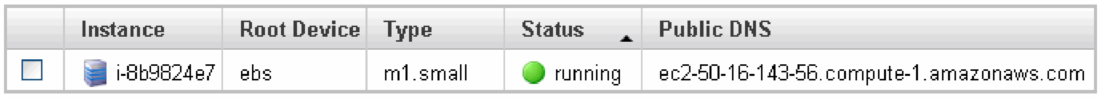
Kernel ID:	Default	Ramdisk ID:	Default
User Data:			

9. Review your settings, and click **Launch** to launch the instance.
10. A confirmation page lets you know your instance is launching. Click **Close** to close the confirmation page.
11. 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 will be *pending* while it's launching.



	Instance	Root Device	Type	Status	Public DNS
	i-8b9824e7	ebs	m1.small	● pending	

After a short period, the status of your instance switches to *running*. You can click **Refresh** to refresh the display.

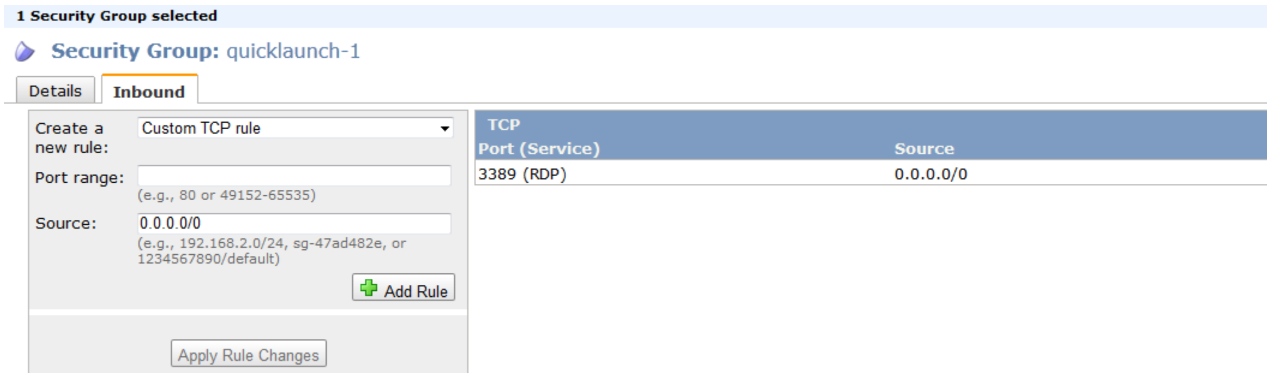


	Instance	Root Device	Type	Status	Public DNS
☐	i-8b9824e7	ebs	m1.small	● running	ec2-50-16-143-56.compute-1.amazonaws.com

12. Record the **Public DNS** name for your instance because you'll need it for the next task. If you select the instance, its details (including the public DNS name) are displayed in the lower pane. You can also click **Show/Hide** in the top right corner of the page to select which columns to display.

13. (Optional) After your instance is launched, you can view the quicklaunch-*x* security group rule that was created.

    a. On the Amazon EC2 console, under **Network and Security**, click **Security Groups**.
    b. Click the quicklaunch-*x* security group and you can view the security rules created.



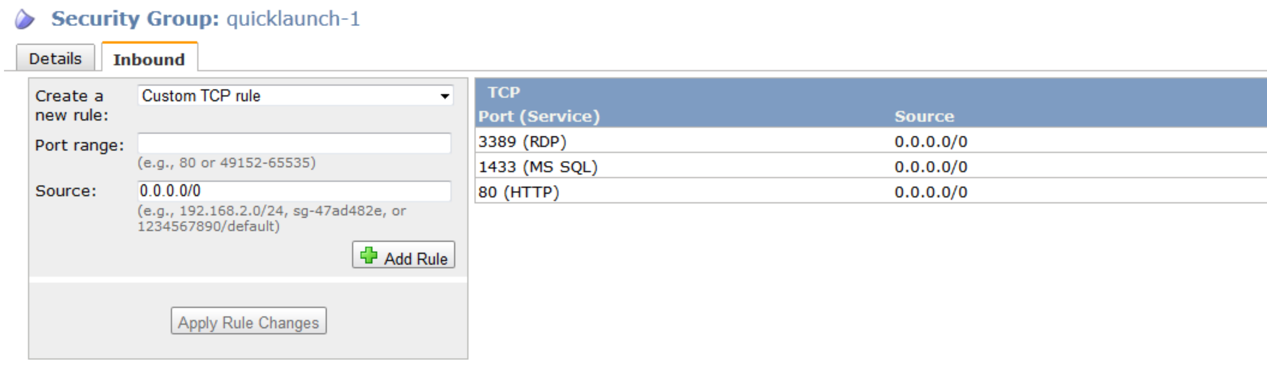
1 Security Group selected

Security Group: quicklaunch-1

Create a new rule: Custom TCP rule
Port range: (e.g., 80 or 49152-65535)
Source: 0.0.0.0/0 (e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

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

As you can see, the security group contains one rule that allows RDP traffic from any IP source to port 3389. If you had launched a Windows instance running IIS and SQL, the Quick Launch wizard would create a security group that would also allow traffic to port 80 for HTTP (for IIS) and port 1433 for MS SQL, as shown in the following figure.



Security Group: quicklaunch-1

Create a new rule: Custom TCP rule
Port range: (e.g., 80 or 49152-65535)
Source: 0.0.0.0/0 (e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

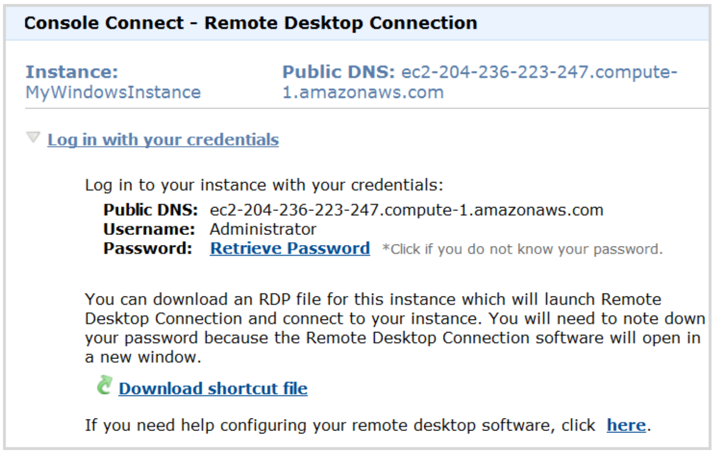
TCP Port (Service)	Source
3389 (RDP)	0.0.0.0/0
1433 (MS SQL)	0.0.0.0/0
80 (HTTP)	0.0.0.0/0

Step 3: Connect to Your Windows Instance

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

To connect to your Windows instance

1. In the AWS Management Console, right-click the instance you created, and then click **Connect**.
2. Click **Retrieve Password**.



Console Connect - Remote Desktop Connection

Instance: MyWindowsInstance
Public DNS: ec2-204-236-223-247.compute-1.amazonaws.com

Log in with your credentials

Log in to your instance with your credentials:
Public DNS: ec2-204-236-223-247.compute-1.amazonaws.com
Username: Administrator
Password: Retrieve Password *Click if you do not know your password.

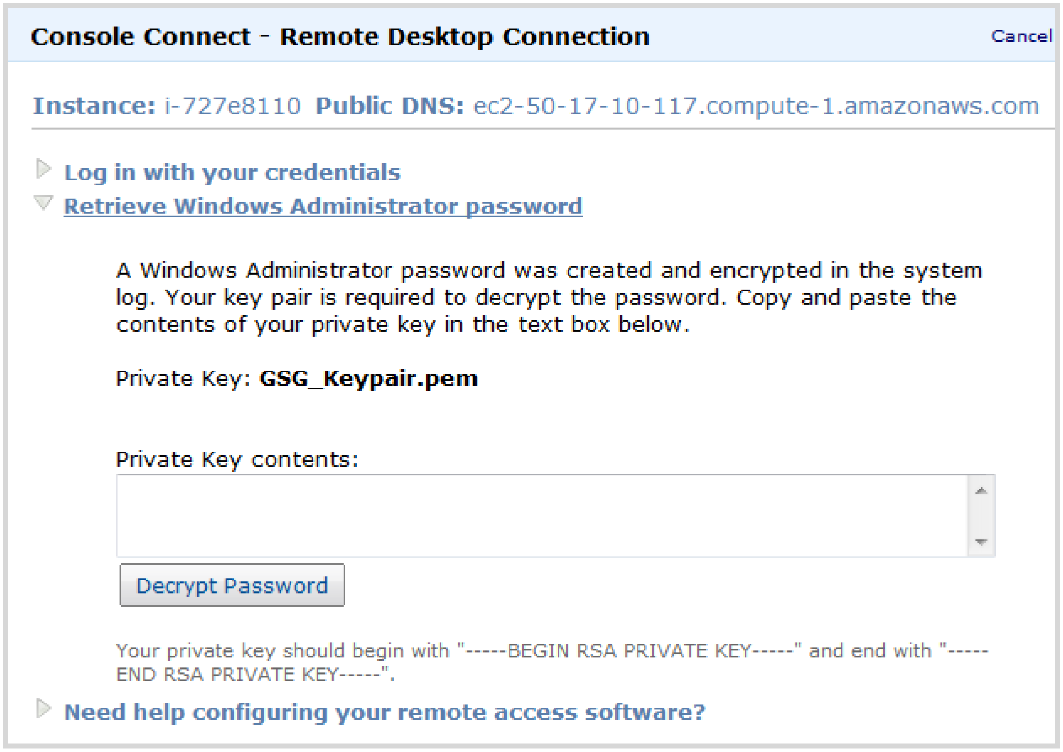
You can download an RDP file for this instance which will launch Remote Desktop Connection and connect to your instance. You will need to note down your password because the Remote Desktop Connection software will open in a new window.

Download shortcut file

If you need help configuring your remote desktop software, click here.

3. Navigate to the directory where you stored the private key file when you launched the instance.
4. Open the file in a text editor and copy the entire contents (including the first and last lines, which contain `BEGIN RSA PRIVATE KEY` and `END RSA PRIVATE KEY`).
5. Paste the contents of the private key file into the **Private Key contents** field, and then click **Decrypt Password**.

   The console returns the default administrator password for the instance.



Console Connect - Remote Desktop Connection

Cancel

Instance: i-727e8110 **Public DNS:** ec2-50-17-10-117.compute-1.amazonaws.com

Log in with your credentials

Retrieve Windows Administrator password

A Windows Administrator password was created and encrypted in the system log. Your key pair is required to decrypt the password. Copy and paste the contents of your private key in the text box below.

Private Key: **GSG_Keypair.pem**

Private Key contents:

Decrypt Password

Your private key should begin with "-----BEGIN RSA PRIVATE KEY-----" and end with "-----END RSA PRIVATE KEY-----".

Need help configuring your remote access software?

6. Save the default administrator password. You will need it to connect to the instance.

Tip

You can also copy and paste the password. Copy it to the clipboard for the next step.

7. Click **Download shortcut file**, and then click **Open**.

   Note

   Most modern Windows operating systems from Windows XP onward already include the Remote Desktop application. If you're using an old version of Windows, you can download the Remote Desktop application from the Microsoft web site and install it on your client computer.

8. Log in using `Administrator` as the username and the administrator password you got in the previous task as the password.

   Important

   After you connect to any new Windows instance you've just launched, we recommend you change the Windows 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.

9. Create another user account with administator privileges on the instance. Another administrator account is a safeguard if you forget your administrator password or have a problem with the administator account.

You can now 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 machine 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.

Step 4: Create an Elastic IP Address

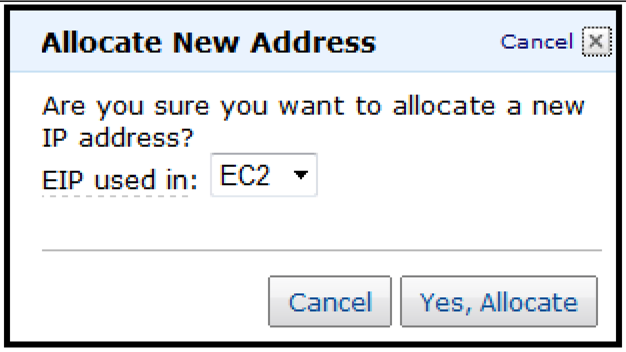
By default, all Amazon EC2 instances are assigned two IP addresses at launch: a private (RFC 1918) address and a public address that is mapped to the private IP address through network address translation (NAT).

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, you can 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**.

Step 5: Create a CloudWatch Alarm to Monitor Your Instance

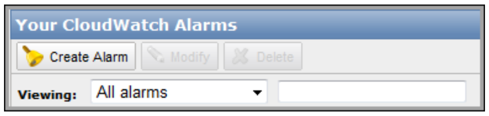
With Amazon CloudWatch, you can monitor various aspects of your instance and set up alarms based on criteria you choose. For example, you could configure an alarm to send you an email when an instance's CPU exceeds 70 percent.

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.

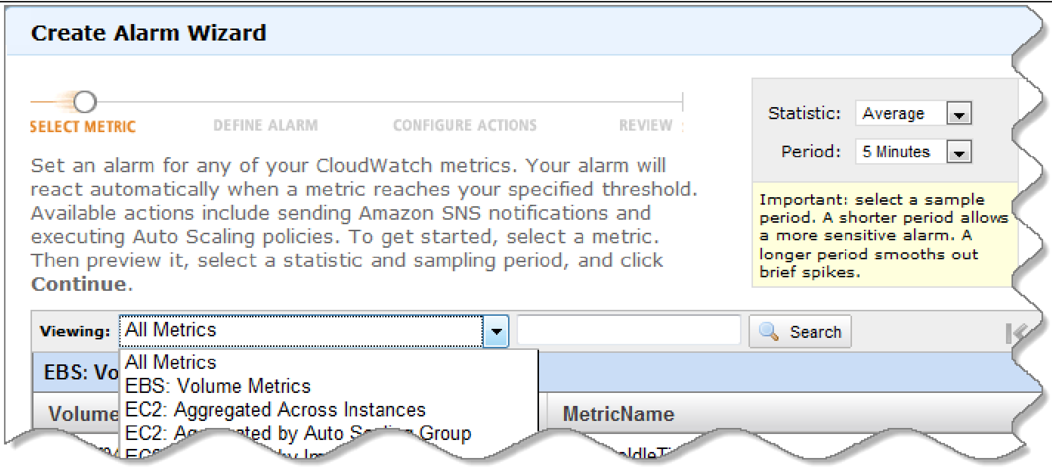
The **Create Alarm Wizard** steps you through the process of creating an alarm.

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.



Create Alarm Wizard

SELECT METRIC · DEFINE ALARM · CONFIGURE ACTIONS · REVIEW

Set an alarm for any of your CloudWatch metrics. Your alarm will react automatically when a metric reaches your specified threshold. Available actions include sending Amazon SNS notifications and executing Auto Scaling policies. To get started, select a metric. Then preview it, select a statistic and sampling period, and click **Continue**.

Statistic: Average
Period: 5 Minutes

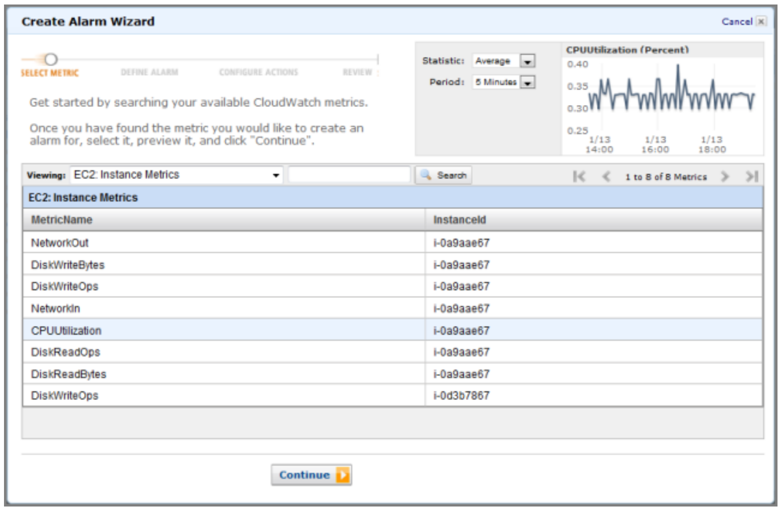
Important: select a sample period. A shorter period allows a more sensitive alarm. A longer period smooths out brief spikes.

Viewing: All Metrics

Search
All Metrics
EBS: Volume Metrics
EC2: Aggregated Across Instances
EC2: Aggregated by Auto Scaling Group
MetricName

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**.



Create Alarm Wizard

SELECT METRIC · DEFINE ALARM · CONFIGURE ACTIONS · REVIEW

Get started by searching your available CloudWatch metrics.

Once you have found the metric you would like to create an alarm for, select it, preview it, and click "Continue".

Statistic: Average
Period: 5 Minutes

Viewing: EC2: Instance Metrics

1 to 8 of 8 Metrics

EC2: Instance Metrics

MetricName	InstanceId
NetworkOut	i-0a9aae67
DiskWriteBytes	i-0a9aae67
DiskWriteOps	i-0a9aae67
NetworkIn	i-0a9aae67
CPUUtilization	i-0a9aae67
DiskReadOps	i-0a9aae67
DiskReadBytes	i-0a9aae67
DiskWriteOps	i-0d3b7867

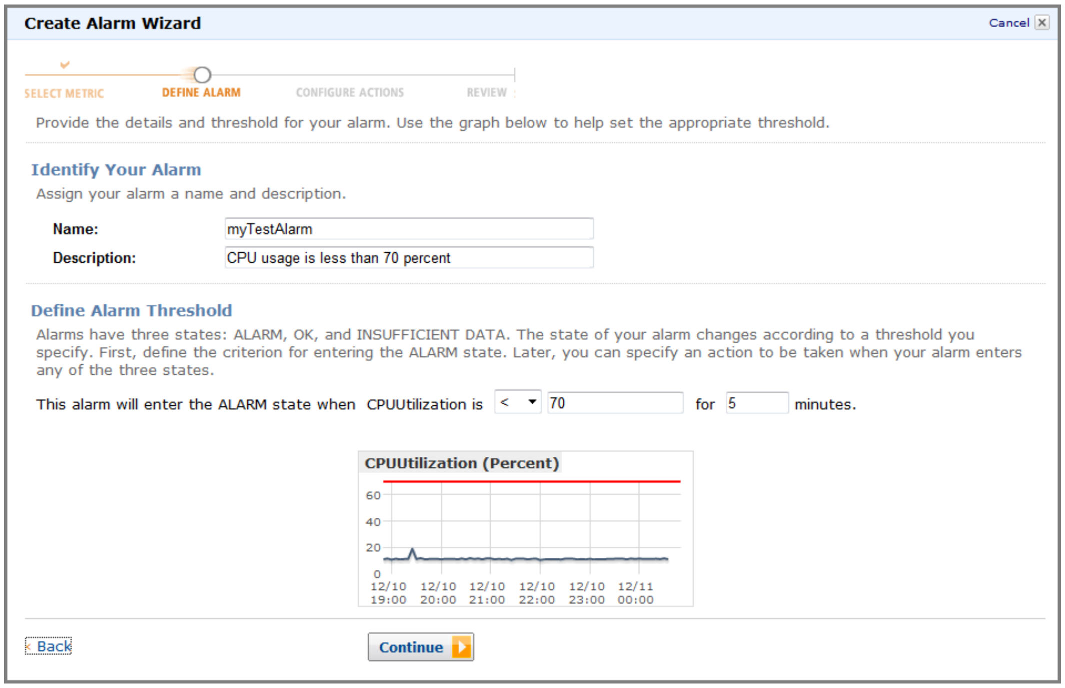
6. The **DEFINE ALARM** page of the **Create Alarm Wizard** opens.

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**.



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.

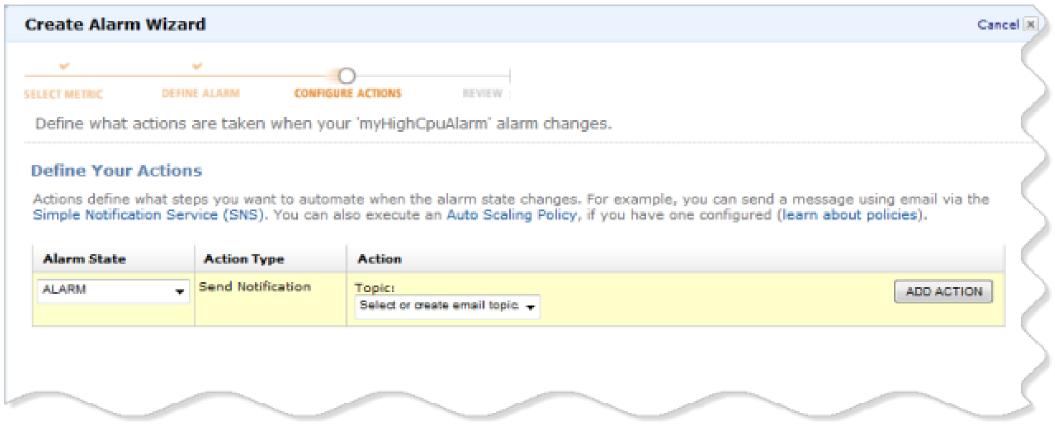
Name:	myTestAlarm
Description:	CPU usage is less than 70 percent

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 < 70 for 5 minutes.

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



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).

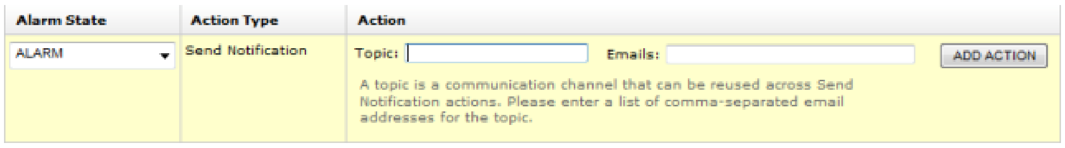
Alarm State	Action Type	Action
ALARM	Send Notification	Topic: Select or create email topic

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: Emails: ADD ACTION 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.

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.

Step 6: Clean Up (Optional)

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 not within the Free Usage Tier as soon as the instance status changes to `shutting down` or `terminated`, you stop incurring charges for that instance. If you launched an instance in the Free Usage Tier, no charges are incurred.

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

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) notification. 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 specific 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

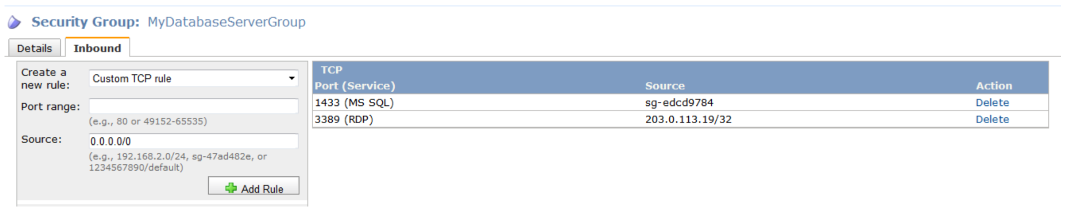
You can also define a security group as the source for a security group rule. For example, let's assume your application will use two Amazon EC2 instances:

- 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 notification for this address. Because the suffix 32 uses the entire IPv4 address, it allows in only a single host.



Create a new rule:	Custom TCP rule
Port range:	(e.g., 80 or 49152-65535)
Source:	0.0.0.0/0 (e.g., 192.168.2.0/24, sg-47ad482e, or 1234567890/default)

TCP

Port (Service)	Source	Action
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 *SSH key pairs*) 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 snaphot	AWS Account ID of the account to share with (without the hyphens)

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. 23)
Task 2: Installing WordPress (p. 27)
Task 3: Configuring Your WordPress Site (p. 35)

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 (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 will not be accessible from outside the instance.
- Connected to your Amazon EC2 instance.

Task 1: Installing the Microsoft Web Deployment Tool

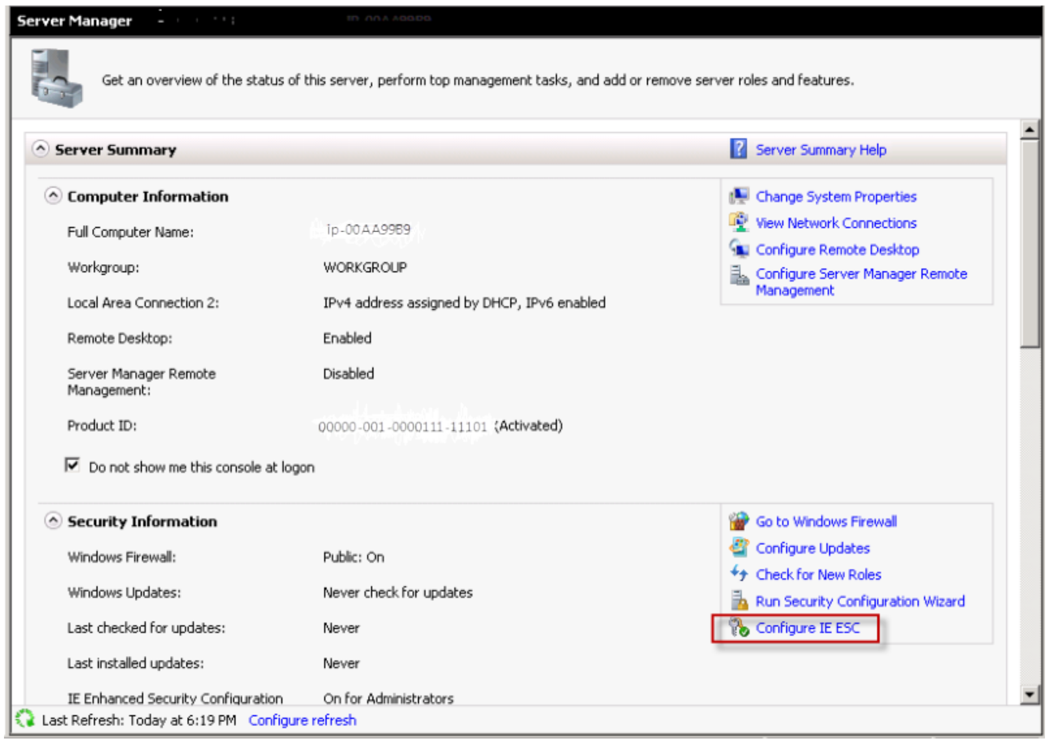
1. Verify you've met the conditions in Prerequisites (p. 23).
2. Configure Server Manager Settings.

a. In your Amazon EC2 instance, click **Start**.

b. Right-click **Administrative Tools**.

c. Select **Server Manager**.

d. In the **Server Manager** box, in the **Security Information** panel, click **Configure IE ESC**.



Server Manager

Get an overview of the status of this server, perform top management tasks, and add or remove server roles and features.

Server Summary — Server Summary Help

Computer Information

Full Computer Name:	Ip-00AA99B9
Workgroup:	WORKGROUP
Local Area Connection 2:	IPv4 address assigned by DHCP, IPv6 enabled
Remote Desktop:	Enabled
Server Manager Remote Management:	Disabled
Product ID:	00000-001-0000111-11101 (Activated)

☑ Do not show me this console at logon

Security Information

Windows Firewall:	Public: On
Windows Updates:	Never check for updates
Last checked for updates:	Never
Last installed updates:	Never
IE Enhanced Security Configuration	On for Administrators

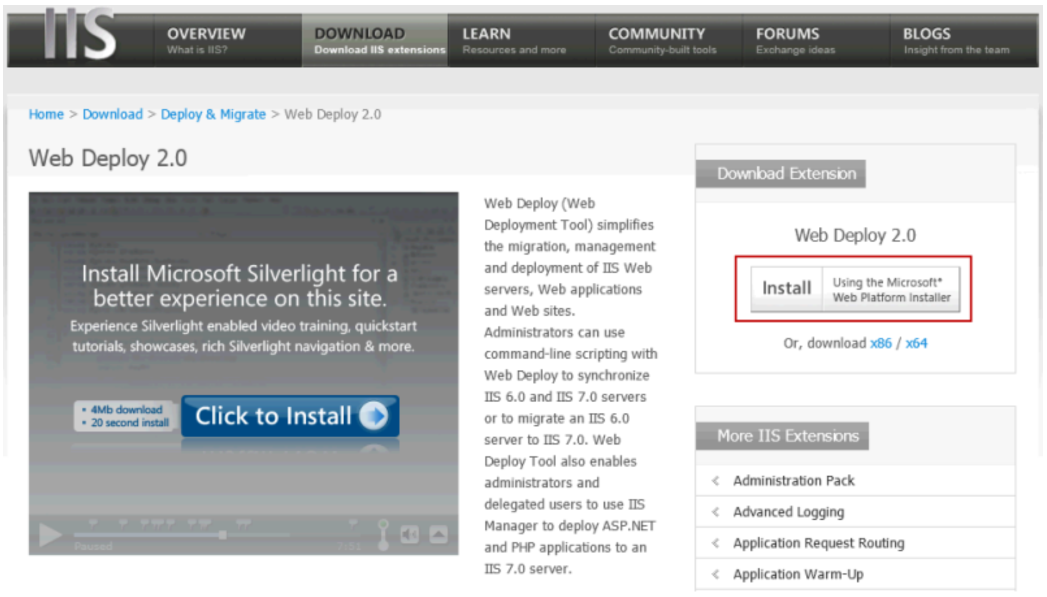
Last Refresh: Today at 6:19 PM Configure refresh

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.

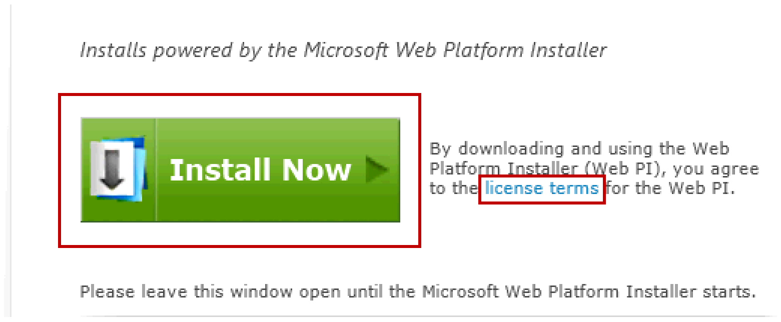

IIS
OVERVIEW What is IIS?
DOWNLOAD Download IIS extensions
LEARN Resources and more
COMMUNITY Community-built tools
FORUMS Exchange ideas
BLOGS Insight from the team
Home > Download > Deploy & Migrate > Web Deploy 2.0
Install Microsoft Silverlight for a better experience on this site.
Experience Silverlight enabled video training, quickstart tutorials, showcases, rich Silverlight navigation & more.
• 4Mb download • 20 second install
Click to Install
Download Extension
Install
Using the Microsoft® Web Platform Installer
Or, download x86 / x64
More IIS Extensions
Administration Pack
Advanced Logging
Application Request Routing
Application Warm-Up

Web Deploy 2.0

Web Deploy (Web Deployment Tool) simplifies the migration, management and deployment of IIS Web servers, Web applications and Web sites. Administrators can use command-line scripting with Web Deploy to synchronize IIS 6.0 and IIS 7.0 servers or to migrate an IIS 6.0 server to IIS 7.0. Web Deploy Tool also enables administrators and delegated users to use IIS Manager to deploy ASP.NET and PHP applications to an IIS 7.0 server.

Web Deploy 2.0

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

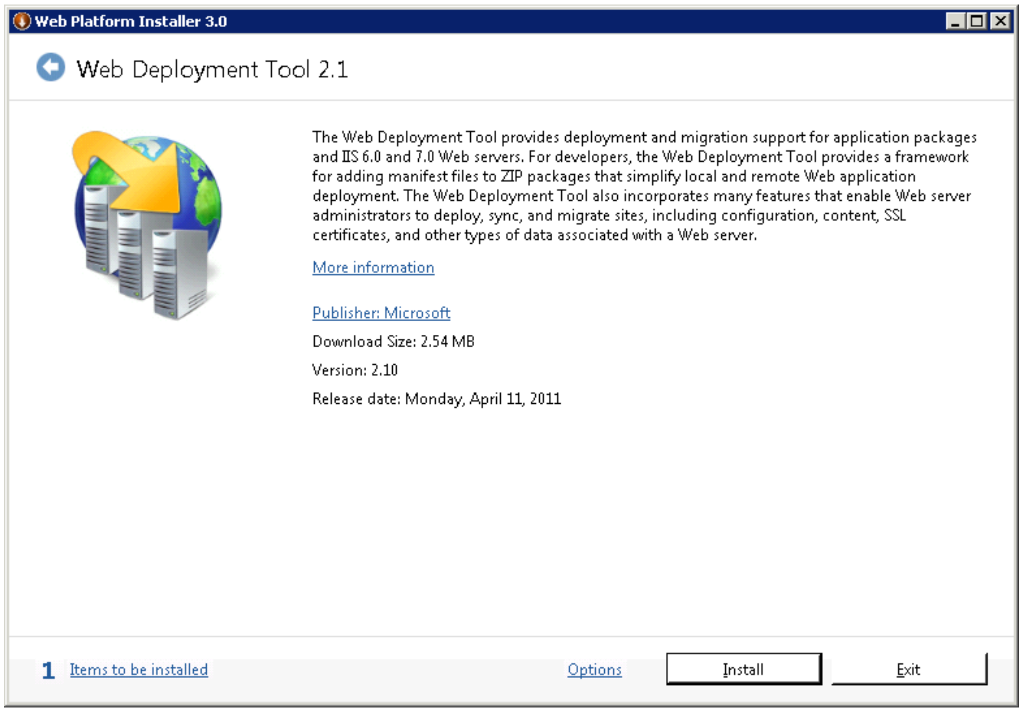


Installs powered by the Microsoft Web Platform Installer

By downloading and using the Web Platform Installer (Web PI), you agree to the license terms for the Web PI.

Please leave this window open until the Microsoft Web Platform Installer starts.

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.



Web Platform Installer 3.0

Web Deployment Tool 2.1

The Web Deployment Tool provides deployment and migration support for application packages and IIS 6.0 and 7.0 Web servers. For developers, the Web Deployment Tool provides a framework for adding manifest files to ZIP packages that simplify local and remote Web application deployment. The Web Deployment Tool also incorporates many features that enable Web server administrators to deploy, sync, and migrate sites, including configuration, content, SSL certificates, and other types of data associated with a Web server.

More information

Publisher: Microsoft

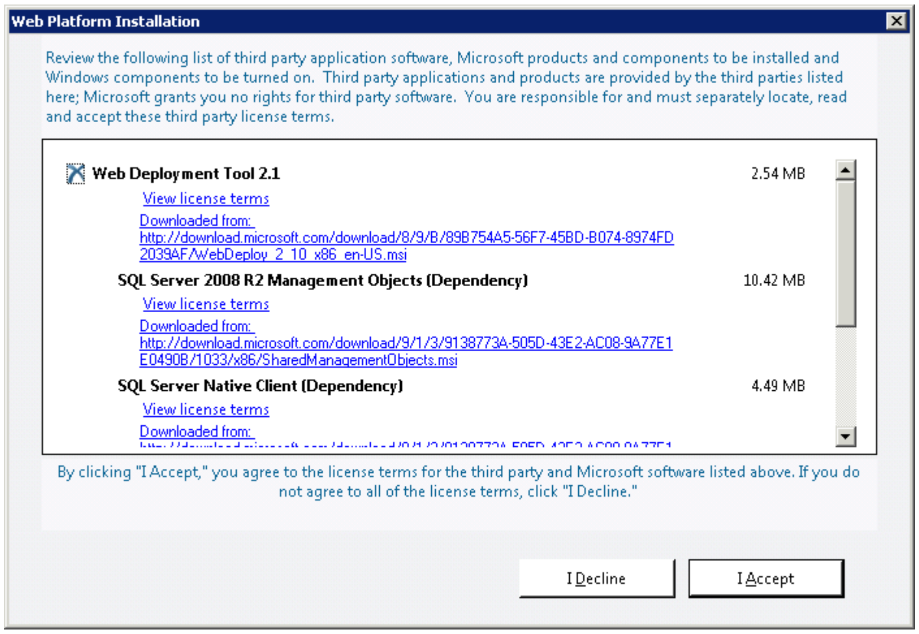
Download Size: 2.54 MB

Version: 2.10

Release date: Monday, April 11, 2011

1 Items to be installed   Options   Install   Exit

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



Web Platform Installation

Review the following list of third party application software, Microsoft products and components to be installed and Windows components to be turned on. Third party applications and products are provided by the third parties listed here; Microsoft grants you no rights for third party software. You are responsible for and must separately locate, read and accept these third party license terms.

Web Deployment Tool 2.1 — 2.54 MB
View license terms
Downloaded from: http://download.microsoft.com/download/8/9/B/89B754A5-56F7-45BD-B074-8974FD2039AF/WebDeploy_2_10_x86_en-US.msi

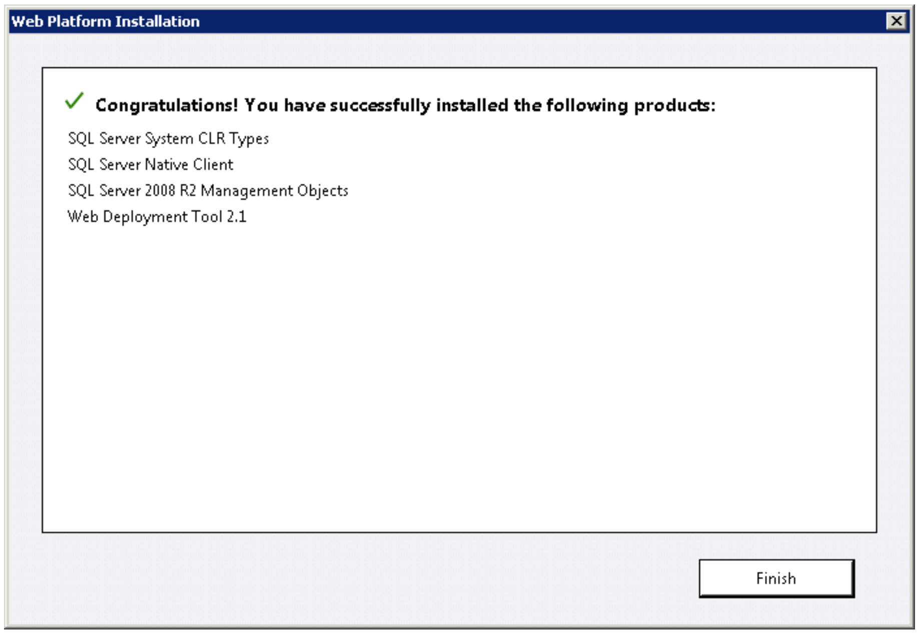
SQL Server 2008 R2 Management Objects (Dependency) — 10.42 MB
View license terms
Downloaded from: http://download.microsoft.com/download/9/1/3/9138773A-505D-43E2-AC08-9A77E1E0490B/1033/x86/SharedManagementObjects.msi

SQL Server Native Client (Dependency) — 4.49 MB
View license terms
Downloaded from:

By clicking "I Accept," you agree to the license terms for the third party and Microsoft software listed above. If you do not agree to all of the license terms, click "I Decline."

I Decline   I Accept

9. Click **I Accept**. The Web Platform Installer begins to install the software. The wizard will indicate when it has finished installing the software.



Congratulations! You have successfully installed the following products:

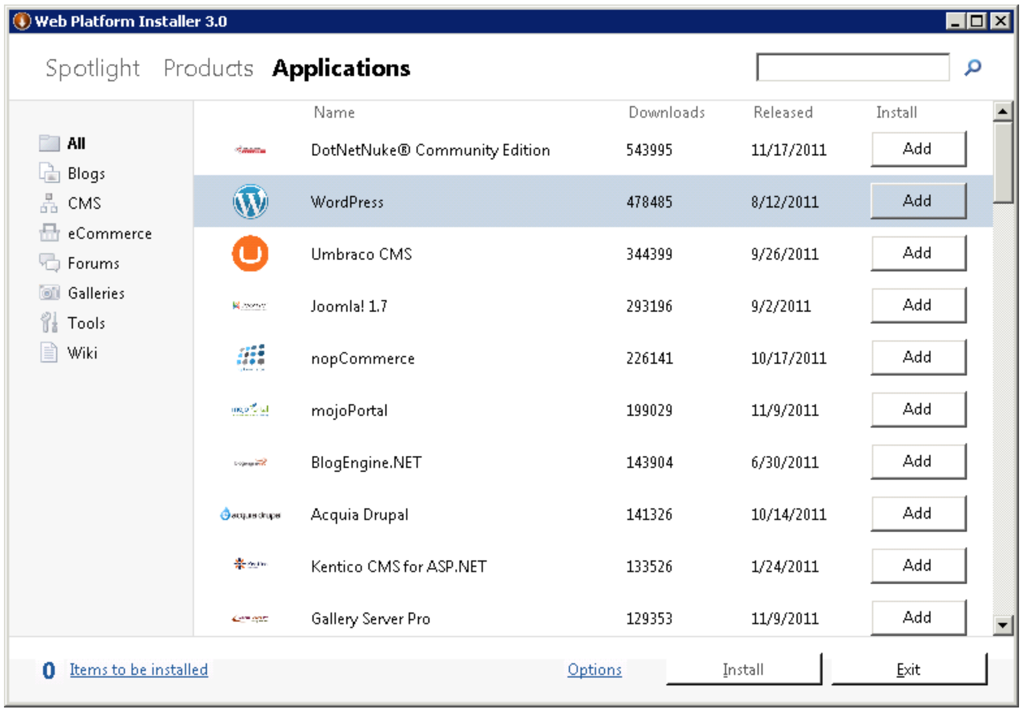
SQL Server System CLR Types
SQL Server Native Client
SQL Server 2008 R2 Management Objects
Web Deployment Tool 2.1

10. Click **Finish**. Next the **Web Platform Installer** launches and you'll need to install WordPress.

Task 2: Installing WordPress

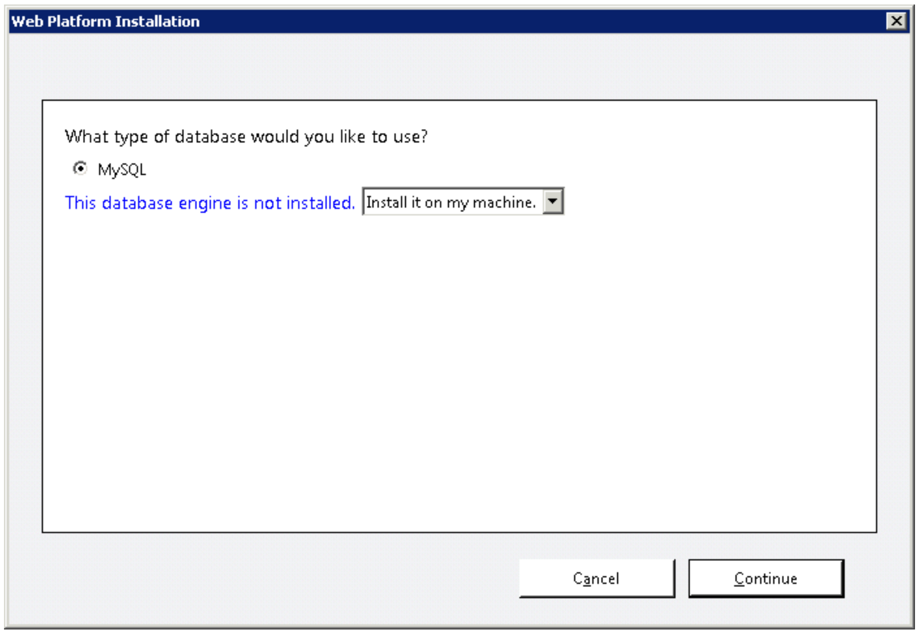
To Install WordPress

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



Name	Downloads	Released	Install
DotNetNuke® Community Edition	543995	11/17/2011	Add
WordPress	478485	8/12/2011	Add
Umbraco CMS	344399	9/26/2011	Add
Joomla! 1.7	293196	9/2/2011	Add
nopCommerce	226141	10/17/2011	Add
mojoPortal	199029	11/9/2011	Add
BlogEngine.NET	143904	6/30/2011	Add
Acquia Drupal	141326	10/14/2011	Add
Kentico CMS for ASP.NET	133526	1/24/2011	Add
Gallery Server Pro	129353	11/9/2011	Add

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

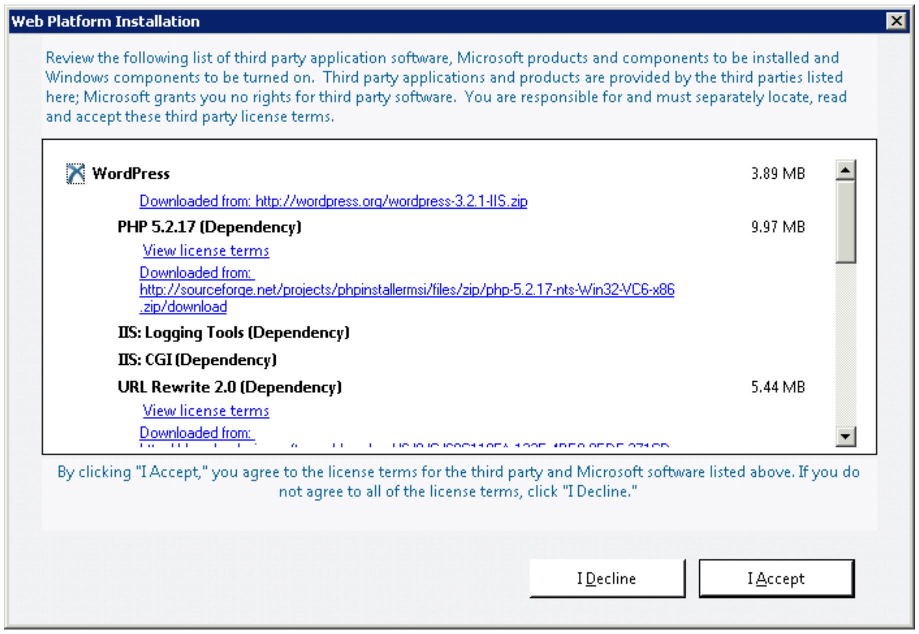


What type of database would you like to use?

[✓ MySQL]

This database engine is not installed. Install it on my machine.

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

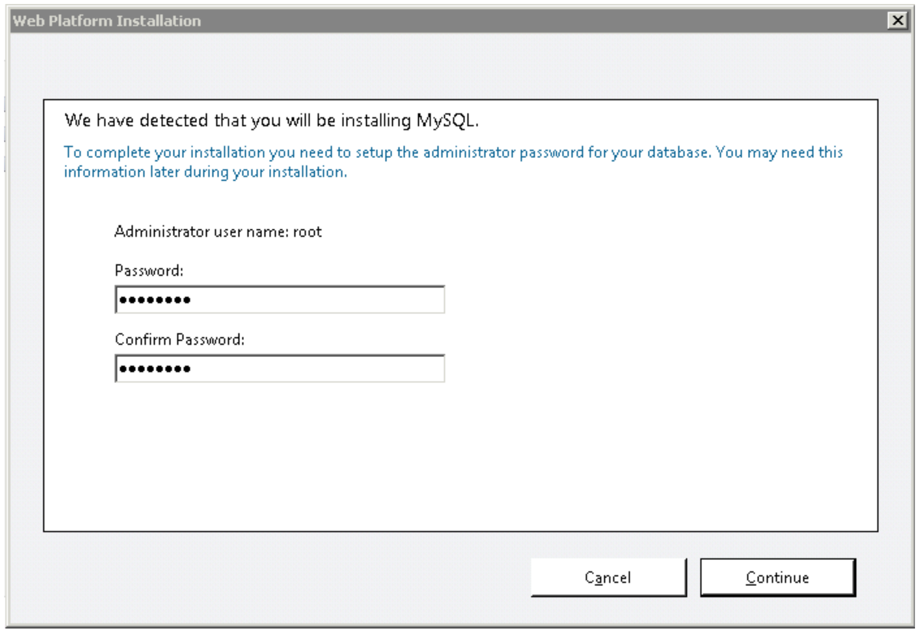

Web Platform Installation

Review the following list of third party application software, Microsoft products and components to be installed and Windows components to be turned on. Third party applications and products are provided by the third parties listed here; Microsoft grants you no rights for third party software. You are responsible for and must separately locate, read and accept these third party license terms.

WordPress 3.89 MB
Downloaded from: http://wordpress.org/wordpress-3.2.1-IIS.zip
PHP 5.2.17 (Dependency) 9.97 MB
View license terms
Downloaded from: http://sourceforge.net/projects/phpinstallermsi/files/zip/php-5.2.17-nts-Win32-VC6-x86.zip/download
IIS: Logging Tools (Dependency)
IIS: CGI (Dependency)
URL Rewrite 2.0 (Dependency) 5.44 MB
View license terms
Downloaded from:

By clicking "I Accept," you agree to the license terms for the third party and Microsoft software listed above. If you do not agree to all of the license terms, click "I Decline."

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


Web Platform Installation

We have detected that you will be installing MySQL.

To complete your installation you need to setup the administrator password for your database. You may need this information later during your installation.

Administrator user name: root

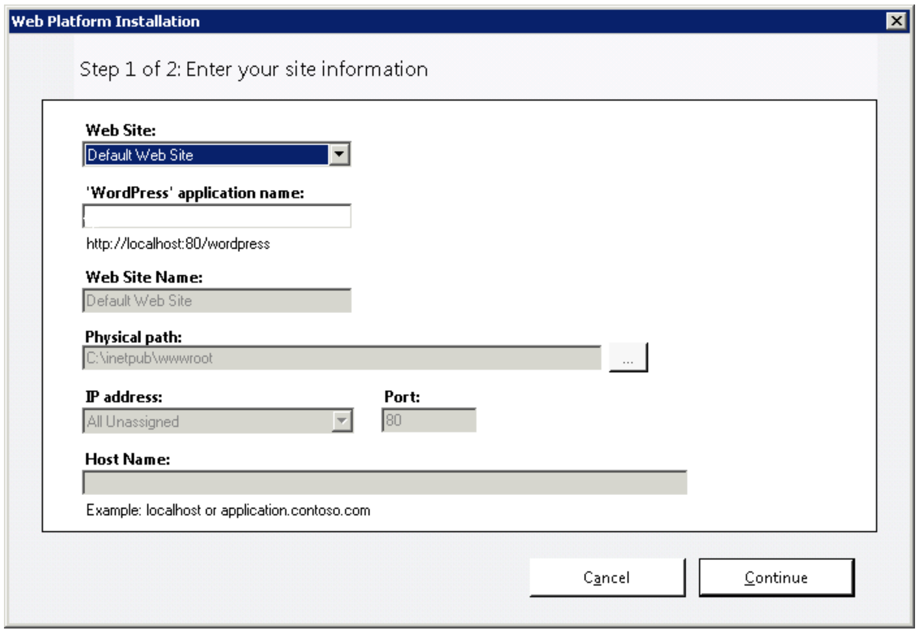
Password: ••••••••

Confirm Password: ••••••••

7. Type a password for your MySQL database in the **Password** box and the **Confirm Password** box.

   After the Web Platform Installer finishes installing the software, you will be 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.



Web Platform Installation

Step 1 of 2: Enter your site information

Web Site: Default Web Site

'WordPress' application name:

http://localhost:80/wordpress

Web Site Name: Default Web Site

Physical path: C:\inetpub\wwwroot

IP address: All Unassigned

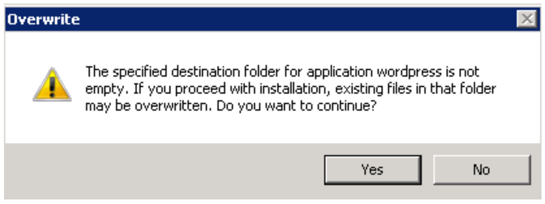
Port: 80

Host Name:

Example: localhost or application.contoso.com

b. Leave the default information in other fields and click **Continue**.

c. In the **Overwrite** box, click **Yes**.



Overwrite

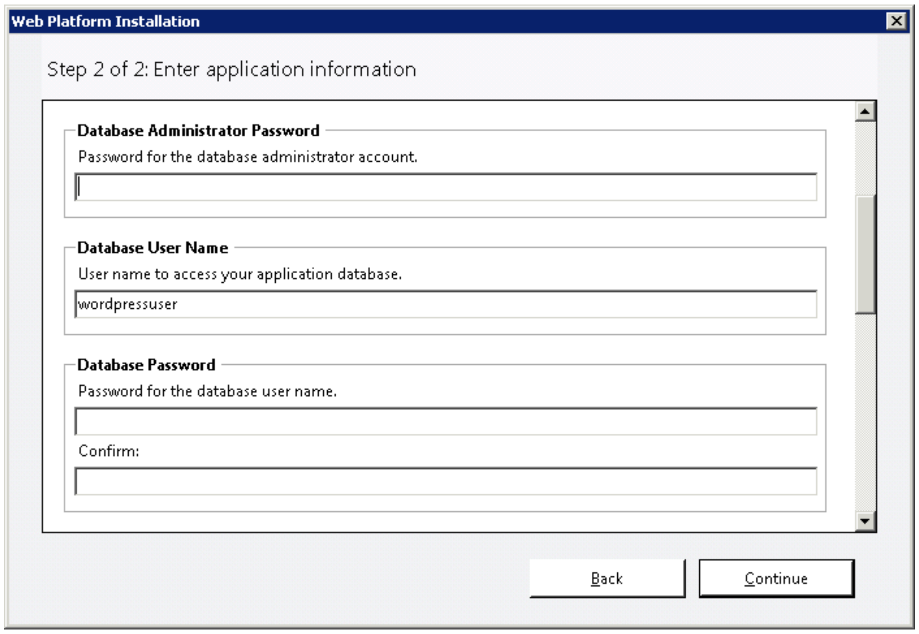
The specified destination folder for application wordpress is not empty. If you proceed with installation, existing files in that folder may be overwritten. Do you want to continue?

9. For **Step 2** of **Enter application information**, enter the following information.

   a. In the **Database Administrator Password** box, type a password.

   b. Leave **Database User Name** set to the default value.

   c. For **Database Password**, type a password in the **Password for the database user name** box and the **Confirm** box.



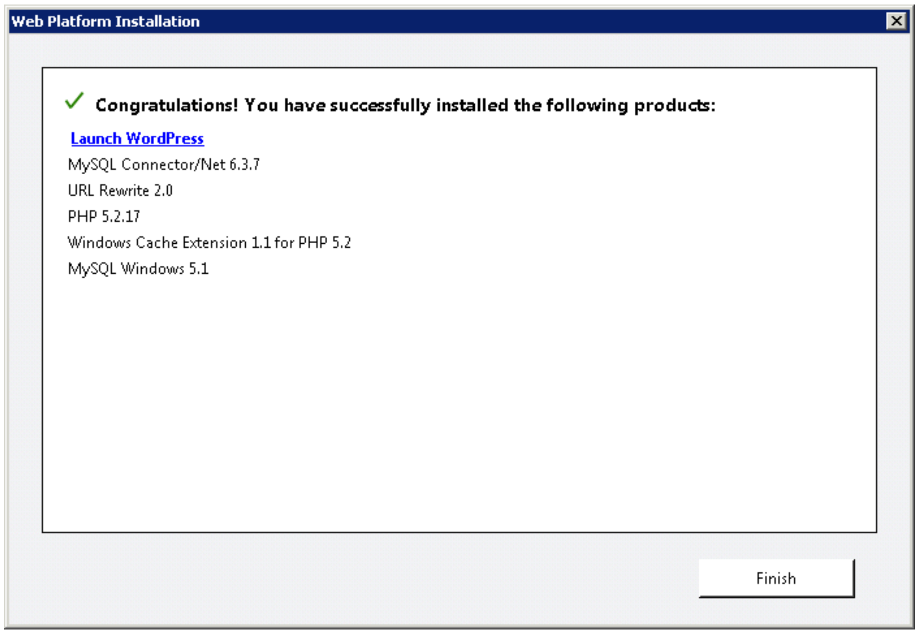
Step 2 of 2: Enter application information

Database Administrator Password
Password for the database administrator account.

Database User Name
User name to access your application database.
wordpressuser

Database Password
Password for the database user name.
Confirm:

d. Click **Continue**. The Web Platform Installation completion page appears.

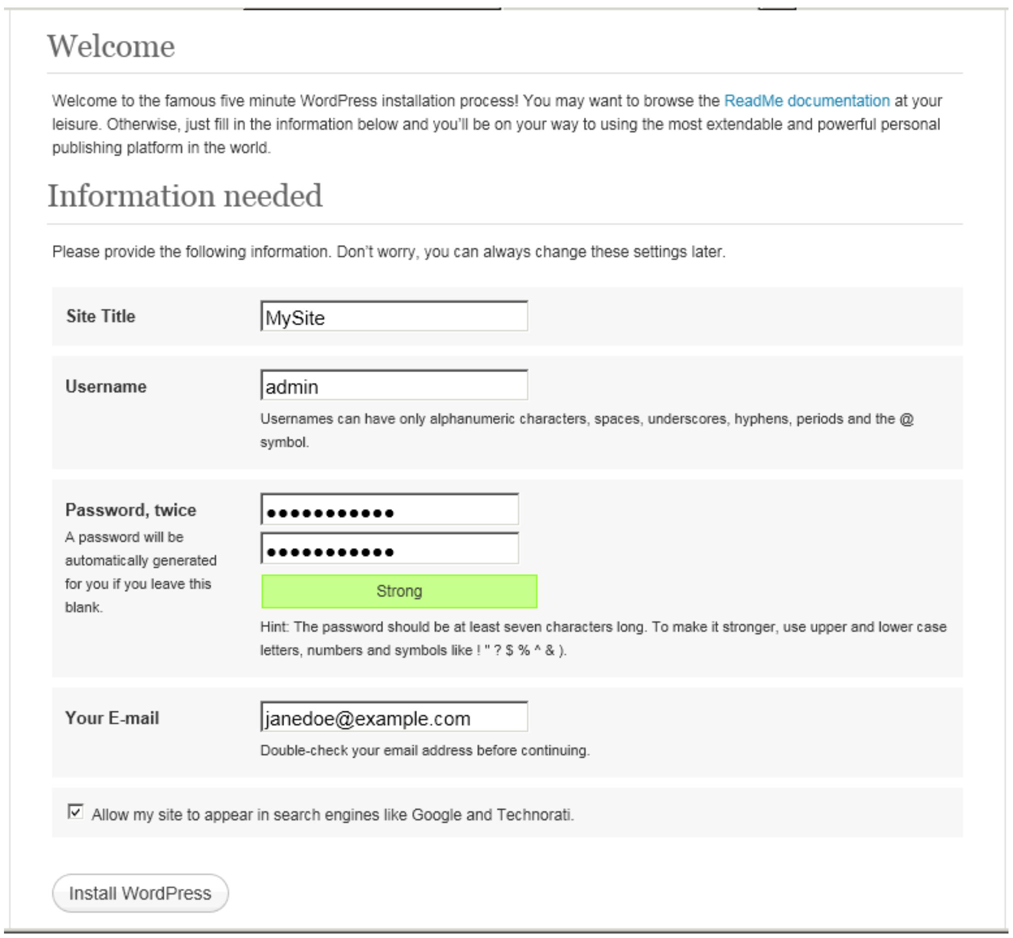


Congratulations! You have successfully installed the following products:

Launch WordPress
MySQL Connector/Net 6.3.7
URL Rewrite 2.0
PHP 5.2.17
Windows Cache Extension 1.1 for PHP 5.2
MySQL Windows 5.1

10. Click **Launch WordPress**. The WordPress **Welcome** page appears.
11. On the **Welcome** page, enter the following information.

    a. In the **Site Title** box, type your site title.
    b. In the **Username** box, leave the default set to admin.
    c. In the **Password, twice** boxes, type your password twice.
    d. In the **Your E-mail** box, 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

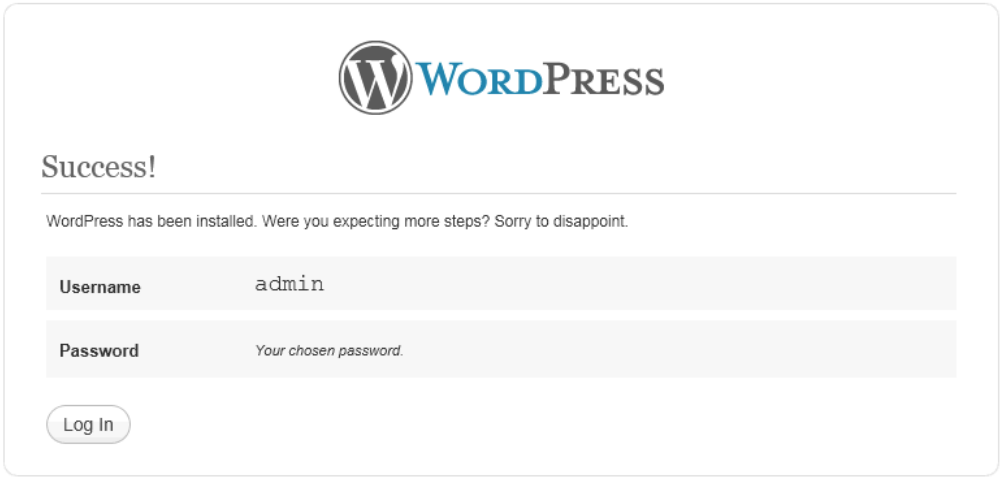
Please provide the following information. Don't worry, you can always change these settings later.

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

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

Install WordPress

e. Click **Install WordPress**. The WordPress **Success** page appears.



Success!

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

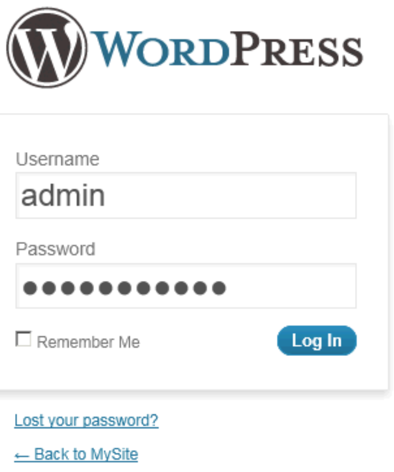
Username	admin
Password	*Your chosen password.*

Log In

12. Click **Log In**. The **Log In** page appears.
13. On the **Log In** page, enter the following information.

a. In the **Username** box, type admin.

b. In the **Password** box, type your password.



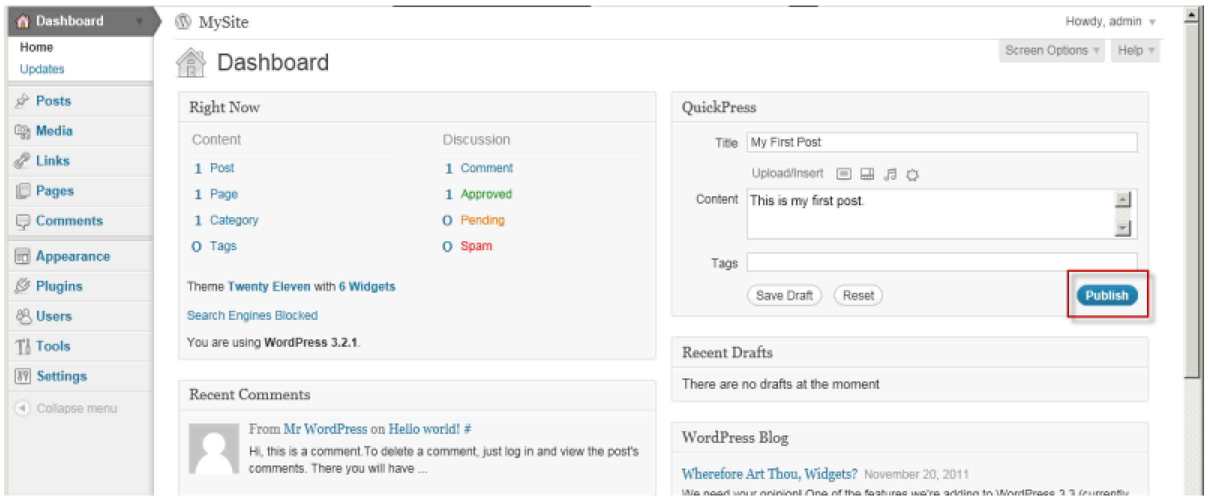
Username
admin
Password
●●●●●●●●●●●

c. Click **Log In**. The WordPress **Dashboard** appears.

14. In the **QuickPress** box, enter the following information.

a. In the **Title** box, type the title you want.

b. In the **Content** box, type some content.



Dashboard

Right Now

Content	Discussion
1 Post	1 Comment
1 Page	1 Approved
1 Category	0 Pending
0 Tags	0 Spam

Theme Twenty Eleven with 6 Widgets
Search Engines Blocked
You are using WordPress 3.2.1.

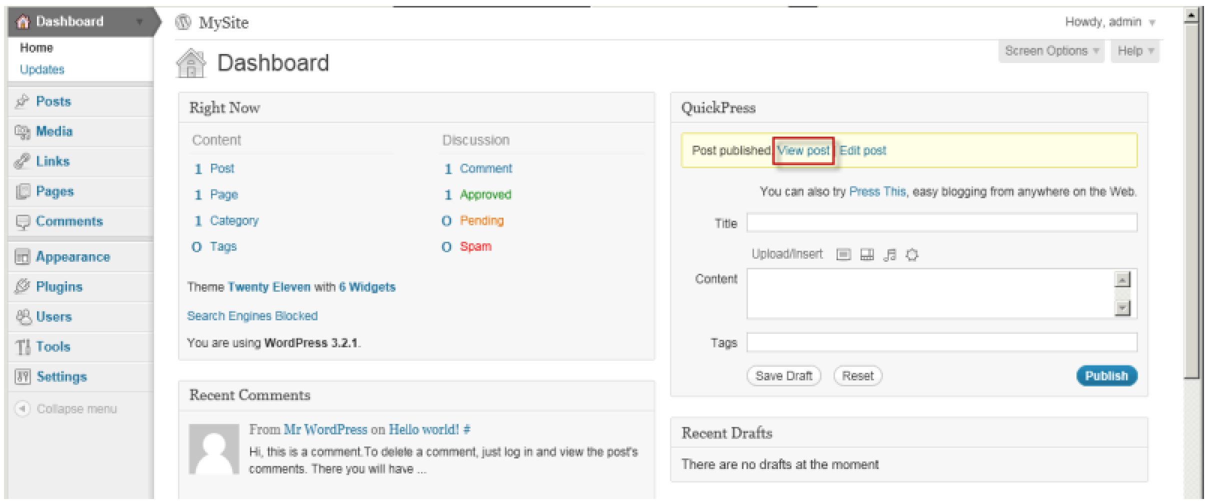
Recent Comments
From Mr WordPress on Hello world! #
Hi, this is a comment. To delete a comment, just log in and view the post's comments. There you will have ...

QuickPress
Title: My First Post
Content: This is my first post.
Tags:

Recent Drafts
There are no drafts at the moment

WordPress Blog
Wherefore Art Thou, Widgets? November 20, 2011
We need your opinion! One of the features we're adding to WordPress 3.3 (currently

c. Click **Publish**. A notification appears in which you can choose to view or edit the post.



Theme Twenty Eleven with 6 Widgets

Search Engines Blocked

You are using WordPress 3.2.1.

Post published View post Edit post

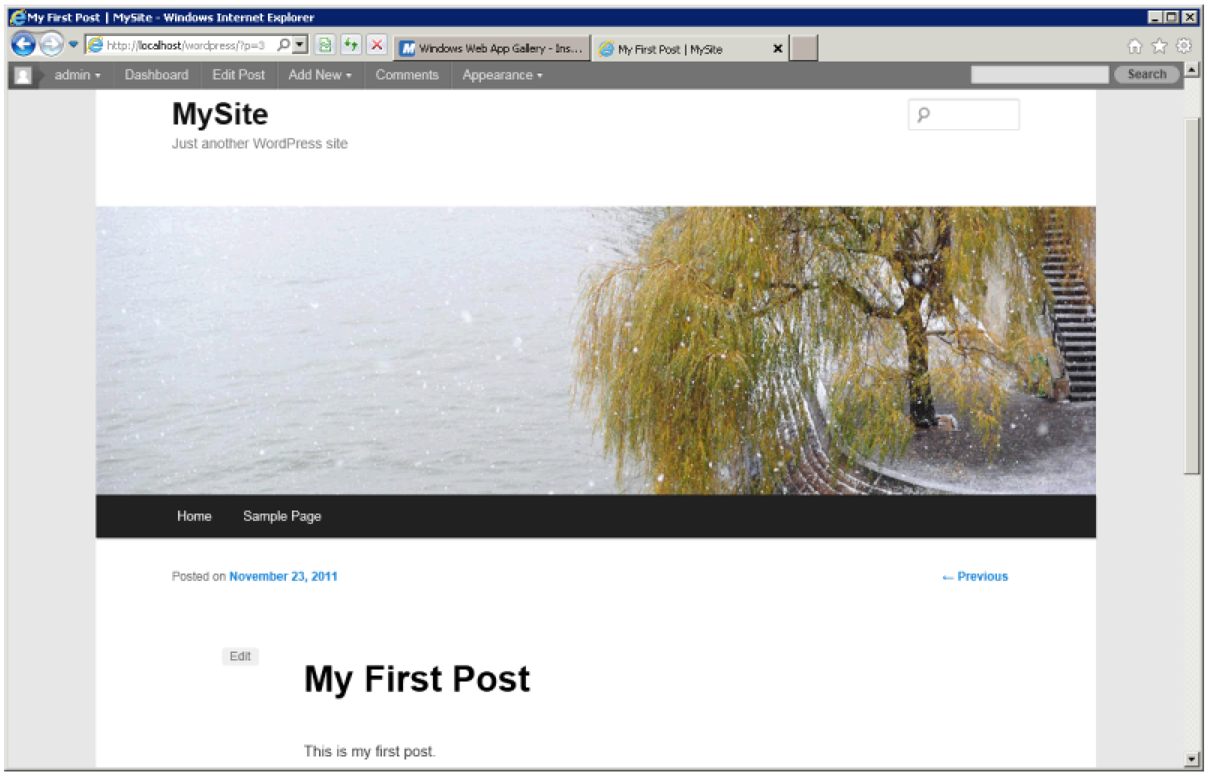
You can also try Press This, easy blogging from anywhere on the Web.

From Mr WordPress on Hello world! #

Hi, this is a comment.To delete a comment, just log in and view the post's comments. There you will have ...

There are no drafts at the moment

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


My First Post | MySite - Windows Internet Explorer
http://localhost/wordpress/?p=3
Windows Web App Gallery - Ins...
My First Post | MySite
admin
Dashboard
Edit Post
Add New
Comments
Appearance
Search
Home
Sample Page
← Previous
Edit

MySite

Just another WordPress site

Posted on November 23, 2011

My First Post

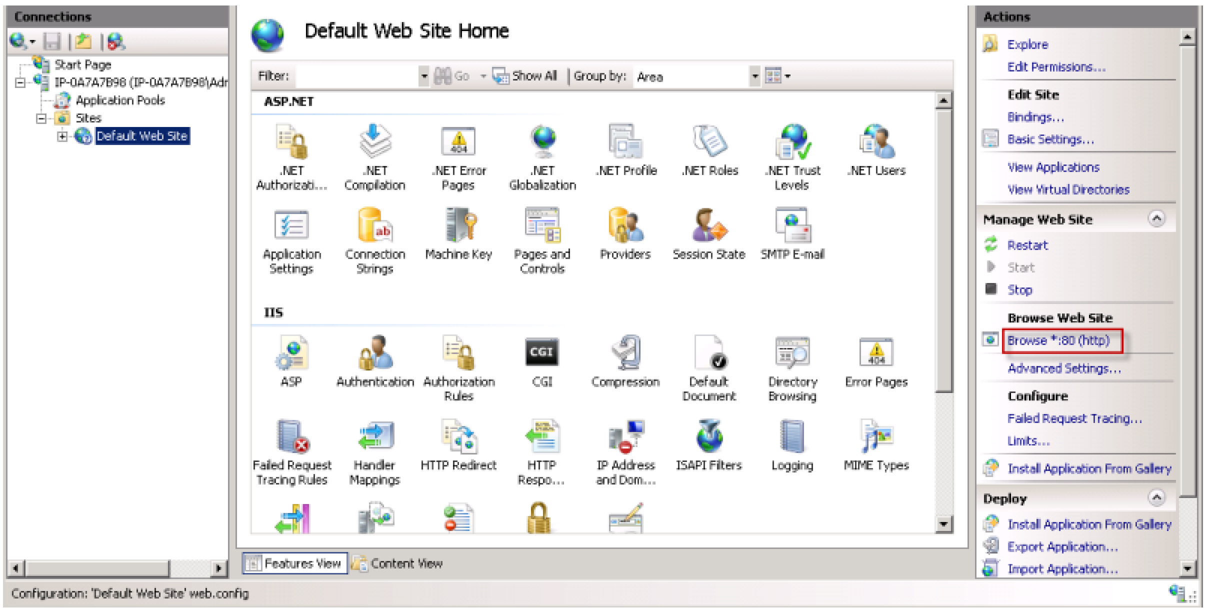
This is my first post.

Now that you can see your WordPress blog on your localhost, you might want to publish this website to other users as your default site on your Amazon EC2 instance. 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

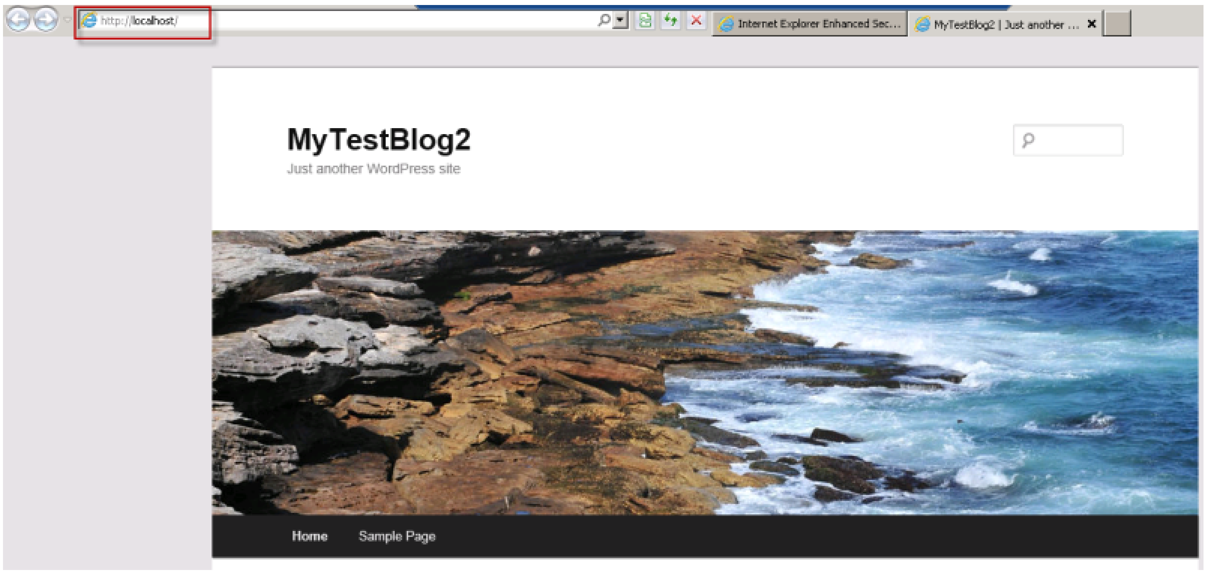
1. Navigate back to IIS, expand the **localhost** and **Sites**nodes, select the **Default Web Site**. In the **Actions** pane on the right, In the **Browse Web Site** group box of the **Internet Information Services (IIS) Manager** window, click **Browse *:80 (http)**.



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

Note

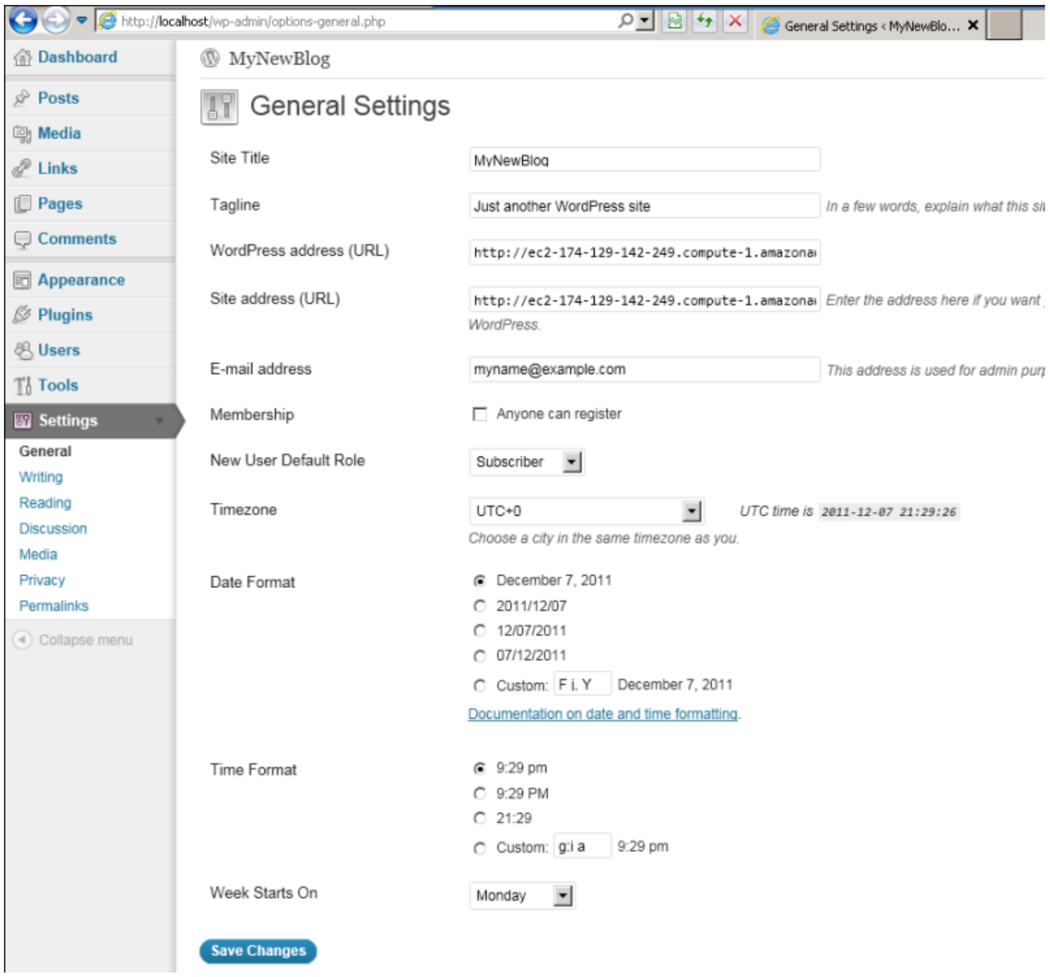
If you do not see your WordPress site, click the **Refresh** button.


http://localhost/
Internet Explorer Enhanced Sec...
MyTestBlog2 | Just another ...
MyTestBlog2
Just another WordPress site
Home
Sample Page

3. Click **Log In** and enter the username and password. The WordPress **Dashboard** appears.
4. In the **Dashboard** left navigation 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 *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.



Site Title: MyNewBlog
Tagline: Just another WordPress site — In a few words, explain what this si
WordPress address (URL): http://ec2-174-129-142-249.compute-1.amazona
Site address (URL): http://ec2-174-129-142-249.compute-1.amazona — Enter the address here if you want WordPress.
E-mail address: myname@example.com — This address is used for admin pur
Membership: ☐ Anyone can register
New User Default Role: Subscriber
Timezone: UTC+0 — UTC time is 2011-12-07 21:29:26 — Choose a city in the same timezone as you.
Date Format: ● December 7, 2011 / ○ 2011/12/07 / ○ 12/07/2011 / ○ 07/12/2011 / ○ Custom: F j, Y December 7, 2011 — Documentation on date and time formatting.
Time Format: ● 9:29 pm / ○ 9:29 PM / ○ 21:29 / ○ Custom: g:i a 9:29 pm
Week Starts On: Monday

   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 your 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.

Installing the Amazon EC2 Command Line Tools on Windows

This section describes how to install the Amazon EC2 command line tools, a set of tools that you can run from the Windows command line that closely mimics the Amazon EC2 API functions.

Process for Installing the Command Line API Tools

Task 1: Download the Command Line Tool

The command line tool is available as a ZIP file on the Amazon EC2 API Tools. The tool is written in Java and includes shell scripts for both Windows and Linux/UNIX/Mac OSX. The ZIP file is self-contained; no installation is required. You just download it and unzip it.

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

Task 2: Set the JAVA_HOME Environment Variable

The Amazon EC2 command line tool reads an environment variable (`JAVA_HOME`) on your computer to locate the Java runtime. The command line tool requires Java version 6 or later to run. Either a JRE or JDK installation is acceptable.

To set the `JAVA_HOME` environment variable

1. If you do not have Java 1.6 or later installed, download and install Java. 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 directory that contains a subdirectory named `bin` that in turn contains the Java executable. For example, if your Java executable is in `C:\jdk\bin`, set `JAVA_HOME` to `C:\jdk`.

   a. On the machine from which you will connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.

      Note

      These instructions are written for a Windows 7 client. The steps may vary depending on the version of Windows you are using.

   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 version. For example, "C:\Program Files (x86)\Java\jre7"

      Note

      Don't include the bin directory in `JAVA_HOME`; that's a common mistake some users make. The command line tool won't work if you do.

   g. Click **OK**.

3. Add your Java directory to your path before other versions of Java.

   a. In **System variables**, select **Path**, and then click **Edit**.

   b. In **Variable values**, before any other versions of Java add **`;%JAVA_HOME%\bin;`**.

4. Verify your `JAVA_HOME` setting with the command **`%JAVA_HOME%\bin\java -version`**.

```
C:\> %JAVA_HOME%\bin\java -version
java version "1.7.0_01"
Java(TM) SE Runtime Environment (build 1.7.0_01-b08)
Java HotSpot(TM) Client VM (build 21.1-b02, mixed mode, sharing)
```

Task 3: Set the EC2_HOME Environment Variable

The command line tool depends on an environment variable (`EC2_HOME`) to locate supporting libraries. You'll need to set this environment variable before you can use the tool.

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-w.x.y.z` (`w`, `x`, `y`, and `z` are version/release numbers) and contains sub-directories named `bin` and `lib`.

    a. On the machine from which you will connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.

    Note

    These instructions are written for a Windows 7 client. The steps may vary depending on the version of Windows you are using.

    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\ec2-1.0.12.0"`***.

2. Add the tool's `bin` directory to your system `PATH`. 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`**.

Task 4: Set the EC2_PRIVATE_KEY and EC2_CERT Environment Variables

The command line tools need access to an X.509 certificate and a corresponding private key that are associated with your account. Amazon EC2 uses the certificate and private key to verify that the commands that you issue come from your account. You can create up to two pairs of X.509 certificates and private keys.

You can either specify your credentials with the *`--private-key`* and *`--cert`* parameters every time you issue a command or you can create environment variables that point to the credential files on your local system. If the environment variables are properly configured, you can omit the parameters when you issue a command.

You can use an existing X.509 certificate and private key, or you can create a new certificate and private key. The following procedure describes how to create a new X.509 certificate and private key and how to create environment variables that point to your credentials.

To set up security credentials for your command line tool

1. If you want to create a new certificate and private key, log in to the AWS security credentials website.

a. Scroll down to the **Access Credentials** section and select the **X.509 Certificates** tab.

b. Click **Create a new Certificate**.

   The **X509 Certificate Created** page appears.

   Note

   The **Create a new Certificate** is available only if you have fewer than two existing X.509 certificates.

c. Select **Download Private Key File** and save it to a convenient location on your computer.

   Important

   Store your Private Key file in a secure location. If you lose your Private Key file you will need to create a new certificate to use with your account. AWS does not store Private Key Information.

d. Select **Download X.509 Certificate** and save it to a convenient location on your computer.

e. Click **Close** to close the **X509 Certificate Created** page.

2. Set the `EC2_CERT` environment variable to the fully qualified path of an existing X.509 certificate or the one you just created.

   a. On the machine from which you will connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.

      Note

      These instructions are written for a Windows 7 client. The steps may vary depending on the version of Windows you are using.

   b. Click **Advanced system settings**.

   c. Click **Environment Variables**.

   d. Under **System variables**, click **New**.

   e. In **Variable name**, type **`EC2_CERT`**.

   f. In **Variable value**, type the path to the directory where you saved your certificate. For example, **`C:\aws\certs\cert\pk-B2QQ72ELI6RDHOOYX6RK6ZPAWEXAMPLE.pem`**.

3. Set the `EC2_PRIVATE_KEY` environment variable to the fully qualified path of the private key that is associated with the X.509 certificate that `EC2_CERT` now points to.

   a. On the machine from which you will connect to Amazon Web Services, click **Start**, right-click **Computer**, and then click **Properties**.

      Note

      These instructions are written for a Windows 7 client. The steps may vary depending on the version of Windows you are using.

b. Click **Advanced system settings**.
c. Click **Environment Variables**.
d. Under **System variables**, click **New**.
e. In **Variable name**, type **`EC2_PRIVATE_KEY`**.
f. In **Variable value**, type the path to the directory where you saved your private key. For example, **`C:\aws\certs\key\pk-LJFUG4MXPQD6VAHJBBQFCQB7KEXAMPLE.pem`**.

Task 5: Set the Region

By default, the Amazon EC2 tools use the Eastern United States 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` parameter 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 go to Regions and Endpoints in the *Amazon Web Services General Reference*.
2. If you want to change the service endpoint, set the `EC2_URL` environment variable.

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

```
C:\> setx EC2_URL https://ec2.eu-west-1.amazonaws.com
```

You're ready to start using Amazon EC2.

Amazon EC2 Infrastructure

Basic Infrastructure Components of Amazon EC2

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

Amazon Machine Images and Instances

An *Amazon Machine Image (AMI)* is a template that contains a software configuration (e.g., operating system, application server, and applications). From an AMI, you launch *instances*, which are running copies of the AMI. You can launch multiple instances of an AMI. For more information on using Windows AMIs, see Using Windows AMIs (p. 46).

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.

You can use a single AMI or multiple AMIs depending on your needs. From a single AMI, you can launch different types of instances. An instance type is essentially a hardware archetype with differing compute and memory facilities. You select a particular instance type based on the amount of memory and computing power you need for the application or software that you plan to run on the instance. For more information about the available Windows instance types, see Windows Instance Types.

Regions and Availability Zones

Amazon has data centers in different areas of the world (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 page).

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

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

Amazon S3

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

Amazon EBS Volumes

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

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

You can attach multiple volumes to an instance. To keep a back-up copy, you can create a snapshot of the volume that 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 one. For more information about Amazon EBS volumes, see Amazon Elastic Block Store.

Instance Store

Some EC2 instance types comes with a preconfigured block of pre-attached disk storage. This is called an instance store; it is also known as an ephemeral store. An instance store is dedicated to a particular instance, and it provides temporary block-level storage. The data on the instance store volumes persists only during the life of the associated Amazon EC2 instance. The amount of this storage ranges from 160 GiB to up to 1.7 TiB and varies by Amazon EC2 instance type. Larger Amazon EC2 instances have more instance volumes and of larger sizes. For more information, see Amazon EC2 Instance Storage.

Root Device Storage

When Amazon EC2 was first introduced, all AMIs were *backed by the Amazon EC2 instance store*, which means that the root device for an instance launched from the AMI is stored in the instance store. After we introduced Amazon EBS, we also introduced AMIs that are *backed by Amazon EBS*, which means that the root device for an instance launched from the AMI is an Amazon EBS volume. At present we provide support for instance store-backed AMIs only on Windows Server 2003. All Amazon EC2 Windows AMIs on Windows Server 2008 and later are backed by Amazon EBS. The description of an AMI includes which type it is (you'll see the root device referred to in some places as either *ebs* (for Amazon EBS-backed) or *instance-store* (for Amazon instance store-backed). For more information, see Root Device Storage on Windows AMIs (p. 52)

Networking and Security

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

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

Monitoring, Auto Scaling, and Load Balancing

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

- Monitor basic statistics for your instances and Amazon EBS volumes.
  For more information, see CloudWatch Developer Guide.
- Automatically scale your Amazon EC2 capacity up or down according to conditions you define.
  For more information, go to the Auto Scaling Developer Guide.
- Automatically distribute incoming application traffic across multiple Amazon EC2 instances.
  For more information, go to the Elastic Load Balancing Developer Guide.

AWS Identity and Access Management

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

- Create users and groups under your organization's AWS account
- Easily share your AWS account resources between the users in the account
- Assign unique security credentials to each user
- Granularly control user's 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 more information, go to Using AWS Identity and Access Management.

Available Interfaces

AWS provides different interfaces to access EC2.

- **AWS Management Console** The AWS Management Console is a simple web-based GUI. For more information about using the console, go to the Amazon Elastic Compute Cloud Getting Started Guide.
- **Command Line Tools (API Tools)** EC2 provides a Java-based command-line client that wraps the EC2 SOAP API. For more information on installing command line tools, see Installing the Amazon EC2 Command Line Tools on Windows (p. 37).

- **Programmatic Interface** The following table lists how you can access EC2 programmatically.

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

Using Windows AMIs

An Amazon Machine Image (AMI) is the image of the machine that contains all the information necessary to boot instances of your software. It is somewhat similar to a snapshot of the boot partition containing the operating system and installed software running on your server. You use these images to launch your Amazon EC2 Windows instances, which are running copies of the AMI.

For more information on Amazon Windows AMIs, see Amazon Windows AMI Basics (p. 46).

You can use the AWS Management Console) to search for Windows AMIs that meet specific criteria, and then launch instances of those AMIs. 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. 49).

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. 53).

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. 66).

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. 72).

To help categorize and manage your AMIs, you can assign *tags* of your choice to them. For more information, go to 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. You first choose the AMI that meets your specific requirements, launch an EC2 instance (virtual server) off of that AMI, connect to the instance just as you would connect to a virtual server, and then use that EC2 instance just as you would use a traditional hardware-based Windows server.

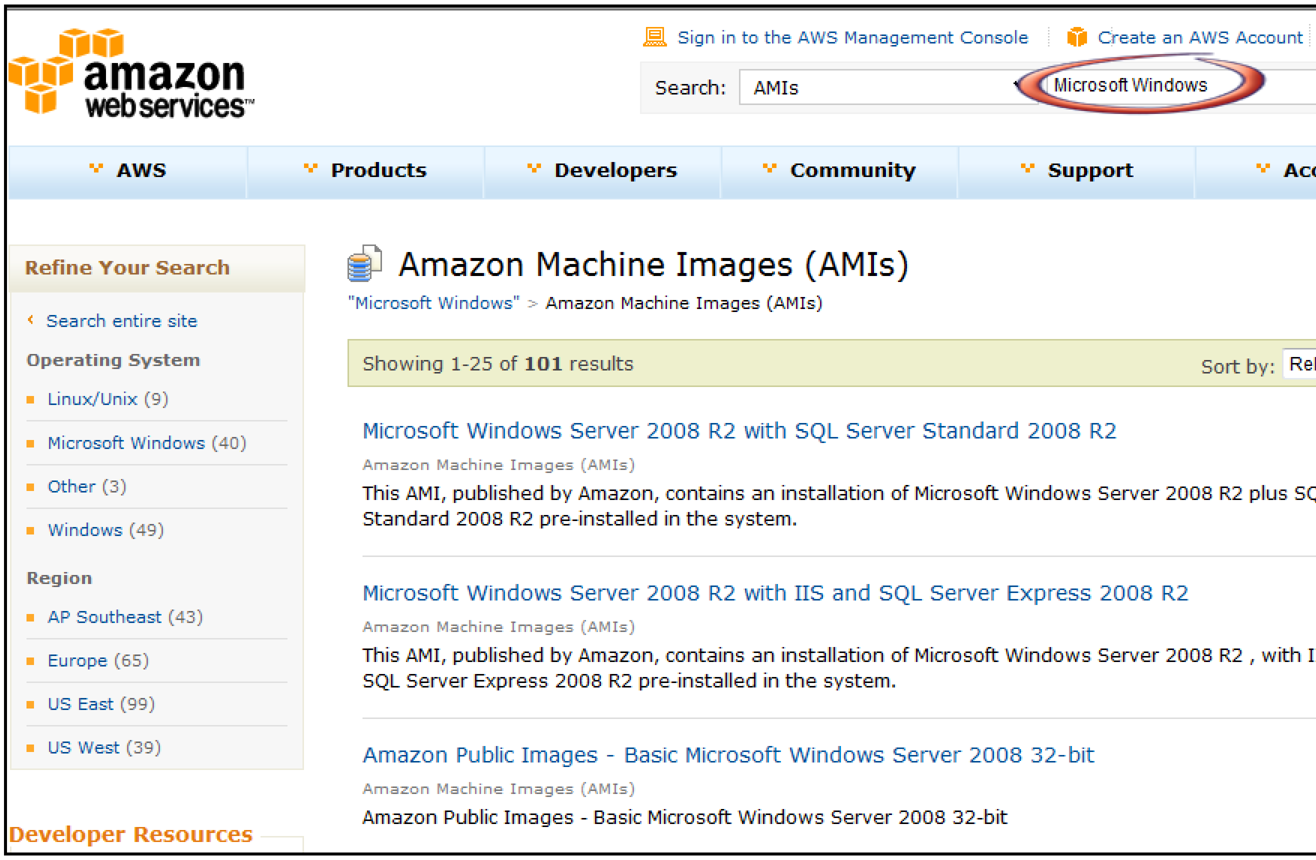
Amazon AWS currently provides the following basic versions of Windows AMIs.

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

AWS also provides a set of publicly available AMIs that are pre-bundled with 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 pre-bundled 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 AMIs 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 AMI, customize the instance by installing 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.

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 mentioning the platform name *Microsoft Windows* in the search box as shown in the following screenshot.



Amazon Machine Images (AMIs)

"Microsoft Windows" > Amazon Machine Images (AMIs)

Showing 1-25 of **101** results

Microsoft Windows Server 2008 R2 with SQL Server Standard 2008 R2

Amazon Machine Images (AMIs)

This AMI, published by Amazon, contains an installation of Microsoft Windows Server 2008 R2 plus SQ
Standard 2008 R2 pre-installed in the system.

Microsoft Windows Server 2008 R2 with IIS and SQL Server Express 2008 R2

Amazon Machine Images (AMIs)

This AMI, published by Amazon, contains an installation of Microsoft Windows Server 2008 R2 , with I
SQL Server Express 2008 R2 pre-installed in the system.

Amazon Public Images - Basic Microsoft Windows Server 2008 32-bit

Amazon Machine Images (AMIs)

Amazon Public Images - Basic Microsoft Windows Server 2008 32-bit

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

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 as a local system 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 Terminal Services
- Syncs the instance clock with a time server

After you launch your Windows instance with the 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. The instances launched from the customized AMI will then be 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` directory on 32-bit instances and in the `%ProgramFiles(x86)%\Amazon` directory on 64-bit instances. For more information, see Creating Your Own Windows AMI (p. 53).

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` directory on the 32-bit instances and on `%ProgramFiles(x86)%\RedHat` directory on 64-bit instances. The two drivers are the `rhelnet`, the RedHat Paravirtualized network driver, and `rhelscsi`, the RedHat SCSI miniport driver.

Keeping Your Instances Updated

The Amazon Windows AMIs provided at the initial launch of your Windows instances contain all the latest security updates. However, once you launch your instance, you are responsible for managing future updates, including the updates issued after the AMI was built. 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 appropriate for that software. We recommend that you first run the Windows Update service with every new Windows instance that you launch.

Note

An Amazon EC2 Windows instance can be rebooted 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. 52).

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 on 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 the AMI that you want to work on. The AMI can either be provided by Amazon, or provided by the Amazon EC2 community, or created by you. However, if you want to create your own AMI, you must start by using one of the base AMIs provided.

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. In the Amazon EC2 console, click **AMIs** in the **Navigation** pane.

   The console displays a list of all available AMIs. Use the **Viewing** options to narrow the list of displayed AMIs.
3. To see a list of all Windows AMIs provided by Amazon, select **Amazon Images** from the **Viewing** drop-down list.
4. Select **Windows** from the list.
5. In the list of Windows AMIs, click an AMI to view its properties. The AMI properties appear in the **Description** tab in the lower pane.

Using Command Line Tools

Amazon EC2 provides a Java-based command-line client that wraps the EC2 SOAP API. You must have installed the command line tool on your machine to use the Java-based command-line client. For information on installing command line tools see Installing the Amazon EC2 Command Line Tools on Windows (p. 37).

To find a suitable AMI

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

  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*.

The following example shows the command and the resulting output consisting of a partial list of all Amazon Windows AMIs.

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

IMAGE   ami-de24d6b7    amazon/Windows-2008R2-SP1-Base-Locale-JA-JP-v101
amazon  available  public  x86_64  machine                 windows
ebs     hvm     xen BLOCKDEVICEMAPPING      /dev/sda1               snap-
3e2c5e52   35
IMAGE   ami-d024d6b9    amazon/Windows-2008R2-SP1-Base-Locale-KO-KR-v101
amazon  available public  x86_64  machine                 windows
ebs     hvm     xen BLOCKDEVICEMAPPING      /dev/sda1               snap-
022c5e6e   35
IMAGE   ami-b424d6dd    amazon/Windows-2008R2-SP1-Base-Locale-ZH-CN-v101
amazon  available public  x86_64  machine                 windows
ebs     hvm     xen  BLOCKDEVICEMAPPING      /dev/sda1               snap-
4612602a   35
IMAGE   ami-b624d6df    amazon/Windows-2008R2-SP1-Base-Locale-ZH-HK-v101
amazon  available public x86_64  machine                 windows
ebs     hvm     xen BLOCKDEVICEMAPPING      /dev/sda1               snap-
5a126036   35
IMAGE   ami-98ef1df1    amazon/Windows-2008R2-SP1-Core-v101 amazon available
 public  x86_64  machine windows ebs hvm
xen  BLOCKDEVICEMAPPING      /dev/sda1               snap-1af16b76   16
IMAGE   ami-1cbd4475    amazon/Windows-2008R2-SP1-English-Base-v103
amazon available public  x86_64  machine  windows ebs
hvm     xen BLOCKDEVICEMAPPING      /dev/sda1               snap-2098704e
  35
IMAGE   ami-42bd442b    amazon/Windows-2008R2-SP1-English-SQLExpress-v102
amazon  available public  x86_64  machine windows
ebs     hvm     xen  BLOCKDEVICEMAPPING      /dev/sda1               snap-
ec896182   35
IMAGE   ami-baed1fd3    amazon/Windows-2008R2-SP1-English-SQLStandard-v101
 amazon  available   public   x86_64  machine windows
ebs     hvm     xen  BLOCKDEVICEMAPPING      /dev/sda1               snap-
26de574a   35
IMAGE   ami-988577f1    amazon/Windows-2008R2-SP1-MultiLang-Base-v101
amazon  available  public  x86_64  machine  windows ebs
hvm     xen   BLOCKDEVICEMAPPING      /dev/sda1               snap-d035a2bc
   35
IMAGE   ami-0a8a7863    amazon/Windows-2008R2-SP1-MultiLang-SQLExpress-v101
 amazon  available       public   x86_64  machine windows
ebs     hvm     xen   BLOCKDEVICEMAPPING      /dev/sda1               snap-
12cb5d7e   35
IMAGE   ami-a8e705c1    ec2-paid-ibm-images/ibm-infosphere-is-winclient.mani
fest .xml    amazon  available public  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
IMAGE   ami-db20c3b2    ec2-public-windows-images/SqlSvrExp2003r2-i386-Win-
v1.07 .manifest.xml   amazon  available  public  i386  machine
        windows instance-store  hvm     xen
IMAGE   ami-d920c3b0    ec2-public-windows-images/SqlSvrExp2003r2-x86_64-
Win-v1. 07.manifest.xml amazon  available public  x86_64  machine
```

```
        windows instance-store  hvm     xen
IMAGE   ami-0535d66c    ec2-public-windows-images/SqlSvrStd2003r2-x86_64-
Win-v1. 07.manifest.xml amazon  available public x86_64  machine
        windows instance-store  hvm     xen
```

Using the Query API

To find a suitable AMI

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

```
https://ec2.amazonaws.com/
?Action=DescribeImages
&Owner.1=amazon
&Filter.1.platform=windows

&...auth parameters...
```

Following is an example response.

```
<DescribeImagesResponse xmlns="http://ec2.amazonaws.com/doc/2012-04-01/">
  <imagesSet>
   <item>
          <imageId>ami-dde40db4</imageId>
          <imageLocation>amazon/Windows-Server2008-x86_64-SqlStandard-
v103</imageLocation>
          <imageState>available</imageState>
          <imageOwnerId>206029621532</imageOwnerId>
          <isPublic>true</isPublic>
          <architecture>x86_64</architecture>
          <imageType>machine</imageType>
          <platform>windows</platform>
          <imageOwnerAlias>amazon</imageOwnerAlias>
          <name>Windows-Server2008-x86_64-SqlStandard-v103</name>
          <description>Microsoft Windows 2008 Datacenter 64-bit, Sql 2008
Standard AMI v1.03</description>
          <rootDeviceType>ebs</rootDeviceType>
          <rootDeviceName>/dev/sda1</rootDeviceName>
          <blockDeviceMapping>
            <item>
              <deviceName>/dev/sda1</deviceName>
              <ebs>
                <snapshotId>snap-349c275c</snapshotId>
                <volumeSize>30</volumeSize>
                <deleteOnTermination>true</deleteOnTermination>
              </ebs>
            </item>
          </blockDeviceMapping>
          <virtualizationType>hvm</virtualizationType>
          <hypervisor>xen</hypervisor>
        </item>
  </imagesSet>
```

An Amazon EC2 Windows instance can be launched from an AMI either *backed by instance store* or *backed by an Amazon Elastic Block Store (Amazon EBS)* volume. Instances launched from an instance store-backed AMI use the instance's instance store as the root device (for example, C:\). An instance launched from an AMI backed by Amazon EBS uses an Amazon EBS volume as its root device. The **Root Device** column on the console indicates whether the AMI is backed by an instance store (instance-store) or Amazon EBS (ebs). There are significant differences between the two types of AMIs. It is essential to consider those differences when you choose an AMI to launch your instance. Refer to the following section for a quick summary of the differences.

After locating your desired AMI, record its AMI ID. You can use the AMI ID to launch and then connect to your instance. For more information on launching and using your Windows instance, go to Launching and Using Instances. For information on connecting to your Windows instance, go to Connecting to Windows Instances.

Root Device Storage on Windows AMIs

The root device of an *instance store-backed AMI* is initially stored in Amazon S3. When an instance is launched using an instance store-backed AMI, an image of the root device is copied from Amazon S3 to the root partition of the instance's instance store. This image is then used to boot the instance.

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

There are other important differences between the two types of AMIs. The following table gives a quick summary of the differences.

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

Creating Your Own Windows AMI

When you are connected to your Amazon Windows EC2 instance, you can use it just like you use a hardware-based Windows Server instance, but with cloud functionality. There are several ways you can use your Windows instance:

- Use the instance as is for a specific task and duration, and stop or terminate the instance when your task is done.
- Customize your instance by installing software, applications, and additional storage to use the instance for a specific task and duration. For example, you can use an Amazon Windows-2008R2-SP1-MultiLang-SQLStandard AMI as a base, install Visual Studio Team Foundation Server, and attach Amazon EBS volumes for additional storage.

  Note

  An Amazon Windows EC2 instance can be rebooted after installing software and applications. Rebooting works the same way for both *instance store-backed* and *Amazon EBS-backed instances*.

- You can 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 on launching, connecting, and using your Windows instance, see Using Instances.

Before you create your own AMI, you can configure your base customized instance. The new configuration will be applied to all the instances that will be launched from the new AMI. Your Amazon Windows instance comes with a configuration tool, EC2Config Service. You can use this tool to configure your instance. For information on using the EC2 Config Service, see Using EC2Config Service (p. 53)

The root storage device you select for the AMI determines the process you follow to create the AMI. The AMI will be either an Amazon EBS-backed AMI or an Amazon EC2 instance store-backed AMI. There are significant differences between Amazon EBS-backed and Amazon EC2 instance store-backed AMIs, including AMI size limits and storage and persistence of data. For information on the differences between these choices, see Root Device Storage on Windows AMIs (p. 52).

For detailed instructions on creating an Amazon EBS-backed Windows AMI, see Creating an Amazon EBS-Backed Windows AMI (p. 60). For detailed instructions on creating an instance store-backed Windows AMI, see Creating an Instance Store-Backed Windows AMI (p. 62).

Using EC2Config Service

EC2Config service can perform various functions to prepare an instance when it first boots up. Each of these functions can be enabled or disabled from the settings before creating a new AMI. The new settings are applied to the instance that will be launched from the new AMI.

EC2Config settings can be `enabled` or `disabled` using either the EC2Config Service Properties user interface tool or by directly editing the XML files. Some advanced configuration settings are currently not available on the user interface tool. For advanced modifications use the XML files. The following procedures describe the user interface tool and the XML files.

Important

Be sure to select the **Set Password** and **Enable SetPassword feature after sysprep** check boxes unless you have included your own Administrator password in the Sysprep configuration file. Otherwise you won't be able to log in to the AMI.

We Sysprep all the Amazon base Windows AMIs before registering and making them available to the public. So, when you launch an Amazon Windows AMI, it has to go through the inital Sysprep cycle. Sysprep is a Microsoft tool that prepares an AMI for multiple launches. However, after you launch your instance and the instance configures itself, Sysprep should not run unless you manually invoke it.

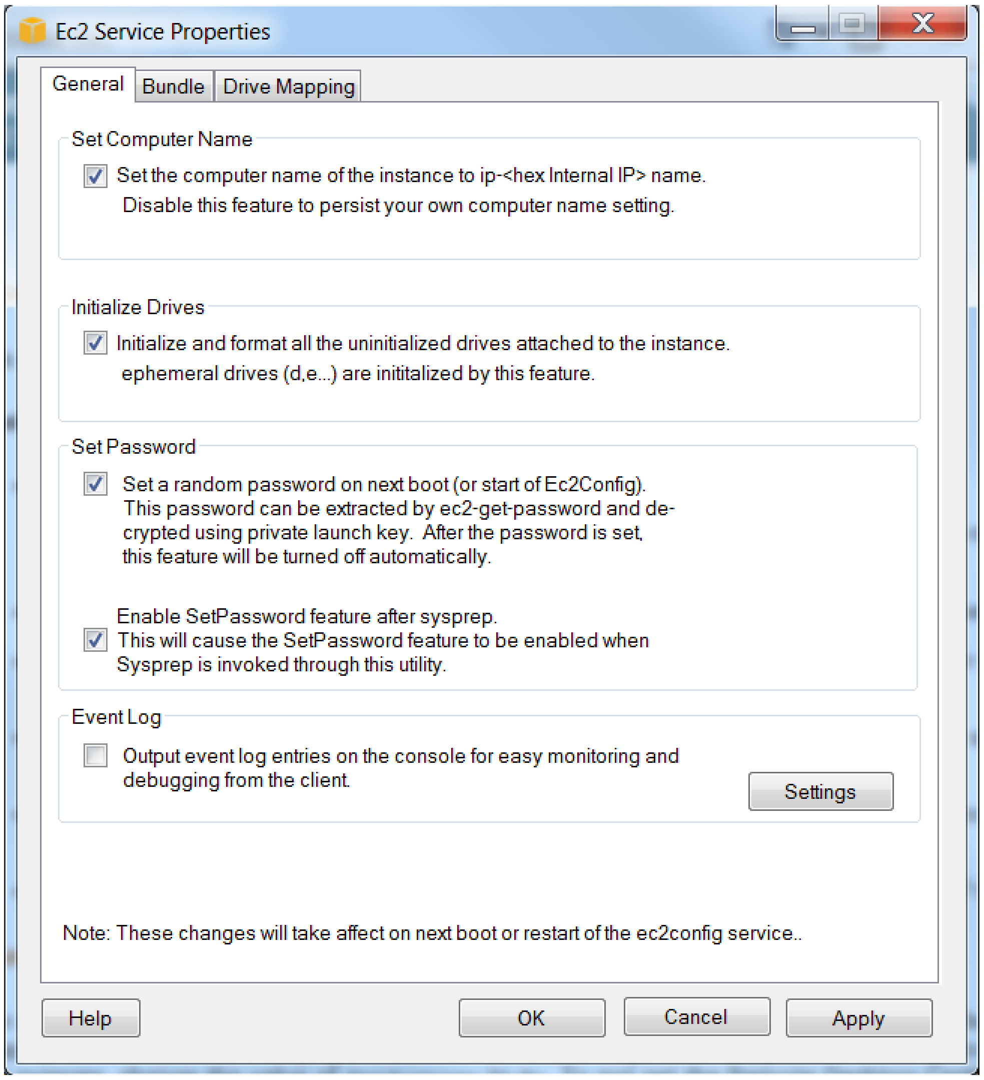
You can manually invoke the Sysprep tool to prepare your instance to create a new Windows AMI. The process of customizing and preparing an instance for creating a new AMI is also called *bundling*. The following procedure describes steps to bundle your instance using the EC2 Service Properties user interface.

Using the EC2Config Service User Interface Tool

Topics

To change the EC2Config settings on your Windows instance using the EC2Config tool

1. Launch and connect to your Windows instance.
2. Go to `C:\Program Files\Amazon\EC2ConfigService` and double-click the **Ec2ConfigServiceSettings** application.
3. Your Windows instance displays the **Ec2 Service Properties** dialog box.



Set Computer Name

☑ Set the computer name of the instance to ip-<hex Internal IP> name.
Disable this feature to persist your own computer name setting.

Initialize Drives

☑ Initialize and format all the uninitialized drives attached to the instance.
ephemeral drives (d,e...) are inititalized by this feature.

Set Password

☑ Set a random password on next boot (or start of Ec2Config).
This password can be extracted by ec2-get-password and de-crypted using private launch key. After the password is set, this feature will be turned off automatically.

☑ Enable SetPassword feature after sysprep.
This will cause the SetPassword feature to be enabled when Sysprep is invoked through this utility.

Event Log

☐ Output event log entries on the console for easy monitoring and debugging from the client.

Note: These changes will take affect on next boot or restart of the ec2config service..

4. Use the **Ec2 Service Properties** dialog box to enable or disable your settings. In the **General Properties** tab you can adjust the following settings:

   a. **Set Computer Name** Enabled by default, sets the hostname of the instance to a unique name based on the IP address of the instance and reboots once after booting. If you want to set your own hostname, or prevent your existing host name from being modified, you must disable this setting.

   b. **Initialize Drives** Initializes and formats all uninitialized instance stores attached to the instance during startup. When an instance is launched, all instance stores that come with the instance are uninitialized. Enabled by default, this feature initializes and mounts the instance stores as drives D:/, E:/, etc. For more information on instance stores that come with Windows instances, see Amazon EC2 Instance Storage.

   c. **Set Password** Sets a random password on the instance every time you launch an instance, encrypts it with the user launch key, and outputs the encrypted password to the console. This feature is disabled by default after the first launch so that any further reboots or restarts of this instance do not change the password set by the user. Select this check box to generate random passwords every time you launch this instance.

Caution

Do not clear this check box unless you included a password for the Administrator account in the Sysprep configuration file. If you do not modify the Sysprep configuration file to include a password for the Administrator account, you must select this check box and the **Enable SetPassword feature after sysprep** check box to correctly generate a password for the AMI.

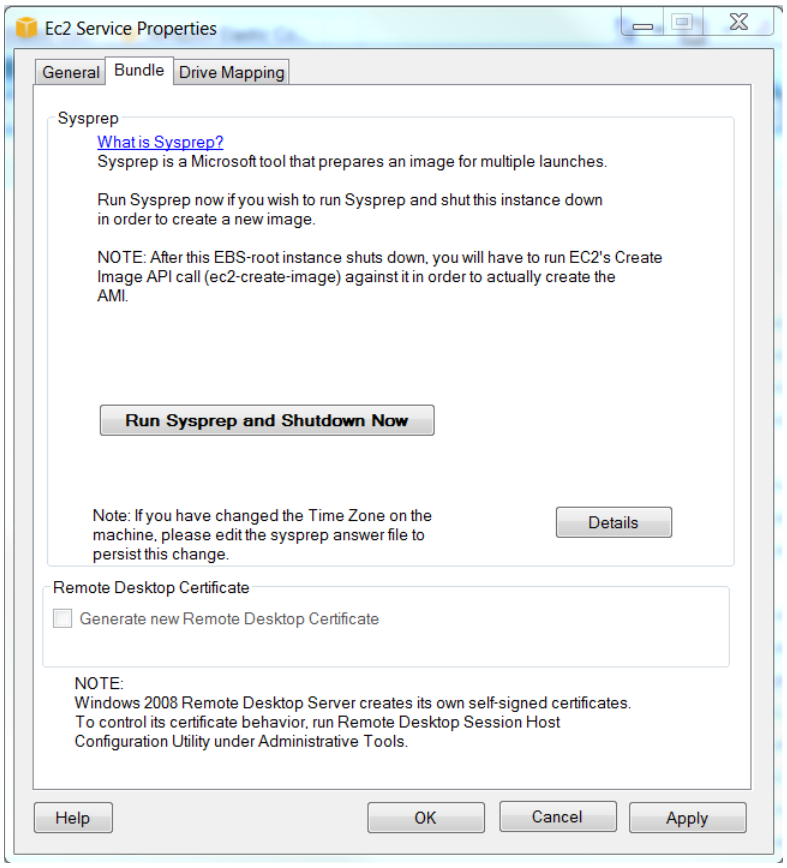
d. **Enable SetPassword feature after sysprep** Enabled by default, sets a random password after you have used Sysprep feature to create an AMI from this instance.

Caution

Do not clear this check box unless you included a password for the Administrator account in the Sysprep configuration file. If you do not modify the Sysprep configuration file to include a password for the Administrator account, you must select this check box and the **Set Password** check box to correctly generate a password for the AMI. In addition, if you do not include a password for the Administrator account in the Sysprep configuration file and you clear this check box, the AMI's password will be unknown and you will not be able to log in to the AMI.

e. **Event Log** Click the empty check box to enable this setting to put eventlog entries on the console during boot for easy monitoring and debugging. Click **Settings>** in the **EventLog** box to specify filters for the log entries that will be sent to the console output. By default, the three most recent error entries from the System event log are sent to the console.

5. Click the **Bundle** tab to open the Sysprep page.



Sysprep

What is Sysprep?

Sysprep is a Microsoft tool that prepares an image for multiple launches.

Run Sysprep now if you wish to run Sysprep and shut this instance down in order to create a new image.

NOTE: After this EBS-root instance shuts down, you will have to run EC2's Create Image API call (ec2-create-image) against it in order to actually create the AMI.

Note: If you have changed the Time Zone on the machine, please edit the sysprep answer file to persist this change.

Remote Desktop Certificate

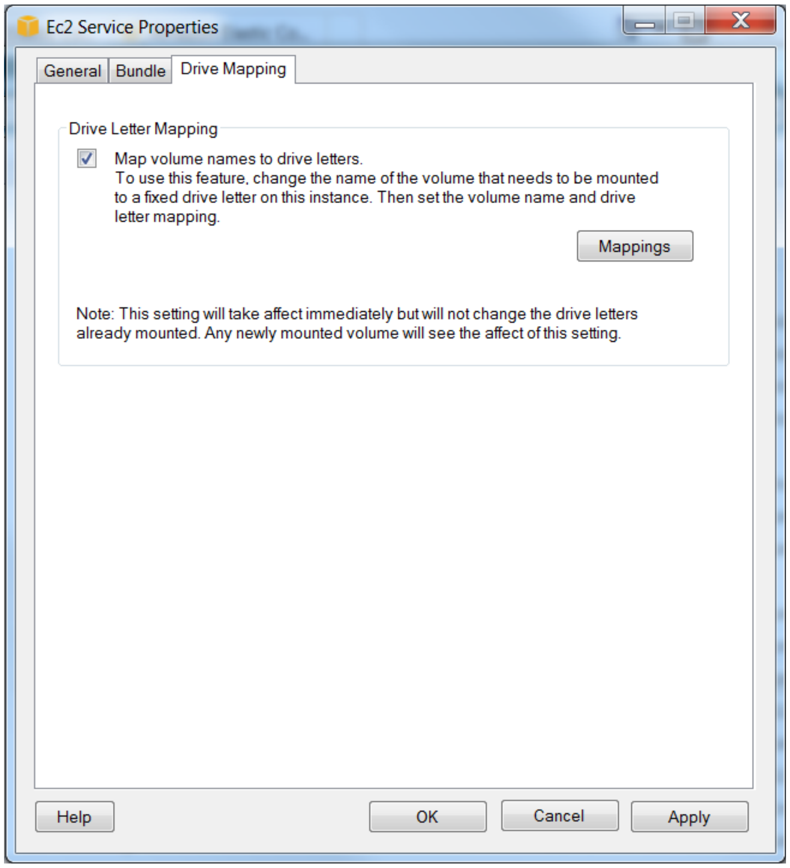
☐ Generate new Remote Desktop Certificate

NOTE:
Windows 2008 Remote Desktop Server creates its own self-signed certificates. To control its certificate behavior, run Remote Desktop Session Host Configuration Utility under Administrative Tools.

6. Click **Run Sysprep and Shutdown Now** to prepare your instance for creating a new AMI, click **Apply** and then click **OK**.

   By default, when an Amazon EBS volume is attached to an instance, it may show up as any drive letter on the instance. You can specify the drive letters of the mounted volumes by mapping the name of a volume to a drive letter.

7. Click the **Drive Mapping** tab to specify your drive mapping.
   The **Drive Letter Mapping** setting is enabled by default. This means that the drives will be mapped to drive letters, but the system decides the mapping.



Drive Letter Mapping

[✓] Map volume names to drive letters.
To use this feature, change the name of the volume that needs to be mounted to a fixed drive letter on this instance. Then set the volume name and drive letter mapping.

Note: This setting will take affect immediately but will not change the drive letters already mounted. Any newly mounted volume will see the affect of this setting.

8. To specify your own mapping, click **Mappings**.
   a. In the **DriveLetterSetting** dialog box, type in the name (disk label) of the volume.
   b. Click the drop-down arrow in the **Drive Letter** box and select the drive letter for the volume.
   c. Keep adding the volume names and the drive letters you want to map it with.
   d. Click **OK** to close the **DriveLetterSetting** dialog box.

9. Click **OK** to close the **Ec2 Service Properties** dialog box.

If you have specified your drive letter mapping, the settings will take effect immediately on volumes you attach after following this procedure. This setting will not change the drive letters on already mounted volumes. However, the setting will fail if the drive letter is already in use. We recommend that you pick drive letters from the end of the alphabet (Z, Y, X, and so on) to avoid this problem.

Using EC2Config XML Files

Topics

By default, the EC2Config service is installed on all public Amazon AMIs. The binaries and additional tools needed to configure the new Windows AMI are contained in the `%ProgramFiles%\Amazon\EC2ConfigService` directory. You can modify the following configuration and settings files located in the directory.

Config.xml File

- **Ec2SetPassword** Generates a new random encrypted password every time you launch an instance. This feature is disabled by default after the first launch so that any further reboots or restarts of this instance do not change the password set by the user. Change this setting to `Enabled` to continue generating random passwords every time you launch an instance.

  This setting is important, if you are planning on creating an AMI from your instance. Before you configure this setting you have to decide whether you want the instances launched from the customized AMI to have random passwords generated.
- **Ec2SetComputerName** Enabled by default, sets the hostname of the instance to a unique name based on the IP address of the instance and reboots once after booting. If you want to set your own host name, or prevent your existing host name from being modified, you must disable this setting.
- **Ec2InitializeDrives** Initializes and formats all uninitialized instance stores attached to the instance during startup. When an instance is launched, all instance stores that come with the instance are uninitialized. `Enabled` by default, this feature initializes and mounts the instance stores as drives D:/, E:/, etc. For more information on the instance stores that come with Windows instances, go to Amazon EC2 Instance Storage.
- **Ec2EventLog** Puts event log entries in the console based on the configuration of the eventlogconfig file. By default, the three most recent error entries from the system event log are sent to the console. To specify the event log entries to output to the console, edit the **EventLogConfig.XML** file located in **Settings** directory.

  For information on the settings in this file, go to the Microsoft MSDN website.
- **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 instances since Windows Server 2008 is able to generate its own certificates.
- **Ec2OutputRDPCert** This setting is `Enabled` by default and copies the Remote Desktop certificate information to the console, so the user can verify it against the thumbprint.
- **Ec2SetDriveLetter** Sets the drive letters of the mounted volumes based on the user defined settings. By default, when an Amazon EBS volume is attached to an instance, it may show up as any drive letter on the instance. To specify your drive letter mappings, edit the **DriveLetterConfig.XML** file located in the **Settings** directory.
- **Ec2WindowsActivate** Enabling this setting causes Windows Server 2008 to attempt activation by searching through the DNS Suffix List for appropriate KMS Server entries. In Windows Server 2008 (not the later Windows Server R2 version), the plug-in performs this search manually. When the appropriate KMS server entries are found, the plug-in sets your activation server to the first server to respond to the request successfully. In Windows Server 2008 R2, the auto discovery method built into Windows Server is able to search the suffix list automatically.

  To modify the settings for the KMS servers, edit the `ActivationSettings.XML` file located in the `Settings` directory.
- **SetDnsSuffixList** Adds entries to your DNS suffix search list to facilitate DNS lookups.

  Note

  This functionality is key to Windows Server 2008 activation. See the preceding list item for more information.

  This plug-in has the ability to make region-specific decisions about the suffix list based on the Availability Zone the instance has been launched in. The default settings have been configured to discover the KMS server in each region. To see the default settings or add settings specific to your requirements, open and edit the **DnsSuffixSettings.XML** file located in the **Settings** directory.

The `BundleConfig.xml` file controls how the EC2Config service prepares an instance for bundling. This includes configuring Sysprep on the system, changing the state of the Ec2ConfigureRDP plug-in, and shutting down the instance for bundling.

BundleConfig.XML

- **AutoSysprep** 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 allows you to securely RDP into the instances. Change the value of this setting to `Yes` if you want the new instances to have the certificate.

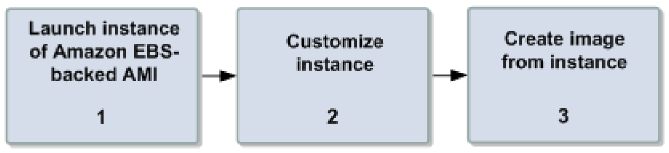
  Note

  This setting is not used in Windows Server 2008, since Windows Server 2008 is able to generate its own certificates.

- **SetPasswordAfterSysprep** Sets a random password on the newly launched instances, encrypts it with the user launch key, and outputs the encrypted password on the console. Change the value of this setting to `No` if you do not want the new instances to set a random encrypted password.

Creating an Amazon EBS-Backed Windows AMI

The process for creating an Amazon EBS-backed AMI is simple. The following diagram and the task lists describe the process.


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

Task List for Creating an Amazon EBS-backed AMI

1. Start by launching an instance of an Amazon EBS-backed AMI that is similar to the AMI you want to create. For example, you might take a public AMI that uses the operating system you want to use for your AMI.

   The instance must be from an Amazon EBS-backed AMI; you can't start with an instance of an Amazon EC2 instance store-backed AMI.

2. When the instance is running, customize it as you want. For example, you could attach additional Amazon EBS volumes, or load applications or data on to the instance.

   Important

   If you customize your instance with additional instance stores, the new AMI contains the mapping information for those instance stores. When you then launch an instance from your new AMI, the instance automatically launches with the additional instance stores initialized and mounted. However, all the attached instance stores are new. They will not contain any data from the instance store of the original AMI.

Important

If you customize your instance with EBS volumes in addition to the root device, the new AMI contains block device mapping information for those volumes. When you then launch an instance from your new AMI, the instance automatically launches with the additional volumes. The data on those volumes will persist.

3. When an instance is launched for the first time, it goes through a cycle of configuration and Sysprep. After the instance boots, some of the configuration settings are either `Enabled` or `Disabled` to persist the settings through instance reboots. Use EC2Config Services tool or EC2Config XML files to change the configuration settings for your new AMI. For more information about Sysprep, see Sysprep Technical Reference.
4. When the instance is set up the way you want it, create an AMI from that instance.

   Amazon EC2 powers down the instance, takes images of any volumes that were attached, creates and registers the AMI, and then reboots the instance. It takes several minutes for the entire process to complete.

You can use the command line, the API, or the AWS Management Console to create your Windows AMIs. For background information about the interfaces and tools, see Available Interfaces (p. 44).

Using the AWS Management Console

For instructions on using the AWS Management Console, go to Creating an Image from a Running Instance.

Using Command Line Tools

To create an Amazon EBS-backed AMI

1. Enter the following command to create an image:

```
C:\>  ec2-create-image -n your_image_name   instance_id
```

   For example:

```
C:\>  ec2-create-image -n "My AMI" i-eb977f82
```

   Amazon EC2 creates an image and returns an AMI ID.

```
IMAGE ami-8675309
```

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

```
C:\>  ec2-describe-images -o self
```

   Amazon EC2 returns information about the AMI.

Using the Query API

To create an Amazon EBS-backed AMI

- Construct the following query request to create an image:

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

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

```
<CreateImageResponse xmlns="http://ec2.amazonaws.com/doc/2012-04-01/">
  <imageId>ami-8675309</imageId>
</CreateImageResponse>
```

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

Creating an Instance Store-Backed Windows AMI

The process for creating an *instance store-backed* AMI is different from the process of creating an *Amazon EBS-backed* AMI. All Amazon EC2 instance store-backed AMIs are loaded from Amazon S3 storage. When you create the AMI, you must upload it to an existing account on Amazon S3. Amazon S3 stores data objects in buckets, which are similar in concept to directories. Buckets have globally unique names and are owned by unique AWS accounts. We refer to this process of creating an AMI as *Bundling*.

Caution

Instance store drives (e.g., the D: drive) are not included in the bundled AMI. Instance store drives and their data are deleted when the instance is terminated. You must store any data that you want to use with the new AMI on the root drive or an Amazon EBS volume. For more information about Amazon EC2 storage options, go to Using Storage.

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

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

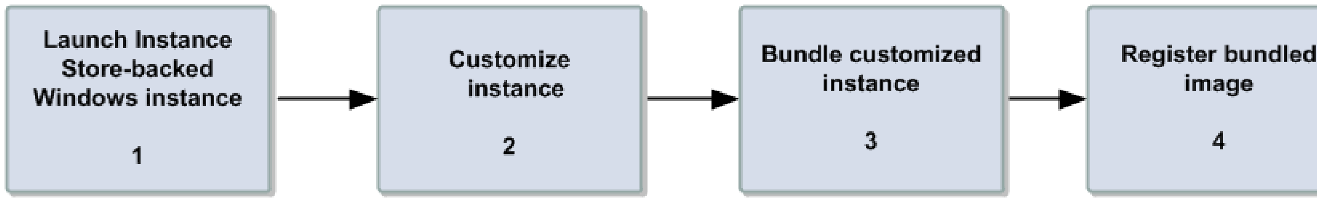
When bundling is complete, the status changes to "complete."

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

Note

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

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


Launch Instance Store-backed Windows instance
1
Customize instance
2
Bundle customized instance
3
Register bundled image
4

Tasks for Creating an Instance Store-Backed Windows AMI

1. Start by launching a Windows instance that is similar to the AMI you want to create and connect to the instance. For information on connecting to a Amazon EC2 Windows instance, go to Connecting to Windows Instances.
2. When you are connected to your instance, customize it as you want. For example, you could attach Amazon EBS volumes, or load applications or data on to the instance.

   Important

   It is very important to remember the following behavior of the *instance store-backed* AMI creation process on the storage drives:

   - The system drive (C:) is automatically attached when the new instances are launched. The data on the C: drive is the only one that will persist when the instance is bundled.
   - Other instance store drives (for example, D:) are temporary drives, and should be strictly used for short-term storage, since they will not persist when instance is bundled.
   - You can add Amazon EBS volumes. The EBS volumes are stored within Amazon S3 buckets and therefore will remain intact when the instance is bundled. We recommend that you store all the data that you need to persist within the Amazon EBS volumes.

3. When an instance is launched for the first time, it goes through a cycle of configuration and Sysprep. After the instance boots, some of the configuration settings are either `Enabled` or `Disabled` to persist the settings through instance reboots. Before bundling an instance, you can configure the instance using the EC2Config Service. Use EC2Config Services tool or EC2Config XML files to change the configuration settings for your new AMI. For more information, see Using EC2Config Service (p. 53). For more information about Sysprep, see Sysprep Technical Reference.
4. Bundle and then register the AMI.
5. After you've prepared your instance, the bundling process performs the following tasks, which are listed in the order that they usually take place:
   - Excludes any instance store drives (i.e., the D: drive on your instance is not included in the bundled AMI)
   - Compresses the image to minimize bandwidth usage and storage requirements
   - Encrypts and signs the compressed image to ensure confidentiality and authenticates the image against its creator
   - Splits the encrypted image into manageable parts for upload

- Runs `Sysprep` to strip out computer-specific information (e.g., the MAC address and computer name) to prepare the Windows image for virtualization
- Creates a manifest file that contains a list of the image parts with their checksums
- Puts all the parts of the AMI into an Amazon S3 bucket you specify

How to Bundle Amazon EC2 Instance Store-Backed Windows AMIs

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

Using the AWS Management Console

Bundling and registering Amazon EC2 instance store-backed Windows images using the console is easy.

To bundle an Amazon EC2 instance store-backed AMIs

1. Log in to the AWS Management Console and click the box next to the drop-down arrow to open a list of services. Click Amazon Elastic Compute Cloud and then select the Windows instance you want to use, and prepare the instance to meet your specifications.

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

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

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

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

   Note

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

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

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

Using the Command Line Tools

To bundle Amazon EC2 instance store-backed AMIs

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

Note

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

2. If you want to reduce your startup time, delete any temporary files on your instance using the Disk Cleanup tool, defragment your system using Disk Defragmenter, and zero out free space using `sdelete -c C:\`.
   You can download the `sdelete` utility from the *sdelete Download Page* or the *Microsoft Web Site*.
3. Enter the following command to bundle the instance into Amazon S3 on your local system where you have installed the API tools (do not enter this command on the instance you are bundling):

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

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

   Note

   To perform this task, you need your AWS Access Key ID (access_key_id) and AWS Secret Access Key (secret_access_key). For more information, go to Amazon EC2 Credentials.

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

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

```
C:\> ec2-bundle-instance -31c2425a -b myawsbucket -p myimage -o AKIAIOSFOD
NN7EXAMPLE -w wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
BUNDLE  bun-e3a4418a   i-31c2425a     myawsbucket myimage 2010-03-
19T08:22:48+0000 2010-03-19T08:22:48+0000        pending
```

   Amazon EC2 shuts down the instance, saves it as an AMI, and restarts it.
4. Enter the following command to register the image:

```
C:\> ec2-register myawsbucket/image.manifest.xml -n image_name
IMAGE ami-2bb65342
```

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

Using Query API

The following procedure steps you through how to bundle an Amazon EC2 instance store-backed AMI. This process mirrors the steps you would use with the command line tools.

To bundle Amazon EC2 instance store-backed AMIs

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

```
https://ec2.amazonaws.com/
?Action=BundleInstance
&InstanceId=-i-e468cd8d
&Storage.S3.AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Storage.S3.Bucket=myawsbucket
&Storage.S3.Prefix=winami
&Storage.S3.UploadPolicy=eyJleHBpcmF0aW9uIjogIjIwMDgtMDgtMzBUMDg6NDk6MDlaIi
wiY29uZGl0aW9ucyI6IFt7ImJ1Y2tldCI6ICJteS1idWNrZXQifSxbInN0YXJ0cy13aXRoIiw
gIiRrZXkiLCAibXktbmV3LWltYWdlIl0seyJhY2wiOiAiZWMyLWJ1bmRsZS1yZWEXAMPLEEXAMPLE
&Storage.S3.UploadPolicySignature=fh5tyyyQD8W4COEtxxxxxxxEXAMPLE
&AuthParams
```

Note

To perform this task, you need your AWS Access Key ID (<aws-access-key-id>) and AWS Secret Access Key (<aws-secret-access-key>). For more information, go to Amazon EC2 Credentials.

For information about the `BundleInstance` command, see BundleInstance in the *Amazon Elastic Compute Cloud API Reference*.

3. Construct the following command to register the image:

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

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

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 yor 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. 66).

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.

```
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:

```
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&nbsp;ID: </strong>[ami-id]<br />
<strong>AMI&nbsp;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 />
            </li>
    <li>[Description]</li>
    <li>This image contains the following daemons / services:
    <ul>

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

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

Identify Yourself

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

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

Using a Shared Windows AMI

This section describes how to find and safely use shared AMIs. One of the easiest ways to get started with Amazon EC2 is to use a shared AMI that has the components you need and add custom content.

Find Shared AMIs

To find shared AMIs

- Enter the `ec2-describe-images` command (or the abbreviated `ec2dim` command) with a flag to filter the results.

  The following examples show how to use a flag to filter the results.

  - The following command displays a list of all public AMIs. The *`-x all`* flag shows AMIs executable by all AWS accounts (i.e., AMIs with public launch permissions). This includes AMIs you own with public launch permissions.

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

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

    ```
    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 a Paid AMIs

This section describes how to launch paid AMIs and launch instances with a support product code.

After you purchase a paid AMI, you can launch instances of it. Launching a paid AMI is the same as launching any other AMI. No additional parameters are required. The instance will be charged according to the rates set by the owner of the AMI.

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.

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. 78) 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. 79) 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.

2. Copy the contents of the E2_CreateInstances.ps1 (p. 79) 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
EC2AccessKey	The access key associated with your AWS account.
EC2SecretKey	The secret access key associated with your AWS account.
matchDescription	The description of the AMI that will be used to create an instance.
RegionName	The region in which to search for the AMIs.
InstanceType	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 *Amazon Elastic Compute Cloud User Guide*.
InstancesToLaunch [optional]	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.
SecurityGroup [optional]	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.
KeyPairName	Enter the name of the key pair you created previously.
NameToTagCreatedInstance	A tag used by the AWS Management Console to help identify the instance. For more information about tags, see Using Tags in the *Amazon Elastic Compute Cloud User Guide*.

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)
```

```
$EC2Client=[Amazon.AWSClientFactory]::CreateAmazonEC2Client($AccountInfo.EC2Ac
cessKey,$AccountInfo.EC2SecretKey,$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_CreateInstances.ps1

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

```
<#
```

```
.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
```

```
$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)
```

```
 {
   ##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"})
```

```
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
}
```

Setting Up a Windows HPC Cluster on Amazon Elastic Compute Cloud

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

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. 37).
- 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. 91), 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. 92), 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" processorArchitec
ture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.mi
crosoft.com/WMIConfig/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.

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.local\hpcuser.

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
```

```
:: 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
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
```

Glossary

Amazon machine image (AMI)	An Amazon Machine Image (AMI) is an encrypted machine image stored in Amazon S3. It contains all the information necessary to boot instances of your software.
Amazon EBS	A type of storage that enables you to create volumes that can be mounted as devices by Amazon EC2 instances. Amazon EBS volumes behave like raw unformatted external block devices. They have user supplied device names and provide a block device interface. You can load a file system on top of Amazon EBS volumes, or use them just as you would use a block device.
Amazon EBS-backed AMI	An instance launched from an AMI backed by Amazon EBS uses an Amazon EBS volume as its root device. See Amazon EBS.
Instance store-backed AMI	An instance launched from an Amazon S3backed AMI uses an instance store as its root device. See instance store.
Availability Zone	A distinct location within a Region that is engineered to be insulated from failures in other Availability Zones and provides inexpensive, low latency network connectivity to other Availability Zones in the same Region.
compute unit	An Amazon-generated measure that enables you to evaluate the CPU capacity of different Amazon EC2 instance types.
EBS	See Amazon EBS.
Elastic Block Store	See Amazon EBS.
elastic IP address	A static public IP address designed for dynamic cloud computing. Elastic IP addresses are associated with your account, not specific instances. Any elastic IP addresses that you associate with your account remain associated with your account until you explicitly release them. Unlike traditional static IP addresses, however, elastic IP addresses allow you to mask instance or Availability Zone failures by rapidly remapping your public IP addresses to any instance in your account.
ephemeral store	See *instance store*.
explicit launch permission	Launch permission granted to a specific AWS account.
filter	Criterion you specify to limit the results when you list or describe your EC2 resources.

gibibyte (GiB)	A contraction of giga binary byte, a gibibyte is 2^30 bytes or 1,073,741,824 bytes. A gigabyte is 10^9 or 1,000,000,000 bytes.
group	See security group.
image	See *Amazon machine image*.
instance	Once an AMI has been launched, the resulting running system is referred to as an instance. All instances based on the same AMI start out identical and any information on them is lost when the instances are terminated or fail.
instance store	Every instance includes a fixed amount of storage space on which you can store data. This is not designed to be a permanent storage solution. If you need a permanent storage system, use Amazon EBS.
instance type	A specification that defines the memory, CPU, storage capacity, and hourly cost for an instance. Some instance types are designed for standard applications, whereas others are designed for CPU-intensive applications, or memory-intensive applications, etc.
launch permission	AMI attribute allowing AWS accounts to launch an AMI
Linux	Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.
maximum price	The maximum price you will pay to launch one or more Spot Instances. If your maximum price exceeds the Spot Price and your restrictions are met, Amazon EC2 launches instances on your behalf.
paid AMI	An AMI that you sell to other Amazon EC2 users. For more information, refer to the *Amazon DevPay Developer Guide*.
private IP address	All Amazon EC2 instances are assigned two IP addresses at launch: a private address (RFC 1918) and a public address that are directly mapped to each other through Network Address Translation (NAT).
public AMI	An AMI that all AWS accounts have launch permissions for.
public data sets	Sets of large public data sets that can be seamlessly integrated into AWS cloud-based applications. Amazon stores the data sets at no charge to the community and, like with all AWS services, you pay only for the compute and storage you use for their your applications. These data sets currently include data from the Human Genome Project, the U.S. Census, Wikipedia, and other sources.
public IP address	All Amazon EC2 instances are assigned two IP addresses at launch: a private address (RFC 1918) and a public address that are directly mapped to each other through Network Address Translation (NAT).
region	A geographical area in which you can launch instances (e.g., US, EU).
reservation	A collection of instances started as part of the same launch request.
Reserved Instance	An additional Amazon EC2 pricing option. With Reserved Instances, you can make a low one-time payment for each instance to reserve and receive a significant discount on the hourly usage charge for that instance.
resource	A general term that refers to the objects you work with in Amazon EC2. This includes instances, images, Amazon EBS volumes, snapshots, etc.

security group	A security group is a named collection of access rules. These access rules specify which ingress (i.e., incoming) network traffic should be delivered to your instance. All other ingress traffic will be discarded.
shared AMI	AMIs that developers build and make available for other AWS developers to use.
Solaris	Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.
snapshot	Amazon EBS provides the ability to create snapshots or backups of your Amazon EBS volumes and store them in Amazon S3. You can use these snapshots as the starting point for new Amazon EBS volumes and to protect your data for long term durability.
Spot Instance	A type of instance that you can bid on to take advantage of unused Amazon EC2 capacity.
Spot Price	The current price for Spot Instances. If your Spot Instance request exceeds this price and your restrictions are met, Amazon EC2 launches instances on your behalf.
supported AMIs	These AMIs are similar to paid AMIs, except that you charge for software or a service that customers use with their own AMIs.
tag	Metadata of your choice (consisting of up to 10 key-value pairs) that you can optionally assign to EC2 resources.
tebibyte (TiB)	A contraction of tera binary byte, a tebibyte is 2^40 bytes or 1,099,511,627,776 bytes. A terabyte is 10^12 or 1,000,000,000,000 bytes.
UNIX	Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.
Windows	Amazon EC2 instances are available for many operating platforms, including Linux, Solaris, Windows, and others.

Document History

This documentation is associated with the 2012-04-01 release of Amazon EC2. This guide was last updated on 23 May 2012.

The following table describes the new and updated content in the current release of the *Amazon EC2 Microsoft Windows Guide* documentation set.

Change	Description	Release Date
New content	This guide now includes Using Windows PowerShell in Amazon EC2 with the AWS SDK for .NET (p. 75) that explains how to configure Windows PowerShell to work with the .NET SDK and provides some code samples to help get you started.	In this release
Added content	This guide now includes Getting Started (p. 8); an overview of controlling access to your instances, Controlling Access: Security Groups and Credentials (p. 20); and a new section for creating and deploying a WordPress blog on your Amazon EC2 instance, Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 23).	17 December 2011
Updated content	The guide contains a new chapter, Setting Up a Windows HPC Cluster on Amazon Elastic Compute Cloud (p. 84) that explains how to configure a Windows HPC Cluster on Amazon Elastic Compute Cloud and updated instructions for Installing the Amazon EC2 Command Line Tools on Windows (p. 37).	14 November 2011
Created content	The guide provides information about using Amazon EC2 on Windows instances. For information on the basic infrastructure components of Amazon EC2 on Windows instances, see Using Amazon EC2 (p. 3). For information on using Windows AMIs, see Using Windows AMIs (p. 46). For information on setting up your command line interface, see Installing the Amazon EC2 Command Line Tools on Windows (p. 37).	23 September 2011

Index