Amazon Elastic Compute Cloud

Microsoft Windows Guide

API Version 2013-08-15

Amazon Web Services

Amazon Elastic Compute Cloud: Microsoft Windows Guide

Amazon Web Services

Copyright © 2013 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Elastic Compute Cloud Microsoft Windows Guide

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides scalable 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 interface. Use this guide to get started with Amazon EC2 with the Windows Server operating system.

How Do I...?

How Do I...?	Relevant Topics
Get a brief overview of Amazon EC2	What is Amazon EC2? (p. 3)
Get up and running right away with Amazon EC2	Getting Started with Amazon EC2 Windows Instances (p. 10)
Set up a WordPress blog on an Amazon EC2 instance	Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 21)
Learn the basic concepts for interacting with EC2	Amazon EC2 Infrastructure (p. 24)
Control access to my Amazon EC2 instances	Controlling Access to Amazon EC2 Resources (p. 30)
Get detailed information about using Windows AMIs	Windows Amazon Machine Images (AMI) (p. 35)
Use the AWS Management Pack for Microsoft System Center 2012 with Amazon EC2	AWS Management Pack for Microsoft System Center Operations Manager (p. 64)
Configure your Windows instance to recognize secondary private IP addresses	Configuring a Secondary Private IP Address for Your Windows Instance in a VPC (p. 92)
Set up an HPC Cluster using Amazon EC2	Setting Up a Windows HPC Cluster on Amazon EC2 (p. 96)

How Do I...?	Relevant Topics
Get started with the command line tools	Installing the Amazon EC2 Command Line Interface Tools on Windows (p. 106)
Run diagnostics on a Windows Server instance	AWS Diagnostics for Microsoft Windows Server (p. 112)

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 about using Amazon EC2	Amazon Elastic Compute Cloud User Guide
Find available libraries for programmatically accessing Amazon EC2	Available Libraries
Get started using the Amazon EC2 API	Making API Requests

What is Amazon EC2?

Amazon Elastic Compute Cloud (Amazon EC2) provides resizable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Topics

Features of Amazon EC2

Amazon EC2 provides the following features:

- Virtual computing environments, known as *instances*.
- Pre-configured templates for your instances, known as *Amazon Machine Images (AMIs)*, that package the bits you need for your server (including the operating system and additional software).
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as *instance types*.
- Secure login information for your instances using *key pairs* (AWS stores the public key, and you store the private key in a secure place).
- Storage volumes for temporary data that's deleted when you terminate your instance, known as *instance store volumes*.
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as *Amazon EBS volumes*.
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as *regions* and *Availability Zones*.

- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using *security groups*.
- Static IP addresses for dynamic cloud computing, known as *Elastic IP addresses*.
- Metadata, known as *tags*, that you can create and assign to your Amazon EC2 resources.
- Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as *virtual private clouds* (VPCs).

For more information about the features of Amazon EC2, see the Amazon EC2 product page.

How to Get Started with Amazon EC2 Windows Instances

The first thing you need to do is get up and running on an Amazon EC2 Windows instance. To do this, complete the Getting Started Tutorial for Amazon EC2 Windows Instances (p. 10). Whenever you need more information about a feature of Amazon EC2, you can find it in the technical documentation.

Get Up and Running

- Getting Started Tutorial for Amazon EC2 Windows Instances (p. 10)
- Set up a WordPress Blog on an Amazon EC2 Windows Instance (p. 21)

Basics

- Instances and AMIs
- Differences Between Windows Server and an Amazon EC2 Windows Instance (p. 5)
- Designing Your Applications to Run on Amazon EC2 Windows Instances (p. 7)
- How You're Charged for Amazon EC2 (p. 8)
- Tips and Tricks for Windows Users (p. 8)

Working with Amazon EC2 Windows Instances

- Controlling Access to Amazon EC2 Windows Instances (p. 30)
- The AWS Management Pack (p. 64)
- Configure a Secondary Private IP Address (p. 92)
- Set Up a Windows HPC Cluster (p. 96)
- Get Started with the Command Line Tools (p. 106)
- Run Diagnostics for an Amazon EC2 Windows Server Instance (p. 112)
- Upgrade PV Drivers (p. 117)

If you have questions about whether AWS is right for you, contact AWS Sales. If you have technical questions about Amazon EC2, use the Amazon EC2 forum.

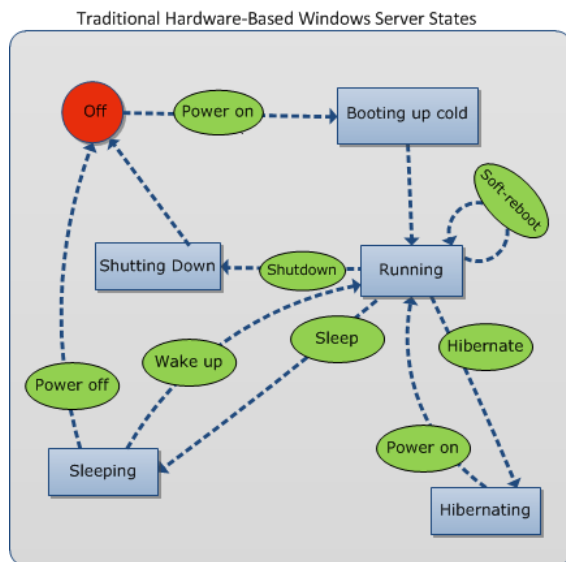# Differences Between Windows Server and an Amazon EC2 Windows Instance

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

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

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

After you launch your Amazon EC2 Windows instance, it behaves a lot like a traditional hardware-based server running Windows Server. For example, both 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 computing model. The way an Amazon EC2 instance runs is not the same as the way a traditional server running Windows Server runs.

A traditional server running 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 server running 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 on the server. However, after the server is powered off, the only way to get it up and running is by cold booting.

When your traditional 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 Windows Server diagram.
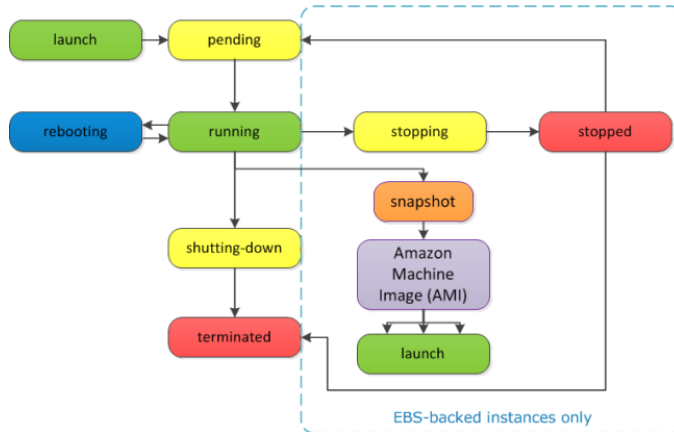
An Amazon EC2 Windows instance starts with the launch of the instance. Next, it briefly goes into the pending state while registration takes place. Then it moves to the running state, where instances can be rebooted, stopped, and then re-started. The Windows instance remains active until you initiate 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, allowing you to scale your infrastructure on demand.

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

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

The following table summarizes the differences between the life cycle of Windows Server and an Amazon EC2 Windows instance.

Windows Server	Amazon EC2 Windows Instance
Starts with push of the power button.	Starts with the launch of the instance.
When the server is up and running, you can keep it running until it is time to shut it down, or keep it in sleep or hibernation states (during which server is powered down).	After launch, it goes into pending state, then running state. While in running state, you can create an image of your instance (AMI) and launch additional instances.

Windows Server	Amazon EC2 Windows Instance
When shut down, all resources remain intact and in the state they were in when you switched it off. Information you stored on the hard drives persists and can be accessed whenever needed.	Active until you initiate shutdown process that terminates instance. After an Amazon EC2 Windows instance is terminated, its infrastructure is no longer available to you.

Designing Your Applications to Run on Amazon EC2 Windows Instances

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

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

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

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

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

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

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

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

Accessing Amazon EC2

Amazon EC2 provides a web-based user interface, the Amazon EC2 console. If you've signed up for an AWS account, you can access the Amazon EC2 console by signing into the AWS Management Console and selecting **EC2** from the console home page.

If you prefer to use a command line interface, there are several options:

Amazon EC2 Command Line Interface (CLI) Tools
Provide commands for Amazon EC2, Amazon EBS, and Amazon VPC, and is supported on Windows, Mac, and Linux/UNIX. To get started, see Installing the Amazon EC2 Command Line Interface Tools on Windows. For more information about the commands, see Commands (CLI Tools) in the *Amazon Elastic Compute Cloud Command Line Reference*.

AWS Command Line Interface (CLI)
Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux/UNIX. To get started, see AWS Command Line Interface User Guide. For more information about the commands for Amazon EC2, see ec2.

AWS Tools for Windows PowerShell
Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see AWS Tools for Windows PowerShell User Guide.

Amazon EC2 provides a Query API. These requests are HTTP or HTTPS requests that use the HTTP verbs GET or POST and a Query parameter named `Action`. For more information about the API actions for Amazon EC2, see Actions in the *Amazon Elastic Compute Cloud API Reference*.

If you prefer to build applications using language-specific APIs instead of submitting a request over HTTP or HTTPS, AWS provides libraries, sample code, tutorials, and other resources for software developers. These libraries provide basic functions that automatically take care of tasks such as cryptographically signing your requests, retrying requests, and handling error responses, so that it is easier for you to get started. For more information, see AWS SDKs and Tools.

How You're Charged for Amazon EC2

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

- Instance usage

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

- Data transfer
- Storage

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

To see your bill, go to AWS Account Activity page.

Tips and Tricks for Windows Users

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

- For the best experience using Internet Explorer, run the latest version.
- If you open an RDP session and are prompted for a domain (for example, 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 Amazon EC2 console: right-click the instance, and then click **Connect**.

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

- Don't launch an instance without a key pair. Without the key pair, you'll be unable to connect to your instance.

- After you launch and connect to an instance, do two things:

  1. Log into the instance and change your administrator password.

  2. While still logged in, create another user account with administrator permissions. This account can be useful if you forget the original administrator password account or if you have a problem using the original administrator account.

Getting Started with Amazon EC2 Windows Instances

This tutorial will help you get started using Amazon Elastic Compute Cloud (Amazon EC2) Windows instances. You'll primarily use the AWS Management Console, a point-and-click web-based interface. You can also watch this short video to get started: Getting Started with Amazon EC2: Launching a Windows Instance.

To complete this Amazon EC2 tutorial

1. Sign Up for EC2 (p. 10)
2. Launch a Windows Instance (p. 11)
3. Connecting to Your Windows Instance (p. 14)
4. (Optional) Create an Elastic IP Address (p. 14)
5. (Optional) Create a CloudWatch Alarm to Monitor Your Instance (p. 15)
6. Clean Up (p. 19)

After you've completed the tutorial, you'll need to clean up.

Sign Up for EC2

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including Amazon EC2. You are charged only for the services that you use.

With Amazon EC2, you pay only for what you use. If you are a new AWS customer, you can get started with Amazon EC2 for free. For more information, see AWS Free Usage Tier.

To create an AWS account

1. Go to http://aws.amazon.com, and then 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.

Launch a Windows Instance

You can launch a Windows instance using the AWS Management Console as described following. An instance is a virtual server in the AWS cloud. Amazon EC2 enables you to set up and configure the operating system and applications that run on your instance.

Important
When you sign up for AWS, you can get started with Amazon EC2 for free using the AWS Free Usage Tier. If you created your AWS account less than 12 months ago, and have not already exceeded the Free Usage Tier benefits for Amazon EC2 and Amazon EBS, it will not cost you anything to complete this tutorial, because we help you select options that are within the Free Usage Tier benefits. Otherwise, you'll incur the standard Amazon EC2 usage fees from the time that you launch the instance until you terminate the instance (which is the final task of this tutorial), even if it remains idle. The total charges to complete this tutorial outside the Free Usage Tier are minimal.

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/.

2. From the navigation bar, select the region for the instance. For this tutorial, you can use the default region. Otherwise, this choice is important because some Amazon EC2 resources can be shared between regions, while others can't. For example, if you'd like to connect your instance to an existing Amazon EBS volume, you must select the same region as the volume.
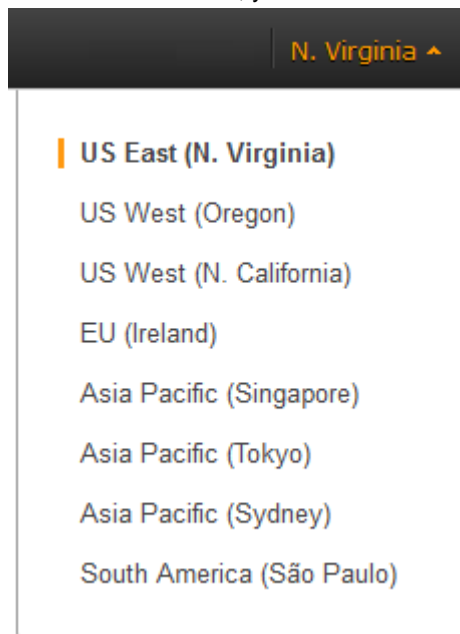
3. From the console dashboard, click **Launch Instance**.

4. On the **Create a New Instance** page, click **Quick Launch Wizard**. (This wizard automatically selects many configuration settings for you, so that you can get started quickly.)

5. (Optional) In **Name Your Instance**, enter a name for the instance that has meaning for you. (If you run multiple instances, naming them helps you identify them in the console.)

6. Under **Choose a Key Pair**, choose from any existing key pairs that you have created or create a new key pair. For this example, we'll create a key pair.

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

   a. Click **Create New**.

   b. Type a name for your key pair and then click **Download**. This is the only chance for you to save the private key file, so be sure to click **Download**.

   c. Save your private key in a safe place on your computer. You'll need to provide your private key each time you connect to your instance.

7. Under **Choose a Launch Configuration**, the Quick Launch Wizard displays a list of basic Amazon Machine Images (AMIs) that serve as templates for your instance. An AMI contains everything we need to create a new instance of a server, for example, a web server or a database server. For this tutorial, select the 64-bit version of Microsoft Windows Server 2008 R2. Because the configuration is marked with a star, it's within the Free Usage Tier.

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

9. In **Security Details**, under **Security Group**, you see the security group that is selected for you by the wizard.

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

If you're new to Amazon EC2 and haven't set up any security groups yet, AWS defines one for you. You can change the name and description by clicking **Edit details**. The security group includes basic firewall rules that enable you to connect your instance. For a Windows instance, you connect through Remote Desktop Protocol (RDP) on port 3389.

If you have used Amazon EC2 before, the wizard looks for an existing security group for a Windows instance.

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

10. Review your settings, confirming that you are launching a `t1.micro` instance using a Windows Server 2008 AMI, and then click **Launch** to launch the instance.

11. A confirmation page lets you know that your instance is launching. Click **Close** to close the confirmation page and return to the console.

12. Click **Instances** in the navigation pane to view the status of the launch. It takes a short time for an instance to launch. When you launch an instance, its initial state is `pending`. After the instance starts, its state changes to `running` and it receives a public DNS name. (If the **Public DNS** column is hidden, click the **Show/Hide** icon in the top right corner of the Instances page and select **Public DNS**.)

13. Record the public DNS name for your instance because you'll need it for the next step.

14. (Optional) After your instance is launched, you can view its security group rules. From the Instances page, select the instance. In the **Description** tab, find **Security Groups** and click **view rules**.

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

Connecting to Your Windows Instance

To connect to a Windows instance, you must retrieve the initial administrator password, and then specify this password when you connect to your instance using Remote Desktop.

> **Note**
> Windows instances are limited to two simultaneous remote connections at one time. If you attempt a third connection, an error will occur. For more information, see Configure the Number of Simultaneous Remote Connections Allowed for a Connection.

To connect to your Windows instance

1. In the EC2 console, select the instance, click **Actions**, and then click **Connect**.
2. In the **Console Connect** dialog box, click **Retrieve Password** (it will take a few minutes after the instance is launched before the password is available).
3. Click **Browse** and navigate to the private key file you created when you launched the instance. Select the file and click **OK** to copy the entire contents of the file into the **Private Key contents** box.
4. Click **Decrypt Password**. The console displays the default administrator password for the instance in the **Console Connect** dialog box, replacing the link to **Retrieve Password** shown previously with the actual password.
5. Record the default administrator password, or copy it to the clipboard. You need this password to connect to the instance.
6. Click **Download shortcut file**. Your browser prompts you to either open or save the .rdp file. Either option is fine. When you have finished, you can click **Close** to dismiss the **Console Connect** dialog box.
7. If you opened the .rdp file, you'll see the **Remote Desktop Connection** dialog box. If you saved the .rdp file, navigate to your downloads directory, and double-click the .rdp file to display the dialog box. You may get a warning that the publisher of the remote connection is unknown. Click **Connect** to connect to your instance. You may get a warning that the security certificate could not be authenticated. Click **Yes** to continue.
8. Log in to the instance as prompted, using `Administrator` as the user name and the default administrator password that you recorded or copied in step 7.

We recommend that you do the following:

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

Create an Elastic IP Address

By default, all Amazon EC2 instances launched into EC2-Classic or a default VPC 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). Instances launched into a nondefault VPC do not get a public IP address by default, but can be assigned one at launch.

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 stops and restarts. If you want a persistent address to connect to, use an Elastic IP address.

Elastic IP addresses are static IP addresses designed for dynamic cloud computing. Additionally, Elastic IP addresses are associated with your account, not specific instances. Any Elastic IP addresses that you associate with your account remain associated with your account until you explicitly release them. Unlike traditional static IP addresses, however, Elastic IP addresses allow you to mask instance or Availability Zone failures by rapidly remapping your public IP addresses to any instance in your account.

To assign an Elastic IP address to your Windows instance

1. Click **Elastic IPs** in the navigation pane.
2. Click **Allocate New Address**.
3. In the **Allocate New Address** dialog box, click **Yes, Allocate**.
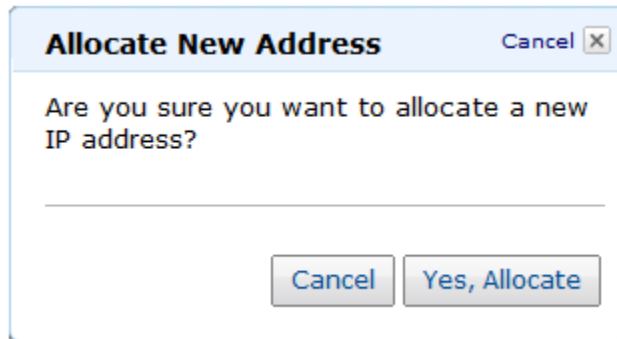
4. Select the Elastic IP address you created, and then click **Associate Address**.
5. In the **Associate Address** dialog box, in the **Instance** list, select your instance and then click **Yes, Associate**.

Create a CloudWatch Alarm to Monitor Your Instance

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

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

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

To open the Create Alarm Wizard

1. Open the Amazon CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. Click **Alarms** in the navigation pane.
3. On the **CloudWatch Alarms** page, click **Create Alarm**.
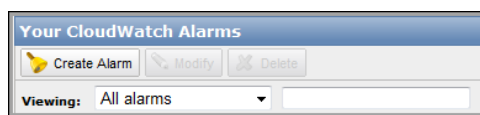
4.   The **SELECT METRIC** page of the **Create Alarm Wizard** opens.
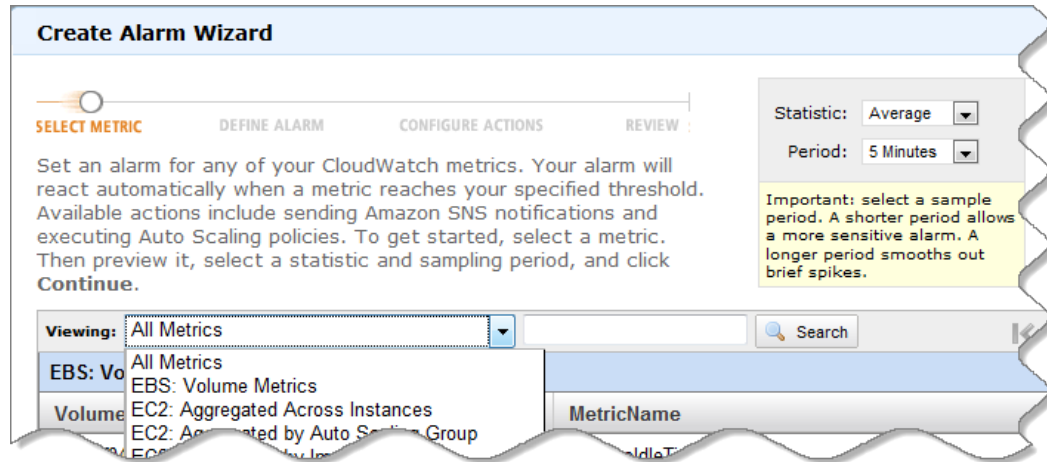
To select a metric for your alarm

1.   In the **SELECT METRIC** page of the **Create Alarm Wizard**, select **EC2: Instance Metrics** from the **Viewing** 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** list.

4.   Select a period from the **Period** list, for example: `5 minutes`.

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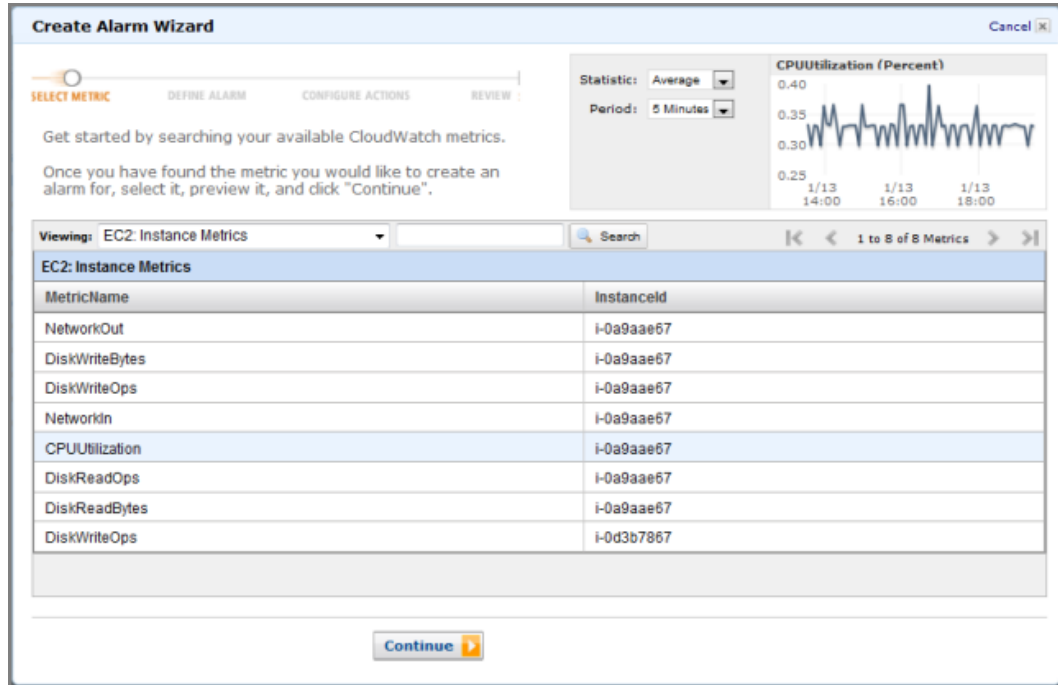
6. The **DEFINE ALARM** page of the **Create Alarm Wizard** opens.

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** 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**.

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

To configure an email action for an alarm

1.   On the **CONFIGURE ACTIONS** page, select **ALARM** from the **Alarm State** list.

2.   Select **Create Email Topic** from the **Topic** list.

       Two new fields named **Topic** and **Emails** replace the **Topic** 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.

Clean Up

Now that you've completed this tutorial, you can clean up the resources that you created. You could also customize your instance to your needs and keep using it.

> **Important**
> Remember, unless you are within the Free Usage Tier, 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).

When you've decided that you no longer need the instance, you need to clean up these resources:

*   The Amazon CloudWatch alarm
*   The Elastic IP address
*   The instance

To delete your CloudWatch alarm

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

If you created an Elastic IP address, you must disassociate it from the instance and release it.

Important

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

To disassociate and release your Elastic IP address

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Click **Elastic IPs** in the navigation pane.
3. Select your Elastic IP address, and then click **Disassociate**.
4. Click **Yes, Disassociate** when prompted.
5. Select your Elastic IP address again, and then click **Release**.
6. Click **Yes, Release** when prompted.

Terminating an instance effectively deletes it; you can't reconnect to an instance after you've terminated it.

If you launched an instance that is not within the Free Usage Tier, you'll stop incurring charges for that instance as soon as the instance status changes to `shutting down` or `terminated`.

To terminate your instance

1. Locate the instance you want to terminate in your list of instances on the **Instances** page.
2. Right-click the instance, and then click **Terminate**.
3. Click **Yes, Terminate** when prompted for confirmation.

Deploying a WordPress Blog on Your Amazon EC2 Instance

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

Topics

Prerequisites

Before you get started, be sure that you do the following:

- Launch an Amazon EC2 instance from an AMI that has Microsoft Windows Server 2008 R2 and Internet Information Services (IIS) pre-installed. For information about launching an EC2 instance, see Getting Started with Amazon EC2 Windows Instances (p. 10).
- Use the AWS free usage tier (if eligible) to launch and use the free EC2 Windows *t1.micro* instance for 12 months. You can use the AWS free usage tier for launching new applications, testing existing applications, or simply gaining hands-on experience with AWS. For more information about eligibility and the highlights, see the AWS Free Usage Tier product page.

  Important
  If you've launched a regular instance and use it to deploy the WordPress website, you will incur the standard Amazon EC2 usage fees for the instance until you terminate it. For more information about Amazon EC2 usage rates, go to the Amazon EC2 product page.

- Ensure that the security group in which you're launching your EC2 instance has ports 80 (HTTP) and 3389 (RDP) open for inbound traffic. Port 80 allows computers outside of the instance to connect with HTTP. If port 80 is not open, the WordPress site can't be accessed from outside the instance. Port 3389 allows you to connect to the instance with Remote Desktop Protocol.
- Connect to your EC2 instance.

Installing the Microsoft Web Deployment Tool

This procedure uses the Microsoft IIS Web Deployment Tool to install and configure WordPress on your server. The Web Deployment Tool simplifies deployment of Web applications and Web sites to IIS servers. For more information, see http://www.iis.net/downloads/microsoft/web-deploy.

1. Verify that you've met the conditions in Prerequisites (p. 21).
2. Disable Internet Explorer Enhanced Security Configuration.

   a. In your EC2 instance, click **Start**, point to **Administrative Tools**, and then click **Server Manager**.
   b. In the **Security Information** pane, click **Configure IE ESC**.
   c. Under **Administrators**, click **Off** and click **OK**.
   d. Close the **Server Manager** window.

3. In the EC2 instance, open **Internet Explorer** and go to
   `http://www.iis.net/download/webdeploy`.
4. Download and install the latest version of Web Deploy.

Installing WordPress

Now that the Web Deployment Tool is installed, you can use it install and configure WordPress on your server.

To install WordPress

1. Open the **Web Platform Installer** and click **Applications**.
2. Select **WordPress**, click **Add**, and then click **Install**.
3. On the **Prerequisites** page, select **MySQL** for the database to use. Enter the desired administrator password for your MySQL database in the **Password** and **Confirm Password** boxes and click **Continue**.
4. Click **I Accept** for the list of third-party application software, Microsoft products, and components. After the Web Platform Installer finishes installing the software, you are prompted to configure your new site.
5. Clear the default application name in the **'WordPress' application name:** box and leave it blank, then leave the default information in the other boxes and click **Continue**
6. Click **Yes** to accept that the contents of the folder will be overwritten, and finish the wizard.
7. On the WordPress **Welcome** page, enter the following information and click **Install WordPress**.

   - **Site Title**—Your site title.
   - **Username**—Leave set to `admin`.
   - **Password, twice**—The password for your site. Re-enter the same password in the second box.
   - **Your E-mail**—Your email address.
   - **Privacy**—Check to allow search engines to index your site.

8. Click **Log In**.
9. On the **Log In** page, enter `admin` for **Username** and the site password you entered previously for **Password**.

Creating Your First Blog Post

Now you can create your first blog post on your new WordPress site.

To create your first blog post

1. Open the WordPress dashboard by going to `http://localhost/wp-admin`. If prompted for your credentials, enter `admin` for the **Username** and your site password for **Password**.
2. In the **QuickPress** box, enter the following information:

    -   **Title**—`My First Post`
    -   **Content**—`This is my first post`

3. Click **Publish** to publish your blog to your localhost. A notification appears in which you can choose to view or edit the post.
4. Click **View post** to see your post.

Making Your WordPress Site Public

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

To configure the default settings for your WordPress site

1. Open the WordPress dashboard by going to `http://localhost/wp-admin`. If prompted for your credentials, enter `admin` for the **Username** and your site password for **Password**.
2. In the **Dashboard** pane, click **Settings**.
3. On the **General Settings** page, enter the following information and click **Save Changes**.

    -   **WordPress address (URL)**—The public DNS address of your EC2 instance. For example, your URL may look something like `http://ec2-203-0-113-25.compute-1.amazonaws.com`.
    -   **Site address (URL)**—The same public DNS address of your EC2 instance that you set in **WordPress address (URL)**.

4. To see your new site, open a browser on a computer other than the EC2 instance hosting WordPress and type the public DNS address of your EC2 instance in the web address field. Your WordPress site appears.

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

Amazon EC2 Infrastructure

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

Topics

Amazon Machine Images and Instances

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

Amazon publishes many AMIs that contain common software configurations for public use. In addition, members of the AWS developer community have published their own custom AMIs. You can also create your own custom AMI or AMIs; doing so enables you to quickly and easily start new instances that have everything you need. For example, if your application is a web site or web service, your AMI could include a web server, the associated static content, and the code for the dynamic pages. As a result, after you launch an instance from this AMI, your web server starts, and your application is ready to accept requests.

You can launch different types of instances from a single AMI. An *instance type* essentially determines the hardware of the host computer used for your instance. Each instance type offers different compute and memory facilities. Select an instance type based on the amount of memory and computing power that you need for the applications or software that you plan to run on the instance. For more information, see Available Instance Types. You can launch multiple instances from an AMI, as shown in the following figure.
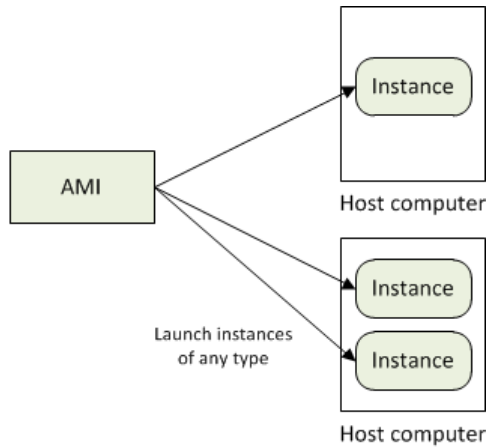
Your Windows instances keep running until you stop or terminate them, or until they fail. If an instance fails, you can launch a new one from the AMI.

For more information about Windows AMIs and instances, see Windows Amazon Machine Images (AMI) (p. 35) and Windows Instance Types.

Regions and Availability Zones

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

Each region contains multiple distinct locations called *Availability Zones*. Each Availability Zone is engineered to be isolated from failures in other Availability Zones, and to provide inexpensive, low-latency network connectivity to other zones in the same region. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.
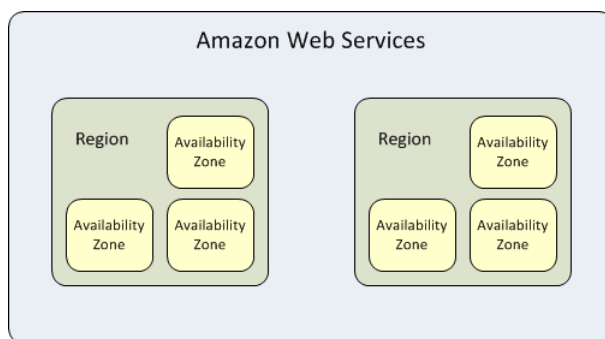
For more information about the available regions and Availability Zones, see Using Regions and Availability Zones.

Storage

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

- Amazon Elastic Block Store (Amazon EBS)
- Amazon EC2 Instance Store
- Amazon Simple Storage Service (Amazon S3)

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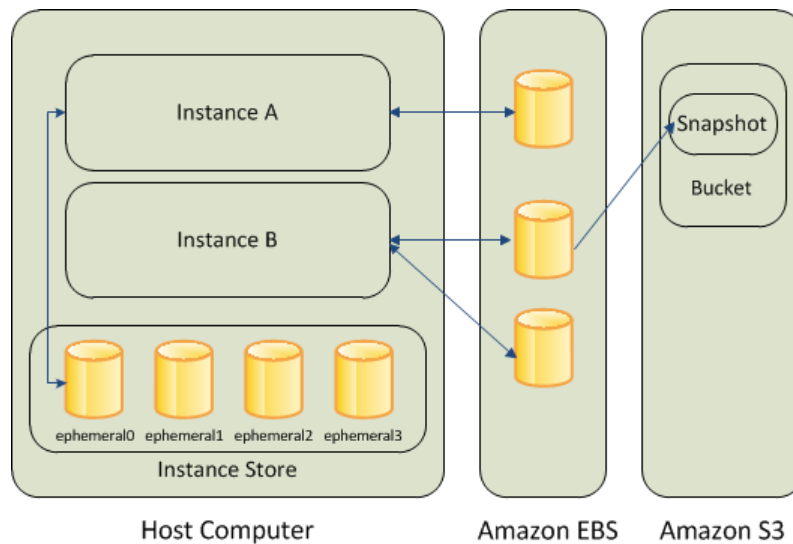
Amazon EBS Volumes

Amazon EBS volumes are the recommended storage option for the majority of use cases. Amazon EBS provides your instances with persistent, block-level storage. Amazon EBS volumes are essentially hard disks that you can attach to a running instance.

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

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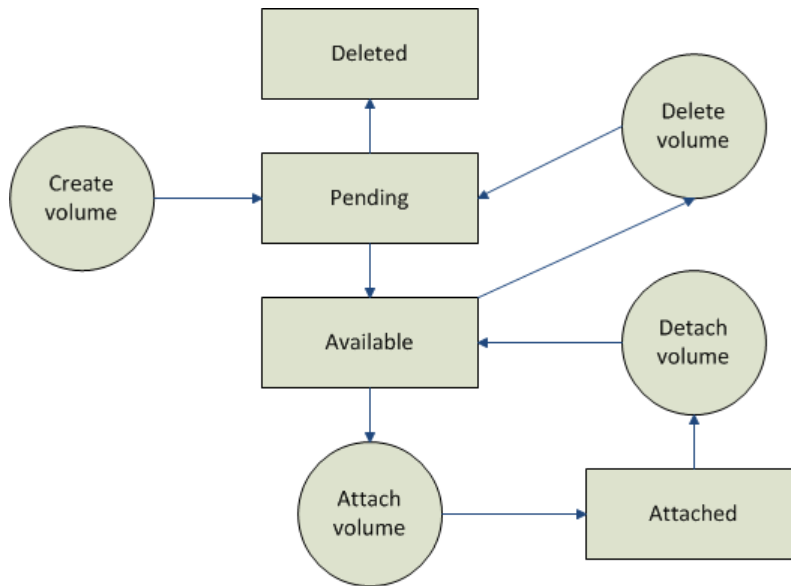
For more information about Amazon EBS volumes, see Amazon Elastic Block Store.

Instance Store

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

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

Amazon S3

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

Root Device Storage

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

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

The description of an AMI indicates the type of root device (either `ebs` or `instance store`). This is important because there are significant differences in what you can do with each type of AMI. For more information about these differences, see Root Device Storage on Windows AMIs (p. 36).

Networking and Security

You can launch instances in one of two platforms: EC2-Classic and EC2-VPC. An instance that's launched into EC2-Classic is assigned a public IP address. An instance that's launched into EC2-VPC is assigned public IP address only if it's launched into a default VPC. An instance that's launched into a nondefault VPC must be specifically assigned a public IP address at launch. For more information about EC2-Classic and EC2-VPC, see Supported Platforms in the *Amazon Elastic Compute Cloud User Guide*.

Instances can fail or terminate for reasons outside of your control. If one fails and you launch a replacement instance, the replacement has a different public IP address than the original. However, if your application needs a static IP address, Amazon EC2 offers *Elastic IP addresses*. For more information, see Using Instance IP Addresses in the *Amazon Elastic Compute Cloud User Guide*.

You can use *security groups* to control who can access your instances. These are analogous to an inbound network firewall that enables you to specify the protocols, ports, and source IP ranges that are allowed to reach your instances. You can create multiple security groups and assign different rules to each group. You can then assign each instance to one or more security groups, and we use the rules to determine which traffic is allowed to reach the instance. You can configure a security group so that only specific IP addresses or specific security groups have access to the instance. For more information about security groups, see Amazon EC2 Security Groups (p. 31).

Monitoring, Auto Scaling, and Load Balancing

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

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

AWS Identity and Access Management

Amazon EC2 integrates with AWS Identity and Access Management (IAM), a service that enables you to do the following:

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

With Amazon EC2, you can use IAM to control which users in your AWS account can create AMIs or launch instances.

For more information about IAM, see the following:

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

Available EC2 Interfaces

AWS provides different interfaces to access EC2.

AWS Management Console

The AWS Management Console is a simple web-based GUI. To get started using the console, see Getting Started with Amazon EC2 Windows Instances (p. 10).

Command Line Tools (API Tools)

EC2 provides a Java-based command-line client that wraps the EC2 API. For more information, see Installing the Amazon EC2 Command Line Interface Tools on Windows (p. 106) and Amazon Elastic Compute Cloud Command Line Reference.

Programmatic Interface

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

Resource	Description
AWS SDKs	AWS SDKs include sample code, libraries, tools, documentation, and templates. To download the AWS SDKs, go to AWS Software Development Kits (SDKs).
Libraries	Developers can provide their own libraries, which you can find at the following AWS developer centers: • Java Developer Center • Mobile Developer Center • PHP Developer Center • Python Developer Center • Ruby Developer Center • Windows and .NET Developer Center
Amazon EC2 API	If you prefer, you can code directly to the Amazon EC2 API. For more information, see Making API Requests and Amazon Elastic Compute Cloud API Reference.

Controlling Access to Amazon EC2 Resources

Amazon EC2 provides features that enable you to access resources and other services in AWS, and use the AWS Management Console, command line interface (CLI) tools, and APIs.

Topics

- Security Credentials (p. 30)
- AWS Identity and Access Management (IAM) (p. 31)
- Amazon EC2 Security Groups (p. 31)
- Passwords for a Windows Server Instance (p. 32)

Security Credentials

If you want to...	Use this...
Connect to an instance	Key pair (used to decrypt the Administrator password)
Use the Amazon EC2 console	Email address and password
Use the Amazon EC2 CLI	Access keys
Use the Amazon EC2 API	Access keys
Share an AMI or EBS snapshot	AWS account ID (without the hyphens)
Bundle a Windows AMI and upload it to Amazon S3	Access keys
Allow your instance to use other services, such as Amazon S3	Access keys (located on the instance itself)

For more information, see AWS Security Credentials.

AWS Identity and Access Management (IAM)

You can use features of IAM to allow other users, services, and applications to use your Amazon EC2 resources without sharing the security credentials for your AWS account. You can choose to allow full use or limited use of your Amazon EC2 resources.

For more information, see Controlling Access in the *Amazon Elastic Compute Cloud User Guide*.

Amazon EC2 Security Groups

A *security group* acts as a virtual firewall that controls the traffic allowed to reach one or more instances. When you launch an instance, you can assign it one or more security groups. You add rules to each security group that control traffic for the instance. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances to which the security group is assigned.

For more information about security groups, see Amazon EC2 Security Groups in the *Amazon Elastic Compute Cloud User Guide*.

Restricting Access to an IP Address Range

When you create a security group rule, the default source is `0.0.0.0/0`. This 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. You can get the public IP address of your local computer using a service. To locate a service that provides your IP address, use the search phrase "what is my IP address". If you are connecting through an ISP or from behind your firewall without a static IP address, you need to find out the range of IP addresses used by client computers.

Restricting Access to a Specific Security Group

When you create a security group rule, you can specify a security group as the source. For example, suppose that your application uses two 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 instance running your web server can connect to your database instance on this port.

For our database server, suppose 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 `203.0.113.19/32`. Because this CIDR block uses the entire IPv4 address range, it allows in only a single host.

Passwords for a Windows Server Instance

When you connect to a Windows instance, you must specify the name of a user account with permissions to access the instance, and the password for the account. The first time that you connect to your instance, you specify the Administrator account and the default administrator password. We recommend that you change the Administrator password from its default value, and create another user account with administrative privileges on the instance.

If you've lost the password for the Administrator account for your Windows Server instance, or if the password has expired, you can reset the password using the Amazon EC2 configuration service.

> **Important**
> If you have disabled the local Administrator account, you cannot reset the password using this method.

In this section, the instance whose password you need to reset will be referred to as the *reset instance*.

Prerequisites for Resetting a Password

You need the following prerequisites to reset the password for a Windows Server instance using the Amazon EC2 configuration service.

• The Amazon EC2 configuration service is installed on the instance whose password is to be reset. This service is available by default on all Amazon Windows AMIs, or you can download it. For more information, see Installing the Latest Version of EC2Config (p. 49).
• A running Windows Server 2003 instance that you can log into, in the same Availability Zone as the instance that needs the password reset. In the following instructions, this instance will be referred to as the *recovery instance*. Note that we recommend Windows Server 2003 instances because they use an older boot loader that does not try to modify the boot files.

Resetting the Administrator Password on a Windows Server Instance

The Amazon EC2 configuration service can reset the administrator password for you if you modify a configuration file on the boot volume of the reset instance. However, this file can only be modified on a volume that is not currently the root volume, so must first detach the root volume from the reset instance, attach the volume to a recovery instance, change the configuration settings, and reattach the root volume to the reset instance.

To reset the administrator password

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

2. **Stop the reset instance**

    a. In the navigation pane, click **Instances**.

    b. Right-click the reset instance and click **Stop**.

    c. In the **Stop Instances** dialog box, click **Yes, Stop**. After the instance has stopped, proceed with
       the next step.

3. **Detach the root volume**

    a. In the navigation pane, click **Volumes**.

    b. In the list of volumes, right-click the root volume of the reset instance, and then click **Detach
       Volume**. After the volume's status changes to **available**, proceed with the next step.

4. **Attach the volume to the recovery instance**

    a. Right-click the volume and click **Attach Volume**.

    b. In the **Attach Volume** dialog box, in the **Instances** list, select your recovery instance.

    c. In the **Device** box, type `xvdf` (if it isn't already there), and then click **Yes, Attach**.

    d. Log in to the recovery instance and set the volume as online. For more information, see Make
       the Volume Available on Windows.

5. **Modify the configuration file for the reset volume**

    a. On the recovery instance, open the `\Program
       Files\Amazon\Ec2ConfigService\Settings\config.xml` file from the volume using a
       text editor, such as Notepad.

    b. At the top of the file, under **<Plugin>**, **<Name>Ec2SetPassword</Name>**, change
       <State>Disabled</State> to **<State>Enabled</State>**, and then save the file.

6. **Detach the volume from the recovery instance**

   a.   In the recovery instance, set the volume to offline.

   b.   In the navigation pane, click **Volumes**.

   c.   In the list of volumes, right-click the volume and click **Detach Volume**. After the volume's status changes to **available**, proceed with the next step.

7. **Reattach the volume to the reset instance**

   a.   Right-click the volume and click **Attach Volume**.

   b.   In the **Attach Volume** dialog box, in the **Instances** drop-down list, select the volume.

   c.   In the **Device** box, type `/dev/sda1`, and then click **Yes, Attach**.

8. **Restart the reset instance**

   a.   In the navigation pane, click **Instances**.

   b.   Right-click the reset instance and click **Start**.

   c.   In the **Start Instances** dialog box, click **Yes, Start**.

> **Important**
> The instance will get a new IP address and DNS name.

   d.   Update your Remote Desktop Protocol connection with the new DNS name.

9. **Retrieve the new default password**

   a.   In the navigation pane, click **Instances**.

   b.   Right-click the reset instance and click **Get Windows Password**.

   c.   In the **Retrieve Default Windows Administrator Password** dialog box, click **Browse**, and then select the appropriate private key (`.pem`) file.

   d.   Click **Decrypt Password** and use the decrypted password to log in to the reset instance as administrator.

Windows Amazon Machine Images (AMI)

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

For more information about AWS Windows AMIs, see Amazon Windows AMI Basics (p. 35).

You can use the AWS Management Console to search for Windows AMIs that meet your specific criteria. For example, you can view the Windows AMIs provided by AWS, or the Windows AMIs provided by the EC2 community. For more information about choosing a Windows AMI, see Choosing a Windows AMI (p. 38).

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. 50).

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 Amazon 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 about how to create and use shared AMIs, see Shared Windows AMIs (p. 55).

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 Amazon EC2 users. For more information about selling or using paid AMIs, see Paid Windows AMIs (p. 60).

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

Amazon Windows AMI Basics

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

using that AMI. You connect to the instance using Remote Desktop Connection, just as you would with any other Windows server.

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

- Microsoft Windows Server 2012 (64-bit)
- Microsoft Windows Server 2008 R2 (64-bit)
- Microsoft Windows Server 2008 (64-bit)
- Microsoft Windows Server 2008 (32-bit)
- Microsoft Windows Server 2003 (64-bit)
- Microsoft Windows Server 2003 (32-bit)

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

We update the AWS Windows AMIs several times a year. When we update an AWS AMI, we deprecate the previous AMI and replace it with a new AMI and AMI ID. To find an AMI after it's been updated, use the name instead of the ID. The basic structure of the AMI name is usually the same, with a new date added to the end. You can use a query or script to search for an AMI by name, confirm that you've found the correct AMI, and then launch your instance.

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 AWS or other reputable sources provide.

For a list of AWS-approved Microsoft Windows AMIs, go to Amazon Machine Images (AMIs) and select Windows as the platform. Click any AMI in the resulting list for more information about the AMI.

Root Device Storage on Windows AMIs

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

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

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

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

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

Characteristic	Amazon EBS-Backed	Amazon Instance Store-Backed
Boot time	Usually less than 1 minute	Usually less than 5 minutes
Size limit	1 TiB	10 GiB
Root device	Amazon EBS volume	Instance store volume
Data persistence	Persists on instance failure and can persist on instance termination	Persists for the life of the instance
Upgrading	The instance type, kernel, RAM disk, and user data can be changed while the instance is stopped.	Instance attributes are fixed for the life of an instance
Charges	Instance usage, Amazon EBS volume usage, and Amazon EBS snapshot (AMI storage)	Instance usage and Amazon S3 (AMI storage)
Stopped state	Can be placed in the stopped state (the instance is not running, but is persisted in Amazon EBS)	Cannot be placed in the stopped state

Configuration of an AWS Windows AMI

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

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

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

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

Xen Drivers

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

Citrix drivers are stored in the `%ProgramFiles%\Citrix` (32-bit instances) and `%ProgramFiles(x86)%\Citrix` (64-bit instances) directories.

Citrix has a few more driver components, which are located in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services`. They are

- xenevtchn
- xeniface
- xennet
- xennet6
- xensvc
- xenvbd
- xenvif

Citrix also has a driver component named XenGuestAgent, which runs as a Windows service. It handles tasks such as time synchronization at boot, and shutdown and restart events from the API. You can access and manage services by typing `services.msc` at the command line.

For more information about upgrading your RedHat drivers on an existing AMI to Citrix drivers, see Upgrading Your PV Drivers on Your Windows AMI (p. 117).

Keeping Your Instances Updated

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

> **Note**
> You can reboot an Amazon EC2 Windows instance after the updates take place. Rebooting works the same way for both instance store-backed instances and Amazon EBS-backed instances. For more information, see Root Device Storage on Windows AMIs (p. 36).

Support

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

You're encouraged to post any questions you have about using AWS Windows AMIs to the Amazon EC2 forum.

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

Choosing a Windows AMI

Amazon Machine Images (AMIs) are the basic building block of Amazon EC2. Before you accomplish anything with Amazon EC2, you must first choose an AMI. The AMI can be provided by AWS or the

Amazon EC2 community, or you can create your own AMIs. To create your own AMI, you must start by using one of the base AMIs provided.

After finding and selecting an AMI, record its AMI ID. You'll use the AMI ID when you launch your instance and then connect to it. For information about launching your instance, see Launch a Windows Instance (p. 11). For information about connecting to your Windows instance, see Connecting to Your Windows Instance (p. 14).

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 navigation pane, click **AMIs**.
3. [Optional] Use the **Filter** options to manipulate the list of displayed AMIs. For example, to see a list of all Windows AMIs provided by Amazon, select **Public Images**, **Amazon Images**, and then **Windows** from the **Filter** lists.
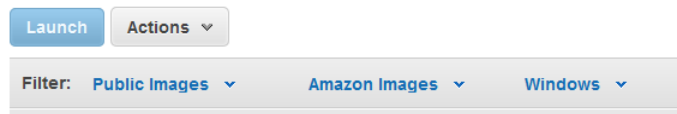
4. Click **Go to Details Page** (the magnifying glass) for an AMI to view its properties in a new screen.

   As you are selecting an AMI, it's important to note whether the AMI is backed by instance store or by Amazon EBS. Select the type of AMI that meets your needs. For more information, see Root Device Storage on Windows AMIs (p. 36).

Using Command Line Tools

Amazon EC2 provides a Java-based command-line client that wraps the Amazon EC2 Query API. You must install the command line tools before you can try the example commands in this section. For information about installing the command line tools, see Installing the Amazon EC2 Command Line Interface Tools on Windows (p. 106).

To find a suitable AMI

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

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

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

IMAGE ami-c941efa0 amazon/Windows_Server-2008-SP2-English-64Bit-Base-
2013.05.15   amazon   available   public   x86_64
machine   windows   ebs   hvm xen
BLOCKDEVICEMAPPING   EBS /dev/sda1   snap-b81a74c9   30   true   standard
IMAGE ami-2b41ef42 amazon/Windows_Server-2008-R2_SP1-English-64Bit-Base-
2013.05.15   amazon   available   public   x86_64
machine   windows   ebs   hvm xen
BLOCKDEVICEMAPPING   EBS /dev/sda1   snap-f00e6081   30   true   standard
IMAGE ami-b340eeda amazon/Windows_Server-2008-R2_SP1-English-64Bit-
```

```
SQL_2008_R2_SP1_Express-2012.07.11   amazon   available   public   x86_64
machine   windows   ebs   hvm   xen
BLOCKDEVICEMAPPING   EBS /dev/sda1   snap-0e2d437f   30   true   standard
IMAGE   ami-a8e705c1   ec2-paid-ibm-images/ibm-infosphere-is-winclient.mani
fest.xml   amazon   available public   [devpay: EC129708}
i386 machine windows instance-store   hvm   xen
IMAGE   ami-df20c3b6   ec2-public-windows-images/Server2003r2-i386-Win-
v1.07.manifest.xml   amazon   available   public   i386
machine   windows   instance-store   hvm   xen
IMAGE   ami-dd20c3b4   ec2-public-windows-images/Server2003r2-x86_64-Win-
v1.07.manifest.xml   amazon   available   public   x86_64
machine   windows   instance-store   hvm   xen
```

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

Configuring a Windows Instance Using the EC2Config Service

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

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

The EC2Config service runs Sysprep, a Microsoft tool that enables you to create a customized Windows
image that can be reused. For more information about Sysprep, see Sysprep Technical Reference.

When EC2Config calls Sysprep, it uses the settings files in `EC2ConfigService\Settings` to determine
which operations to perform. You can edit these files indirectly using the **Ec2 Service Properties** dialog
box, or directly using an XML editor or a text editor. However, there are some advanced settings that
aren't exposed in the **Ec2 Service Properties** dialog box, so you must edit those entries directly.

If you create an AMI from an instance after updating its settings, the new settings are applied to any
instance that's launched from the new AMI. For information about creating an AMI, see Creating an
Amazon EBS-Backed Windows AMI (p. 51).

Topics
- Overview of EC2Config Tasks (p. 41)
- Ec2 Service Properties (p. 41)
- EC2Config Settings Files (p. 46)
- Installing the Latest Version of EC2Config (p. 49)
- Stopping, Deleting, or Uninstalling EC2Config (p. 50)

Overview of EC2Config Tasks

EC2Config runs initial startup tasks when the instance is first started and then disables them. To run these tasks again, you must explicitly enable them prior to shutting down the instance, or by running Sysprep manually. These tasks are as follows:

- Set the computer name (to match the private DNS name).
- Set a random, encrypted password for the administrator account.
- Generate and install the host certificate used for Remote Desktop Connection.
- Dynamically extend the operating system partition.
- Execute the specified user data (and CloudInit.NET, if it's installed).

EC2Config performs the following tasks every time the instance starts:

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

While the instance is running, you can request that EC2Config perform the following task on demand:

- Run Sysprep and shut down the instance so that you can create an AMI from it. For more information, see Creating an Amazon EBS-Backed Windows AMI (p. 51).

Ec2 Service Properties

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

To change settings using the Ec2 Service Properties dialog box

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

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

    User Data
        User data execution enables you to inject scripts into the instance metadata during the first launch. From an instance, you can read user data at http://169.254.169.254/latest/user-data/.

This information remains static for the life of the instance, persisting when the instance is stopped and started, until it is terminated.

If you use a large script, we recommend that you use user data to download the script, and then execute it.

For EC2Config to execute user data, you must enclose the lines of the script within one of the following special tags:

`<script></script>`

Run any command that you can run at the cmd.exe prompt.

Example: `<script>dir > c:\test.log</script>`

`<powershell></powershell>`

Run any command that you can run at the Windows PowerShell prompt.

If you use an AMI that includes the AWS Tools for Windows PowerShell, you can also use those cmdlets. If you specify an IAM role when you launch your instance, then you don't need to specify credentials to the cmdlets, as applications that run on the instance can use the role's credentials to access AWS resources such as Amazon S3 buckets.

Example: `<powershell>Read-S3Object -BucketName myS3Bucket -Key myFolder/myFile.zip -File c:\destinationFile.zip</powershell>`

If both `script` and `powershell` tags are present, the batch script is executed first, and then the PowerShell script, regardless of the order in which they appear.

EC2Config expects the user data to be available in base64 encoding. If the user data is not available in base64 encoding, EC2Config logs an error about being unable to find `script` or `powershell` tags to execute. If your encoding is not correct, the following is an example that sets the encoding using PowerShell.

```
$UserData = [System.Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($Script))
```

Initial Boot

By default, all Amazon AMIs have user data execution enabled for the initial boot. If you click **Shutdown with Sysprep** in EC2Config, user data execution is re-enabled, regardless of the setting of the **User Data** check box.

User data execution happens under the local administrator user only when a random password is generated. This is because EC2Config generates the password and is aware of the credentials briefly (prior to sending to the console). EC2Config doesn't store or track password changes, so when you don't generate a random password, user data execution is performed by the EC2Config service account.

Subsequent Boots

Because the user data plug-in automatically disables after initial boot, you must do one of the following to persist user data across reboots:

- Programmatically create a scheduled task to run at system start using schtasks.exe /Create, and point the scheduled task to the user data script (or another script) at `C:\Program Files\Amazon\Ec2ConfigServer\Scripts\UserScript.ps1`.

- Programmatically re-enable the user data plug-in in `Settings.xml` using a script similar to the following.

```
<powershell>
$EC2SettingsFile="C:\Program Files\Amazon\Ec2ConfigService\Settings\Con
fig.xml"
$xml = [xml](get-content $EC2SettingsFile)
$xmlElement = $xml.get_DocumentElement()
$xmlElementToModify = $xmlElement.Plugins

foreach ($element in $xmlElementToModify.Plugin)
{
    if ($element.name -eq "Ec2SetPassword")
    {
        $element.State="Enabled"
    }
    elseif ($element.name -eq "Ec2HandleUserData")
    {
        $element.State="Enabled"
    }
}
$xml.Save($EC2SettingsFile)
</powershell>
```

- Starting with EC2Config version 2.1.10, you can use <persist>true</persist> to re-enable the plug-in after user data execution.

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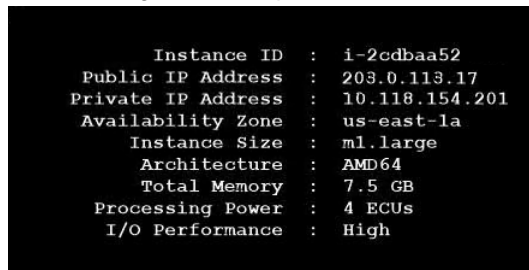
Enables the display of event log entries on the console during boot for easy monitoring and debugging.

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

Wallpaper Information

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

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

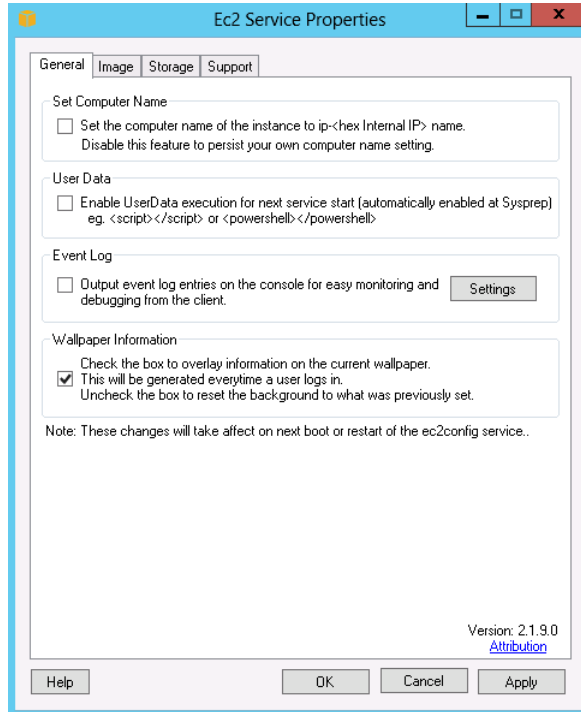4. Click the **Storage** tab. You can enable or disable the following settings.

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

    Initialize Drives
    Formats and mounts all instance store volumes attached to the instance during start.

    Drive Letter Mapping
    By default, the system maps the volumes attached to an instance to drive letters. The system can choose any available drive letter. To choose the drive letters for your volumes, click **Mappings**. In the **DriveLetterSetting** dialog box, specify the **Volume Name** and **Drive Letter** values for each volume, and then click **OK**. We recommend that you select driver letters starting at the end of the alphabet (Z:, Y:, and so on) to avoid conflicts with drive letters that are already in use.

    After you specify a drive letter mapping and attach a volume with same label as one of the volume names that you specified, EC2Config automatically assigns that volume the drive letter that you specified for it. However, the drive letter mapping fails if the drive letter is already in use. Note that EC2Config doesn't change the drive letters of volumes that were already mounted when you specified the drive letter mapping.
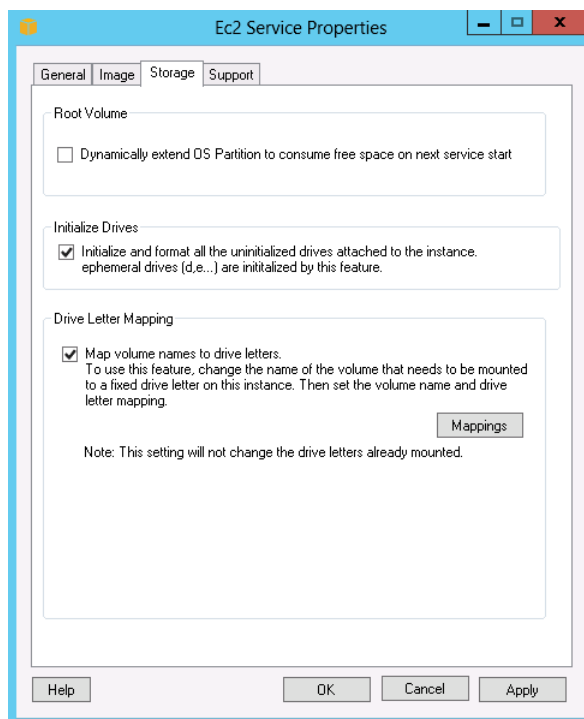
5. To save your settings and continue working on them later, click **OK** to close the **Ec2 Service Properties** dialog box.

   Otherwise, if you have finished customizing your instance and are ready to create your AMI from this instance, click the **Image** tab. Select an option for the Administrator password, and then click **Shutdown with Sysprep** or **Shutdown without Sysprep**. EC2Config edits the settings files based on the password option that you selected.
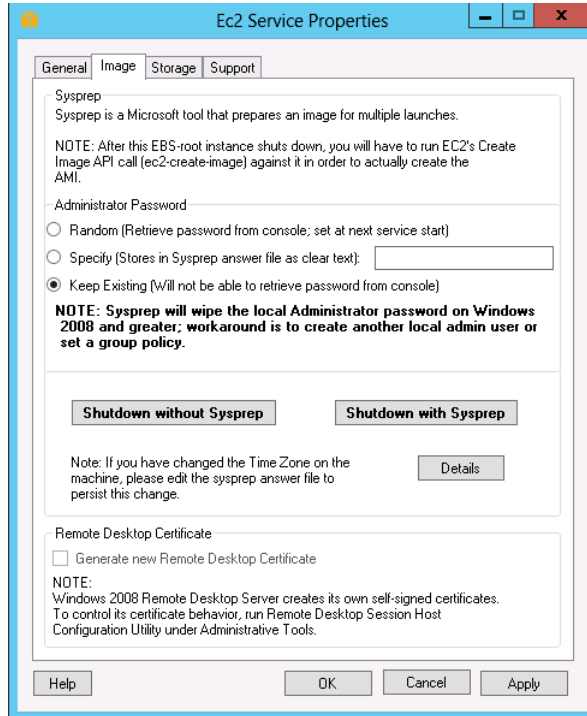
When you are asked to confirm that you want to run Sysprep and shut down the instance, click **Yes**. You'll notice that EC2Config runs Sysprep. Next, you are logged off the instance, and the instance is shut down. If you check the **Instances** page in the Amazon EC2 console, the instance state changes from `running` to `stopping`, and then finally to `stopped`. At this point, it's safe to create an AMI from this instance.

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

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

However, you must be very careful that the XML file options specified in the `Ec2ConfigService\Settings` folder are correct; otherwise, you might not be able to connect to the instance. For more information about the settings files, see EC2Config Settings Files (p. 46). For an example of configuring and then running Sysprep from the command line, see `Ec2ConfigService\Scripts\InstallUpdates.ps1`.

EC2Config Settings Files

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

- `ActivationSettings.xml`—Controls product activation using a key management server (KMS).
- `BundleConfig.xml`—Controls how EC2Config prepares an instance for AMI creation.
- `Config.xml`—Controls the primary settings.
- `DriveLetterConfig.xml`—Controls drive letter mappings.
- `EventLogConfig.xml`—Controls the event log information that's displayed on the console while the instance is booting.
- `WallpaperSettings.xml`—Controls the information that's displayed on the desktop background.

The settings in these files control the operation of the EC2Config service.

ActivationSettings.xml

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

BundleConfig.xml

- `AutoSysprep`—Indicates whether to use Sysprep automatically. Change the value to `Yes` to use Sysprep.
- `SetRDPCertificate`—Sets a self-signed certificate to the Remote Desktop server running on a Windows 2003 instance. This enables you to securely RDP into the instances. Change the value to `Yes` if the new instances should have the certificate.

  This setting is not used with Windows Server 2008 or Windows Server 2012 instances because they can generate their own certificates.
- `SetPasswordAfterSysprep`—Sets a random password on a newly launched instance, encrypts it with the user launch key, and outputs the encrypted password to the console. Change the value of this setting to `No` if the new instances should not be set to a random encrypted password.

Config.xml

Plug-ins

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

  This setting is important if you are planning to create an AMI from your instance.
- `Ec2SetComputerName`—Sets the hostname of the instance to a unique name based on the IP address of the instance and reboots the instance. To set your own hostname, or prevent your existing hostname from being modified, you must disable this setting.
- `Ec2InitializeDrives`—Initializes and formats all instance store volumes during startup. This feature is enabled by default, and initializes and mounts the instance store volumes as drives D:/, E:/, and so on. For more information about instance store volumes, see Amazon EC2 Instance Store in the Amazon Elastic Compute Cloud User Guide.
- `Ec2EventLog`—Displays event log entries in the console. By default, the three most recent error entries from the system event log are displayed. To specify the event log entries to display, edit the `EventLogConfig.xml` file located in the `EC2ConfigService\Settings` directory. For information about the settings in this file, see Eventlog Key in the MSDN Library.
- `Ec2ConfigureRDP`—Sets up a self-signed certificate on the instance, so users can securely access the instance using Remote Desktop. This feature is disabled on Windows Server 2008 and Windows Server 2012 instances because they can generate their own certificates.
- `Ec2OutputRDPCert`—Displays the Remote Desktop certificate information to the console so that the user can verify it against the thumbprint.

- `Ec2SetDriveLetter`—Sets the drive letters of the mounted volumes based on user-defined settings. By default, when an Amazon EBS volume is attached to an instance, it can be mounted using the drive letter on the instance. To specify your drive letter mappings, edit the `DriveLetterConfig.xml` file located in the `EC2ConfigService\Settings` directory.
- `Ec2WindowsActivate`—Indicates whether to search through the DNS Suffix List for appropriate KMS entries. When the appropriate KMS entries are found, the plug-in sets your activation server to the first server to respond to the request successfully. Starting with Windows Server 2008 R2, Windows Server is able to search the suffix list automatically. With Windows Server 2008 R2 and Windows Server 2012, the plug-in performs this search manually.

  To modify the KMS settings, edit the `ActivationSettings.xml` file located in the `EC2ConfigService\Settings` directory.
- `Ec2DynamicBootVolumeSize`—Extends Disk 0/Volume 0 to include any unpartitioned space.
- `Ec2HandleUserData`—Creates and executes scripts created by the user on the first launch of an instance after Sysprep is run. Commands wrapped in script tags are saved to a batch file, and commands wrapped in PowerShell tags are saved to a .ps1 file.

Global Settings

- `ManageShutdown`—Ensures that instances launched from instance store-backed AMIs do not terminate while running Sysprep.
- `SetDnsSuffixList`—Sets the DNS suffix of the network adapter for Amazon EC2. This allows DNS resolution of servers running in Amazon EC2 without providing the fully qualified domain name.
- `WaitForMetaDataAvailable`—Ensures that the EC2Config service will wait for metadata to be accessible and the network available before continuing with the boot. This check ensures that EC2Config can obtain information from metadata for activation and other plug-ins.
- `ShouldAddRoutes`—Adds a custom route to the primary network adapter to enable the following IP addresses when multiple NICs are attached: 169.254.169.250, 169.254.169.251, and 169.254.169.254. These addresses are used by Windows Activation, and when you access instance metadata.
- `RemoveCredentialsfromSyspreponStartup`—Removes the administrator password from `Sysprep.xml` the next time the service starts. To ensure that this password persists, edit this setting.

DriveLetterConfig.xml

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

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

- `VolumeName`—The volume label. For example, `My Volume`.
- `DriveLetter`—The drive letter. For example, `X:`.

EventLogConfig.xml

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

WallpaperSettings.xml

- `Instance ID`—Displays the ID of the instance.
- `Public IP Address`—Displays the public IP address of the instance.
- `Private IP Address`—Displays the private IP address of the instance.
- `Availability Zone`—Displays the Availability Zone in which the instance is running.
- `Instance Size`—Displays the type of instance.
- `Architecture`—Displays the setting of the `PROCESSOR_ARCHITECTURE` environment variable.
- `AddMemory`—Displays the system memory, in GB.
- `AddECU`—Displays the processing power, in ECU.
- `AddIO`—Displays the I/O performance.

Installing the Latest Version of EC2Config

By default, the EC2Config service is included in each AWS Windows AMI. When we release an updated version, we update all AWS Windows AMIs with the latest version. However, you'll need to update your own Windows AMIs and instances with the latest version.

To find notifications of updates to EC2Config, go to the Amazon EC2 forum.

To verify the version of EC2Config included with your Windows AMI

1. Launch an instance from your AMI and connect to it.
2. In Control Panel, select **Programs and Features**.
3. In the list of installed programs, look for `Ec2ConfigService`. Its version number appears in the **Version** column.

To install the latest version of EC2Config

1. Go to Amazon Windows EC2Config Service.
2. Click **Download**.
3. Download and unzip the file.
4. Run `EC2Install.exe`. The setup program stops the service, uninstalls it, and reinstalls the new version.
5. Reboot your instance.
6. Connect to your instance, run the Services administrative tool, and verify that the status of `EC2Config service` is `Started`.

For more information about the changes in each version, see the What's New section on the download page.

Stopping, Deleting, or Uninstalling EC2Config

You can manage the EC2Config service just as you would any other service.

To apply updated settings to your instance, you can stop and restart the service. If you're manually installing EC2Config, you must stop the service first.

To stop the EC2Config service

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

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

To delete the EC2Config service

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

```
sc delete ec2config
```

If you don't need to update the configuration settings or create your own AMI, you can uninstall EC2Config. Uninstalling a service removes the files, the registry subkey, and any shortcuts to the service.

To uninstall EC2Config

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

Creating Your Own Windows AMI

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

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

For information about launching, connecting, and using your Windows instance, see Amazon EC2 Instances.

Before you create your own AMI, you can configure your base customized instance. The new configuration applies to all the instances that are launched from the new AMI. Your Amazon EC2 Windows instance comes with a configuration tool, the EC2Config. You can use this tool to configure your instance. For information about using the EC2Config Service, see Configuring a Windows Instance Using the EC2Config Service (p. 40)

The root storage device that you selected for the AMI determines the process you follow to create the AMI. The AMI is an Amazon EBS-backed AMI or an Amazon EC2 instance store-backed AMI. There are significant differences between Amazon EBS-backed and Amazon EC2 instance store-backed AMIs, including AMI size limits, storage, and persistence of data. For information about the differences between these AMI types, see Root Device Storage on Windows AMIs (p. 36).

For more information about instructions for creating an Amazon EBS-backed Windows AMI, see Creating an Amazon EBS-Backed Windows AMI (p. 51). For more information about instructions for creating an instance store-backed Windows AMI, see Creating an Instance Store-Backed Windows AMI (p. 53).

Creating an Amazon EBS-Backed Windows AMI

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

The process for creating an instance store-backed AMI is different. For more information, see Creating an Instance Store-Backed Windows AMI (p. 53).

To prepare to create an Amazon EBS-backed AMI

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, click **AMIs**. Select an Amazon EBS-backed AMI that is similar to the AMI that you will create. To view the Amazon EBS-backed Windows AMIs, select the following options from the **Filter** lists: **Public Images**, **EBS Images**, and then **Windows**.

   You can select any public AMI that uses the version of Windows Server that you will use for your AMI. However, you must select an Amazon EBS-backed AMI; don't start with an instance store-backed AMI.
3. Click **Launch** to launch an instance of the Amazon EBS-backed AMI that you've selected. Accept the default values as you step through the wizard.

   For more information about launching a Windows instance using the AWS Management Console, see Launch a Windows Instance (p. 11).
4. While the instance is running, connect to it and customize it. For example, you can perform any of the following actions on your instance:

   a. Install software and applications.
   b. Copy data.
   c. Reduce start time by deleting temporary files, defragmenting your hard drive, and zeroing out free space.
   d. Create a new user account and add it to the Administrators group.
   e. Configure the settings using EC2Config. For more information, see Configuring a Windows Instance Using the EC2Config Service (p. 40).

   For information about connecting to a Windows instance using the AWS Management Console, see Connecting to Your Windows Instance (p. 14).
5. When the instance is set up the way you want it, it is best to stop the instance before you create the AMI, to ensure data integrity. If you didn't use EC2Config to stop the instance already, use the following steps to stop the instance.

    a.   Right-click your running instance and select **Stop Instance**.

    b.   In the confirmation dialog box, click **Yes, Stop Instance**.

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

To create an Amazon EBS-backed AMI

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

       The **Create Image** dialog box opens.

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

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

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

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

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

5.   Click **Yes, Create** to start creating the AMI.

6.   Go to the **AMIs** page and view the status of your AMI. While your AMI is being created, its status is `pending`.

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

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

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

To delete an AMI and a snapshot

1.   Go to the **AMIs** page. Select the AMI, click **Actions**, and select **Deregister**. When asked for confirmation, click **Continue**.

2.   Go to the **Snapshots** page. Right-click the snapshot and select **Delete Snapshot**. When asked for confirmation, click **Yes, Delete**.

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

Creating an Instance Store-Backed Windows AMI

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

The process for creating an Amazon EBS-backed Windows AMI is different. For more information, see Creating an Amazon EBS-Backed Windows AMI (p. 51).

Overview of Instance Store-Backed Windows AMIs

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

When you create an instance store-backed AMI, it must be uploaded to Amazon S3. Amazon S3 stores data objects in buckets, which are similar in concept to directories. Buckets have globally unique names and are owned by unique AWS accounts.

Bundling Process

The bundling process comprises the following tasks:

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

Storage Volumes

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

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

For more information about Amazon EC2 storage options, see Storage.

Preparing to Create an Instance Store-Backed Windows AMI

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

To prepare to create an instance store-backed Windows AMI

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, click **AMIs**. Select an instance store-backed AMI that is similar to the AMI that you will create. To view the instance store-backed Windows AMIs, select the following options from the **Filter** lists: **Public Images**, **Instance Store Images**, and then **Windows**.

   You can select any public AMI that uses the version of Windows Server that you will use for your AMI. However, you must select an instance store-backed AMI; don't start with an Amazon EBS-backed AMI.
3. Click **Launch** to launch an instance of the instance store-backed AMI that you've selected. Accept the default values as you step through the wizard.

   For more information about launching a Windows instance using the AWS Management Console, see Launch a Windows Instance (p. 11).
4. While the instance is running, connect to it and customize it. For example, you can perform any of the following on your instance:

   a. Install software and applications.
   b. Copy data.
   c. Reduce start time by deleting temporary files, defragmenting your hard drive, and zeroing out free space.
   d. Create a new user account and add it to the Administrators group.
   e. Configure settings using EC2Config. For more information, see Configuring a Windows Instance Using the EC2Config Service (p. 40).

   For information about connecting to a Windows instance using the AWS Management Console, see Connecting to Your Windows Instance (p. 14).

Bundling an Instance Store-Backed Windows AMI

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

To bundle an Amazon EC2 instance store-backed AMIs

1. Determine whether you'll use an existing Amazon S3 bucket for your new AMI or create a new one. To create a new Amazon S3 bucket, use the following steps:

   a. Open the Amazon S3 console at https://console.aws.amazon.com/s3.
   b. Click **Create Bucket**.
   c. Specify a name for the bucket and click **Create**.

2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
3. Right-click the instance and select **Bundle Instance (instance store AMI)**.

The **Bundle Instance** dialog box opens.

4. Fill in the requested information, and then click **Bundle**.

    a. Specify the name of an S3 bucket that you own in **Amazon S3 Bucket Name**.
    b. Specify a prefix for the files to be generated by the bundle process in **Amazon S3 Key Name**.

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

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

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

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

Registering an Instance Store-Backed Windows AMI

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

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

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, click **AMIs**. By default, the console displays the AMIs that you own.
3. Select your newly-bundled AMI, then click **Actions** and select **Register New AMI**.
4. In the **Register Image** dialog box, provide the **AMI Manifest Path** and click **Register**.

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

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

Shared Windows AMIs

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

Creating Windows AMIs for Sharing

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

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

1. Follow the instructions to launch and connect to a Windows instance.
2. Customize the instance by installing the software and applications to share. Do the following to make your AMI safe and secure for sharing:

- Always delete the shell history before bundling. The shell history may contain sensitive information.
- If you have saved your instance credentials, such as your key pair, remove them or move them to a location that is not going to be included in the AMI.
- Ensure that the administrator password and passwords on any other accounts are set to an appropriate value for sharing. These passwords are available for anyone who launches your shared AMI.
- Remove any saved passwords that you do not want to share.
- Make sure to test your AMI before you release it to the public.

3. Run Sysprep to prepare the instance and enable the new password generation on new instance launch. The instance shuts 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. 55).

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 does 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. You can also paste the following information into the document. You must be in HTML edit mode.

```
<strong>&AMI; ID: </strong>[ami-id]<br />
<strong>AMI Manifest: </strong>[myawsbucket/image.manifest.xml]<br />
```

```
<h2>About this &AMI;</h2>
<ul>

    <li>Published by [Publisher] (<a href="http://www.mysite.com">[ht
tp://www.mysite.com]</a>).<br />
            </li>
    <li>[Key Features] <br />
            </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">     - [Note 1]</span><br />
<span style="font-size: x-small; font-family: courier new,couri
er">     - [Note 2]</span><br />
<span style="font-size: x-small; font-family: courier new,couri
er">     - [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 (that is, 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, see ec2-describe-images in the *Amazon Elastic Compute Cloud Command Line Reference*.

Safe Use of Shared AMIs

You launch AMIs at your own risk. We cannot vouch for the integrity or security of AMIs shared by other Amazon 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 Amazon 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 about 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 that you can purchase from other developers.

Amazon EC2 integrates with Amazon DevPay, allowing developers to charge other Amazon EC2 users for the use of their AMIs or to provide support for instances. For more information about Amazon DevPay, see the Amazon DevPay Developer Guide.

Note
All paid AMIs from Amazon DevPay are backed by Amazon instance store.

Find Paid AMIs

There are several ways you can determine what paid AMIs are available for 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 (for example, `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 into AWS, and then you see the paid AMI's price and confirm that you will purchase the AMI.

Important
You don't get the discount from Amazon EC2 Reserved Instances with paid AMIs. That is, if you purchase Reserved Instances, you don't get the lower price associated with them when you launch a paid AMI. You always pay the price that the seller of the paid AMI specified. For more information about Reserved Instances, see On-Demand and Reserved Instances.

Launch Paid AMIs

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

After you purchase a paid AMI, you can launch instances of it. Launching a paid AMI is the same as launching any other AMI. No additional parameters are required. The instance is 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 is able to confirm if a particular instance was launched using that 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, see 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
After it is 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 more information about the usage information for your paid and supported AMIs, go to the Amazon Payments sign-in page.

AWS Management Pack for Microsoft System Center Operations Manager

Amazon Web Services (AWS) offers a complete set of infrastructure and application services that enable you to run virtually everything in the cloud—from enterprise applications and big data projects, to social games and mobile apps. The AWS Management Pack for Microsoft System Center Operations Manager provides availability and performance monitoring capabilities for your applications running in AWS.

The AWS Management Pack links Amazon EC2 instances and the Microsoft Windows or Linux operating systems running inside them. The AWS Management Pack is an extension to Microsoft System Center Operations Manager. It uses a designated computer in your datacenter (called a watcher node) and the Amazon Web Services APIs to remotely discover and collect information about your AWS resources. You configure the AWS Management Pack to discover information about your AWS resources by running the Operations Manager Add Monitoring Wizard. For more information, see Step 1: Installing the AWS Management Pack (p. 67).

You can use the AWS Management Pack to monitor the following AWS resources:

- Amazon Elastic Compute Cloud (Amazon EC2) instances
- Amazon Elastic Block Store (Amazon EBS) volumes
- Elastic Load Balancing
- AWS Elastic Beanstalk
- AWS CloudFormation stacks
- Auto Scaling groups and Availability Zones

The AWS Management Pack uses Amazon CloudWatch metrics and alarms to monitor AWS resources. Amazon CloudWatch metrics appear in Microsoft System Center as performance counters, while Amazon CloudWatch alarms appear as alerts.

System Requirements

Before downloading the AWS Management Pack, you must ensure that your systems meet the following requirements:

- System Center Operations Manager 2007 R2 or System Center Operations Manager 2012 SP1
- For System Center 2012, the Amazon Web Services Management pack has dependencies on Microsoft.Unix.Library MP version 7.3.2026.0 or above.
- For System Center 2007 R2, the Amazon Web Services Management Pack has dependencies on Microsoft.Unix.Library MP version 6.1.7000.256 or above.
- For System Center Operations Manager 2012, Cumulative Update 1 or above is required. The update must at least be deployed to the management servers participating in Amazon Web Services monitoring, as well as agents running the watcher nodes and agents that will be monitored by Amazon MP. It is recommended that you run the latest publicly available Operations Manager updates on all computers participating in Amazon Web Services monitoring.

Prerequisites

Before downloading the AWS Management Pack, you must ensure that your systems meet the following prerequisites:

- A designated agent-managed computer in your datacenter that you designate as the watcher node with the Agent Proxy option, **Allow this agent to act as a proxy and discover managed objects on other computers**, enabled.
- The action account for the watcher node must have local administrator privileges on the watcher node.
- The watcher node must have Internet connectivity so that it can make AWS API calls.
- The Microsoft .NET Framework version 3.5.1 or later must be installed on the watcher node.
- The Amazon CloudWatch service must be enabled for your AWS account.
- The Amazon EC2 instances you want to manage must be running Microsoft System Center – Operations Manager agents for linkage between Amazon EC2 instances and the Windows or Linux operating system running inside them to work. If you use this feature, you must make sure that the agents are deployed, running, and can communicate with the management servers in your datacenter.

Downloading the AWS Management Pack

Before you can monitor your AWS resources, you must download the AWS Management Pack. The AWS Management Pack is free. You only pay for the AWS resources that you choose to monitor (for example, Amazon EC2 instances, Elastic Load Balancing, or Amazon CloudWatch metrics and alarms).

To download the AWS Management Pack

1. On the AWS Management Pack for Microsoft System Center website, click **Download AWS MP for SCOM 2007 R2** or **Download AWS MP for SCOM 2012**.
2. When prompted, save **Amazon.AmazonWebServices.mpb** or the **AWS MP Setup.msi** file to your computer.

Deploying the AWS Management Pack

Before using the following steps to import and deploy the AWS Management Pack, you should familiarize yourself with the various components for monitoring your AWS resources and choose the computer that will serve as the watcher node. You also must determine the AWS credentials that you want to require for monitoring your AWS resources. The main components are shown in the following diagram.

Item	Component	Description
1	Operations Manager infrastructure	One or more management servers and its dependencies, such as Microsoft SQL Server and a Microsoft Active Directory domain. These servers can either be deployed on-premises or in the AWS cloud; both scenarios are supported.
2	Watcher node	A designated agent-managed computer used for communicating with AWS using the AWS SDK for .NET. It can either be deployed on-premises or in the AWS cloud, but it must be an agent-managed computer and must have Internet connectivity. You can use exactly one watcher node for monitoring an AWS account. However, you can share the same watcher node for monitoring multiple AWS accounts.

Item	Component	Description
❸	AWS credentials	An access key ID and a secret access key used by the watcher node to make AWS API calls. You must specify these credentials while configuring the AWS Management Pack. We recommend that you create an IAM user with read-only privileges and use those credentials. For more information about creating an IAM user, see Adding a New User to Your AWS Account in *Using IAM*.
❹	Amazon EC2 instances	Virtual computers running in the AWS cloud. Some Amazon EC2 instances may have the Operations Manager Agent installed, others may not. You can get deeper insights when Operations Manager Agent is installed because you can see the operating system and application health apart from the instance health.

Topics

Step 1: Installing the AWS Management Pack

After Downloading the AWS Management Pack you must import and configure it for monitoring one or more AWS accounts.

To install the AWS Management Pack for System Center 2012

1. In the Microsoft System Center Operations Manager **Operations Console**, on the **Go** menu, click **Administration**.
2. Right-click **Management Packs**, and then click **Import Management Packs**.
3. In the **Import Management Packs Wizard**, click **Add**, and then click **Add from disk**.
4. In the **Select Management Packs to import** dialog box, click **Amazon.AmazonWebServices.mpb** to import from the directory you downloaded it in, and then click **Open**.
5. On the **Select Management Packs** page, the Amazon Web Services Management pack that you selected for import is listed. Click **Import**.

    Note
    When you click **Import**, any management packs in the **Import** list that display the **Error** icon are not imported.

6. The **Import Management Packs** page appears and shows the progress for the management pack. If there is a problem at any stage of the import process, select the management pack in the list to view the status details. Click **Close**.

To install the AWS Management Pack for System Center 2007 R2

For System Center 2007 the management pack is distributed as a Microsoft System Installer file, AWS_MP_Setup.msi. It contains the required DLLs for the watcher node and System Center Operations Manager Root Server and Operations Console, as well as the Amazon.AmazonWebServices.mp file.

Note
If your Root Management Server, Operations Console, and AWS Watcher Node are on different computers, you will have to run the installer on each computer.

1. Run the **AWS_MP_Setup.msi** file.

2. On the **Welcome to the Amazon Web Services Management Pack Setup Wizard** screen, click **Next**

3. On the **End-User License Agreement** screen, read the license agreement, select the **I accept the terms in the License Agreement** check box, and then click **Next**.

4. On the **Custom Setup** screen, select the features you want to install, and then click **Next**.

   • Operations Console Component — installs the Amazon.AmazonWebServices.UI.Pages.dll library and registers it in the Global Assembly Cache (GAC), and installs the AWS management pack file Amazon.AmazonWebServices.mp.

   • Root Management Server — installs the Amazon.AmazonWebServices.Modules.dll library and registers it in the GAC.

   • AWS Watcher Node — installs the Amazon.AmazonWebServices.Modules.dll library and registers it in the GAC, and installs the AWS SDK for .NET (AWSSDK.dll) into the GAC.

5. On the **Ready to install Amazon Web Services Management Pack** screen, click **Install**.

6. On the **Completed the Amazon Web Services Management Pack Setup Wizard** screen, click **Finish**.

   Note
   The required DLLs will be copied and registered in the GAC, and the management pack file (*.mp) will be copied to the Program Files (x86)/Amazon Web Services Management Pack folder on the computer running the Operations Console. You must manually import the management pack into SCOM 2007 R2 SP1, just like any other management pack.

7. In the **Operations Console**, on the **Go** menu, click **Administration**.

8. In the **Administration** navigation pane, right-click **Administration**, and then click **Import Management Packs**.

9. In the **Import Management Packs** wizard, click **Add**, and then click **Add from disk**.

10. In the **Select Management Packs to import** dialog box, change to the directory to `C:\Program Files (x86)\Amazon Web Services Management Pack`, which holds your management pack file, click **Amazon.AmazonWebServices.mp**, and then click **Open**.

11. On the **Select Management Packs** page, in the **Import list**, select the **Amazon Web Services** management pack, and then click **Install**.

    Note
    When you click **Install**, any management packs in the **Import list** that display an **Error** icon are not imported.

12. The **Import Management Packs** page appears and shows the progress for the management pack. If there is a problem at any stage of the import process, select the management pack in the list to view the status details. Click **Close**.

Step 2: Configuring the Watcher Node

The watcher node runs discoveries that go beyond the watcher node computer, so you must enable the proxy agent option on the watcher node. The proxy agent allows those discoveries to manipulate the objects on other computers.

To enable the proxy agent

1. In the Microsoft System Center Operations Manager **Operations Console**, on the **Go** menu, click **Administration**.
2. In the **Administration** workspace, under **Device Management**, click **Agent Managed**.
3. In the list of Agent Managed items, right-click the watcher node, and then click **Properties**.
4. In the **Agent Properties** dialog box, click the **Security** tab, select the **Allow this agent to act as proxy and discover managed objects on other computers** check box, and then click **OK**.
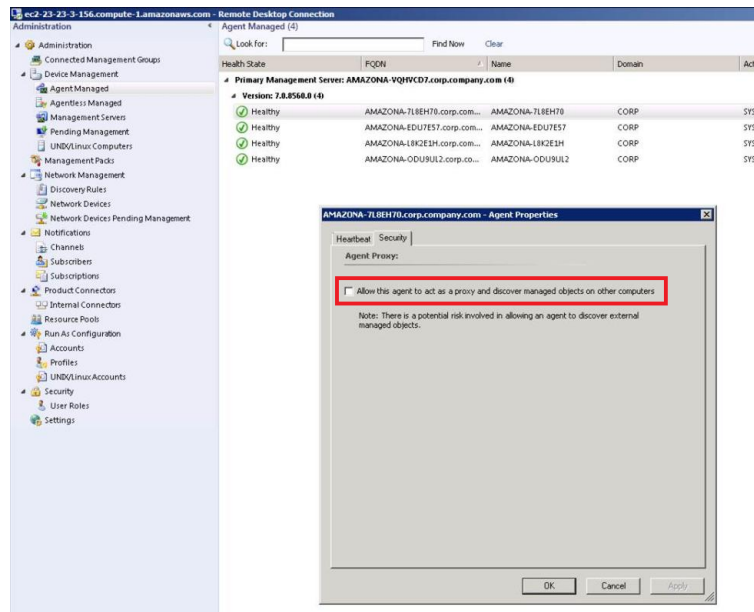
Step 3: Create an AWS Run As Account

To create an AWS Run As account

1. In the Microsoft System Center Operations Manager **Operations Console**, on the **Go** menu, click **Administration**.
2. In the **Administration** workspace, expand the **Run As Configuration** node, and then select **Accounts**.
3. Right-click the **Accounts** pane, and then click **Create Run As Account**.
4. In the **Create Run As Account Wizard**, on the **General Properties** page, in the **Run As account type** drop-down list, select **Basic Authentication**.
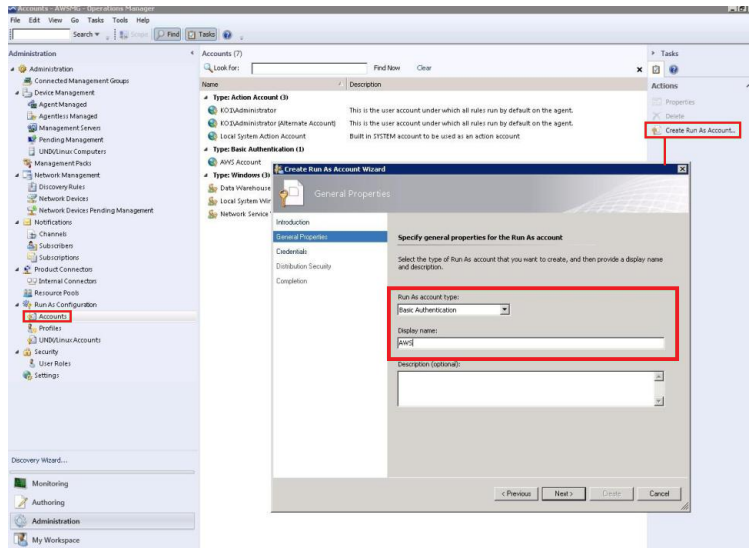
5.   In the **Display name** box provide a display name (for example, "John IAM Account") and in the
     Description box, provide a description.
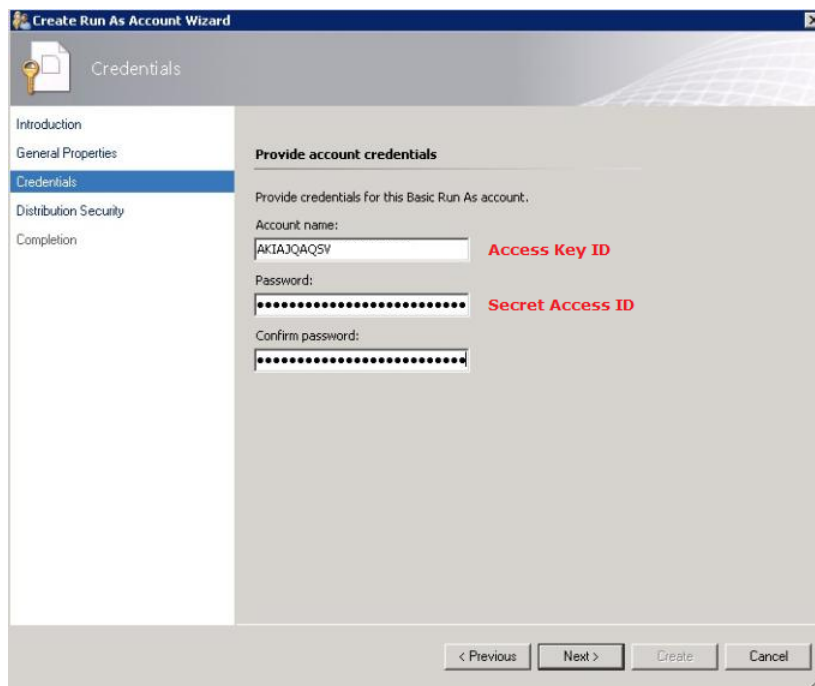
6.   Click **Next**, on the **Credentials** page, in the **Account name** box, enter the access key ID and in the
     Password box, enter the secret access key.
7.   Click **Next**, on the **Distribution Security** page, select **More secure - I want to manually select the
     computers to which the credentials will be distributed**.

8. Click **Create**, and then click **Close** to complete the Run As account creation.

Step 4: Run the Add Monitoring Wizard

You configure the AWS Management Pack for monitoring a particular AWS account by using the Add Monitoring Wizard, which is available in the Authoring workspace of the Operations Console. It creates a new management pack containing the settings for the AWS account to monitor. You have to run the wizard every time you want to monitor a new AWS account; that is, if you want to monitor two AWS accounts, you run the wizard twice.

To run the Add Monitoring Wizard

1. In the Microsoft System Center Operations Manager **Operations Console**, on the **Go** menu, click **Authoring**.
2. In the **Authoring** workspace, expand the **Management Pack Templates** node, right-click **Amazon Web Services**, and then click **Add Monitoring Wizard**.
3. In the **Add Monitoring Wizard**, in the **Select the monitoring type list**, select **Amazon Web Services**, and then click **Next**.
4. On the **General Properties** page, in the **Name** box, enter a name (for example, "John AWS Resources") and in the **Description** box, enter a description.
5. In the **Select destination management pack** drop-down list, select an existing management pack (or click **New** to create a new one) where you'd like to save the settings, and then click **Next**.
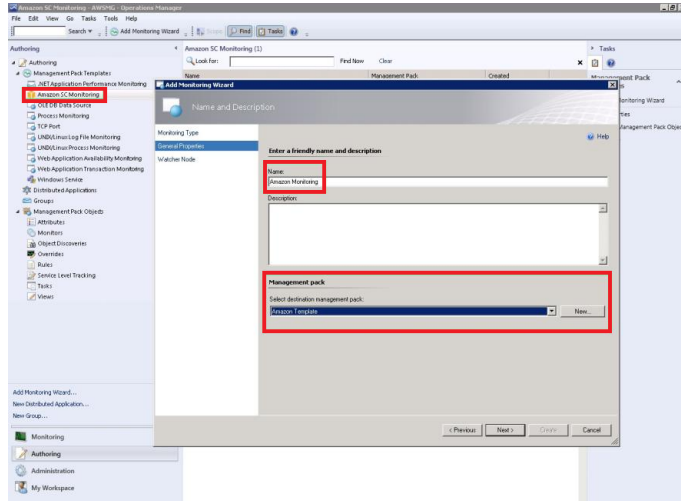
Note

By default, when you create a management pack object, disable a rule or monitor, or create an override, Operations Manager saves the setting to the default management pack. As a best practice, you should create a separate management pack for each sealed management pack that you want to customize, instead of saving your customized settings to the default management pack.

6. On the **Watcher Node Configuration** page, in the **Watcher Node** list, select an agent-managed computer to act as the watcher node.
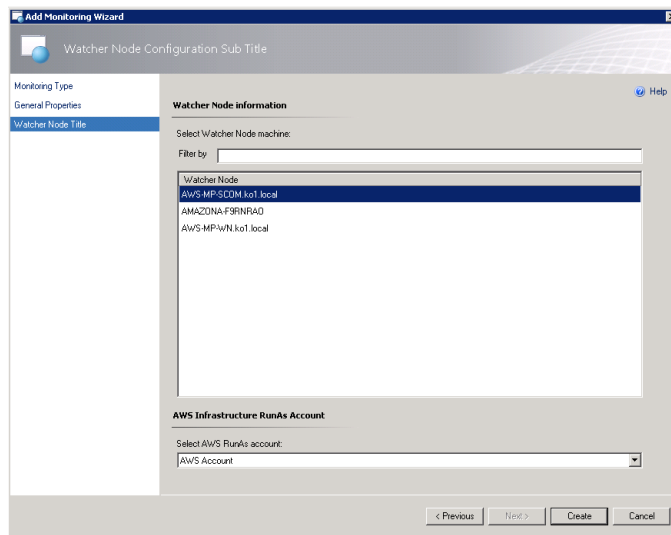
7. In the **Select AWS Run As account** drop-down list, select the Run As account you created in the previous step, and then click **Create**.

8. After the AWS Management Pack is configured, it first discovers the watcher node. To verify that the watcher node was discovered successfully, navigate to the **Monitoring** workspace in the Operations Console. You should see a new Amazon Web Services folder and an Amazon Watcher Nodes subfolder under it. This subfolder displays the Watcher Nodes. The AWS Management pack automatically checks and monitors the watcher node connectivity to Amazon Web Services. When the watcher node is discovered, it shows up in this list. When the watcher node is ready, its state changes to Healthy.

Note

To establish connectivity with Amazon Web Services, the AWS Management Pack requires the AWS SDK for .NET, modules, and scripts—to be deployed to the watcher node. This may take about ten minutes. If the watcher node doesn't appear, or if you see the state as Not Monitored, then double-check your Internet connectivity and IAM permissions. For more information, see Troubleshooting the AWS Management Pack (p. 86).

9. After the watcher node is discovered, dependent discoveries are triggered and you see AWS resources appearing in the Monitoring workspace of the Operations Console.

Note

The discovery of AWS resources should complete within twenty minutes, but may take more time, based on your Operations Manager environment, your AWS environment, the load on the management server, and the load on the watcher node. For more information, see Troubleshooting the AWS Management Pack (p. 86).

Using the AWS Management Pack

This section shows you how to use AWS Management Pack views and tasks to monitor the health of your AWS resources, your metrics, and to perform context-aware tasks.

Topics

Views

The AWS Management Pack provides the following views, which are displayed in the Monitoring workspace of the Operations Console:

- **Watcher Nodes State View**

  Shows the health state of the watcher nodes across all of the AWS accounts that are being monitored. A Healthy state means that the watcher node is configured correctly and can communicate with AWS.
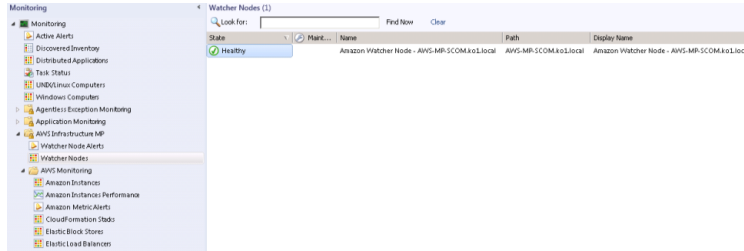
- **EC2 Instances State View**
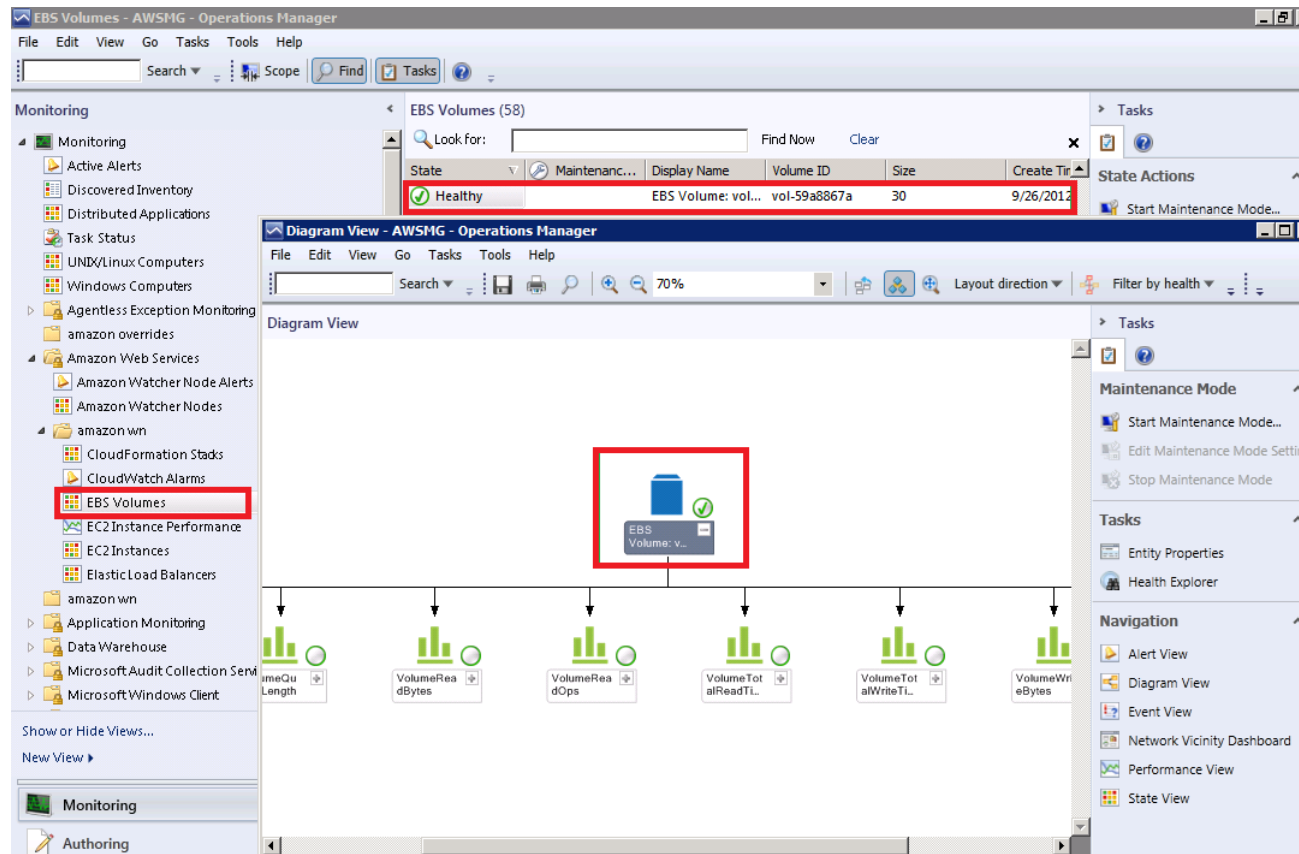
    Shows the health state of all the Amazon EC2 instances for a particular AWS account, from all Availability Zones and regions. The view also includes Amazon EC2 instances running in Amazon Virtual Private Cloud (VPC). The AWS Management Pack retrieves Amazon EC2 tags, so you can search and filter the list using those tags. The "Windows Computer" and "UNIX/Linux Computer" columns help you determine whether Operations Manager Agent is running inside the Amazon EC2 instance.
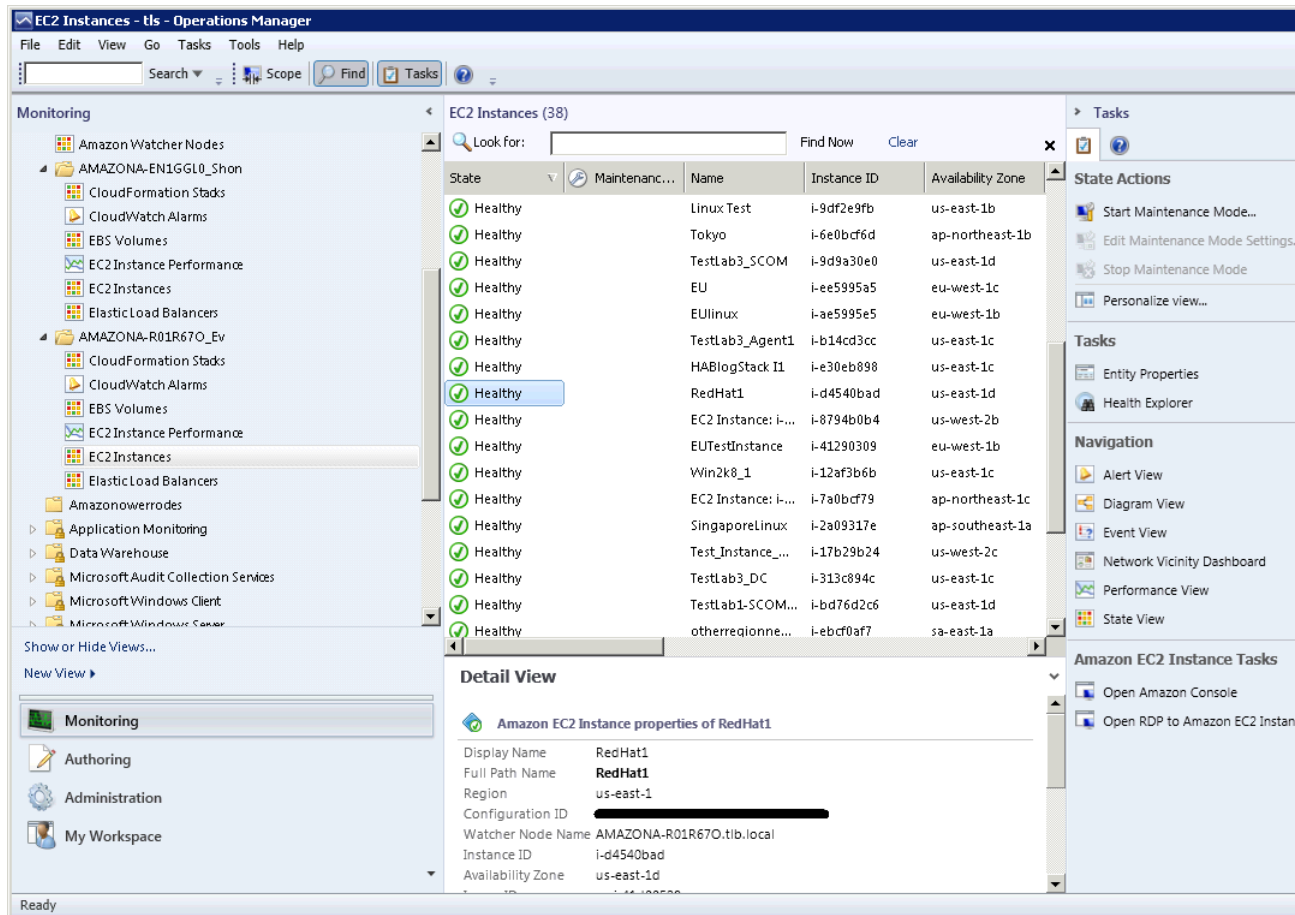
- **AWS Performance View**

    Shows the default Amazon CloudWatch metrics for Amazon EC2, Amazon EBS, and Elastic Load Balancing. For more information about these metrics, see the CloudWatch Metrics, Namespaces, and Dimensions Reference in the *Amazon CloudWatch Developer Guide*.

    The following illustration shows an example:
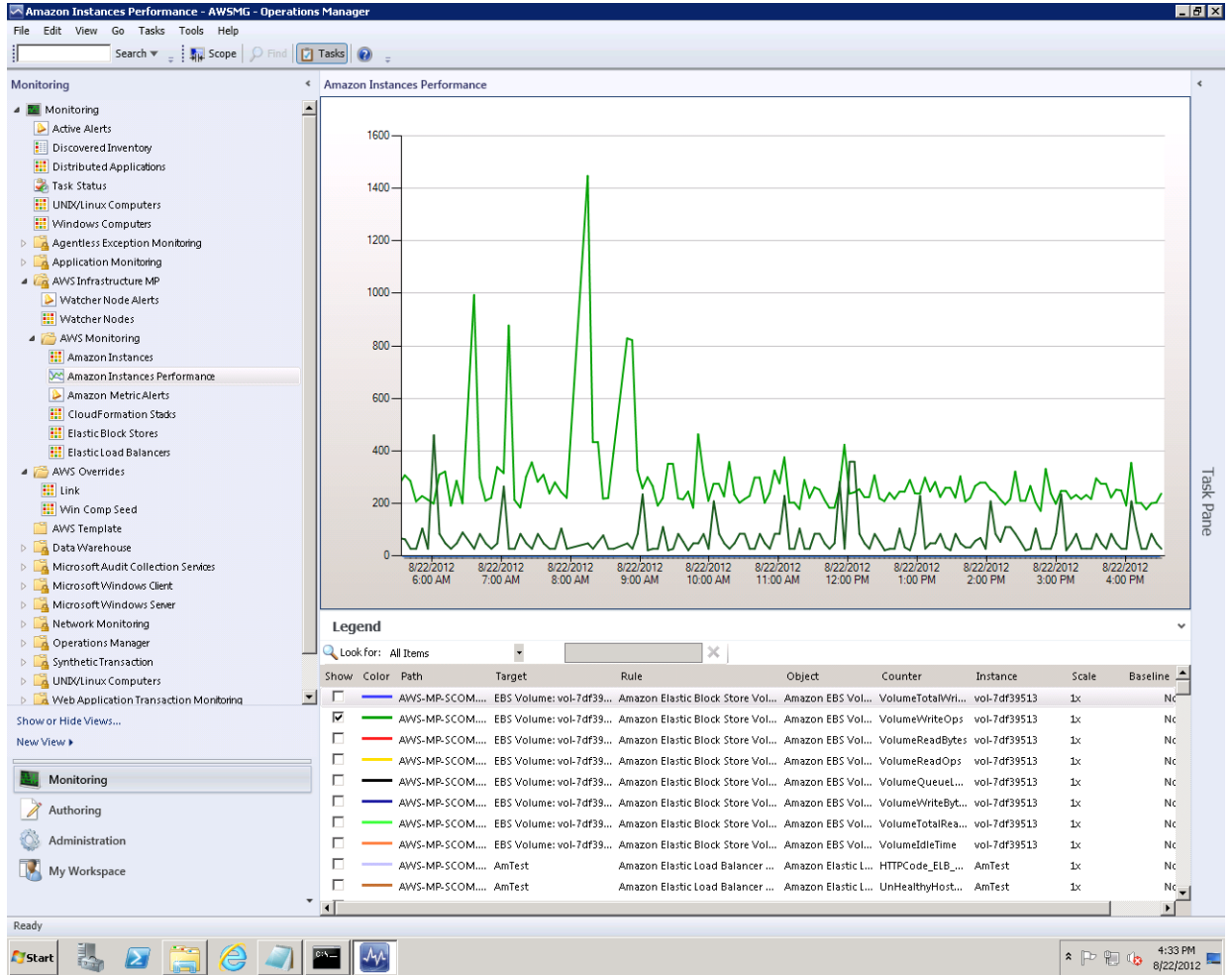
- **Instance Diagram View**

  Shows the relationship of an Amazon EC2 instance with other components.
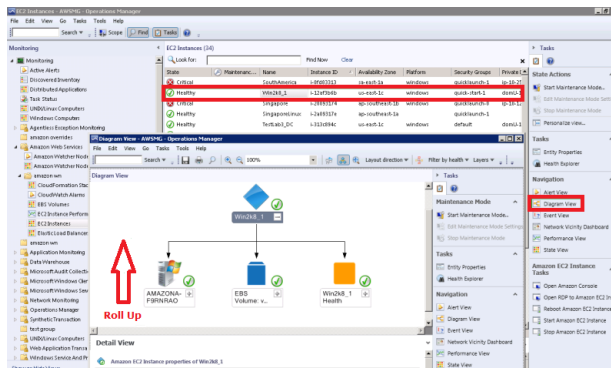
  If a relationship between an Amazon EC2 instance and its operating system can be established, the diagram view automatically displays the operating system along with its underlying components.

  The following illustration shows an example of an Amazon EC2 instance running Windows:
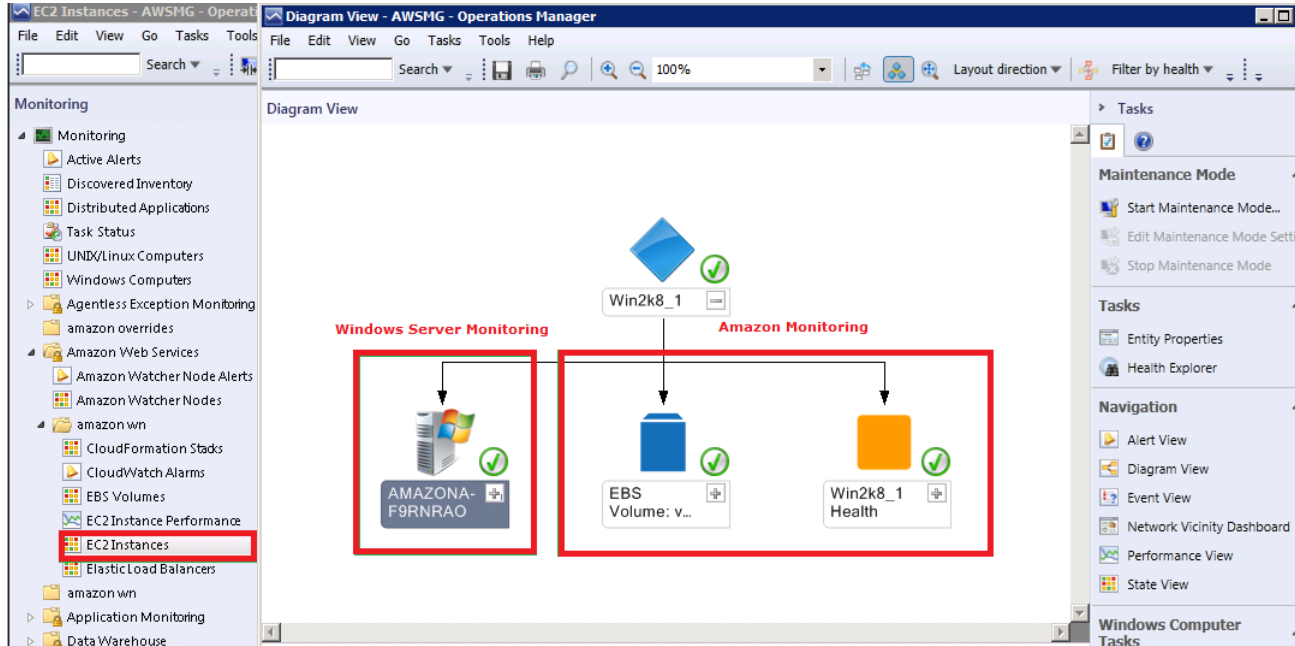
The following illustration shows an example of an Amazon EC2 instance running UNIX:
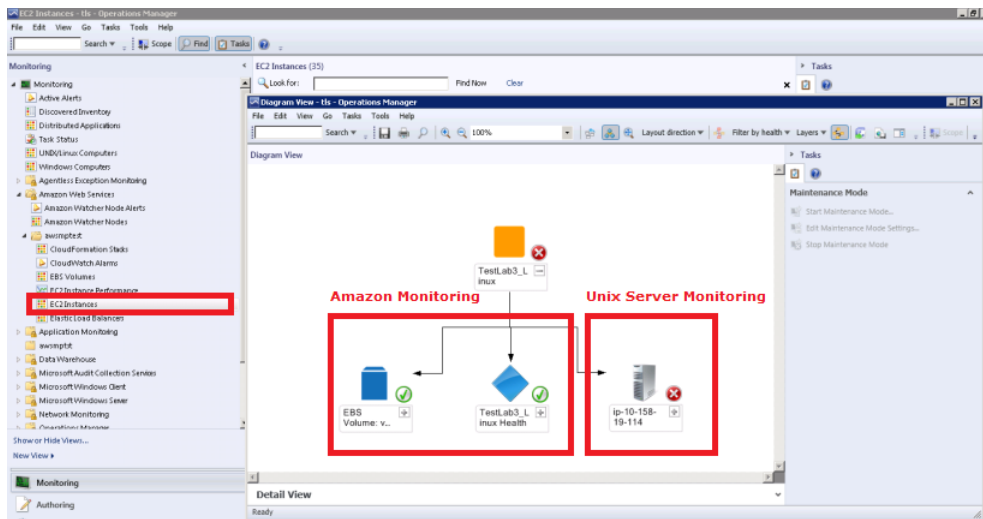
- **AWS Alerts View**

  Shows Amazon CloudWatch alarms related to the discovered AWS resources.
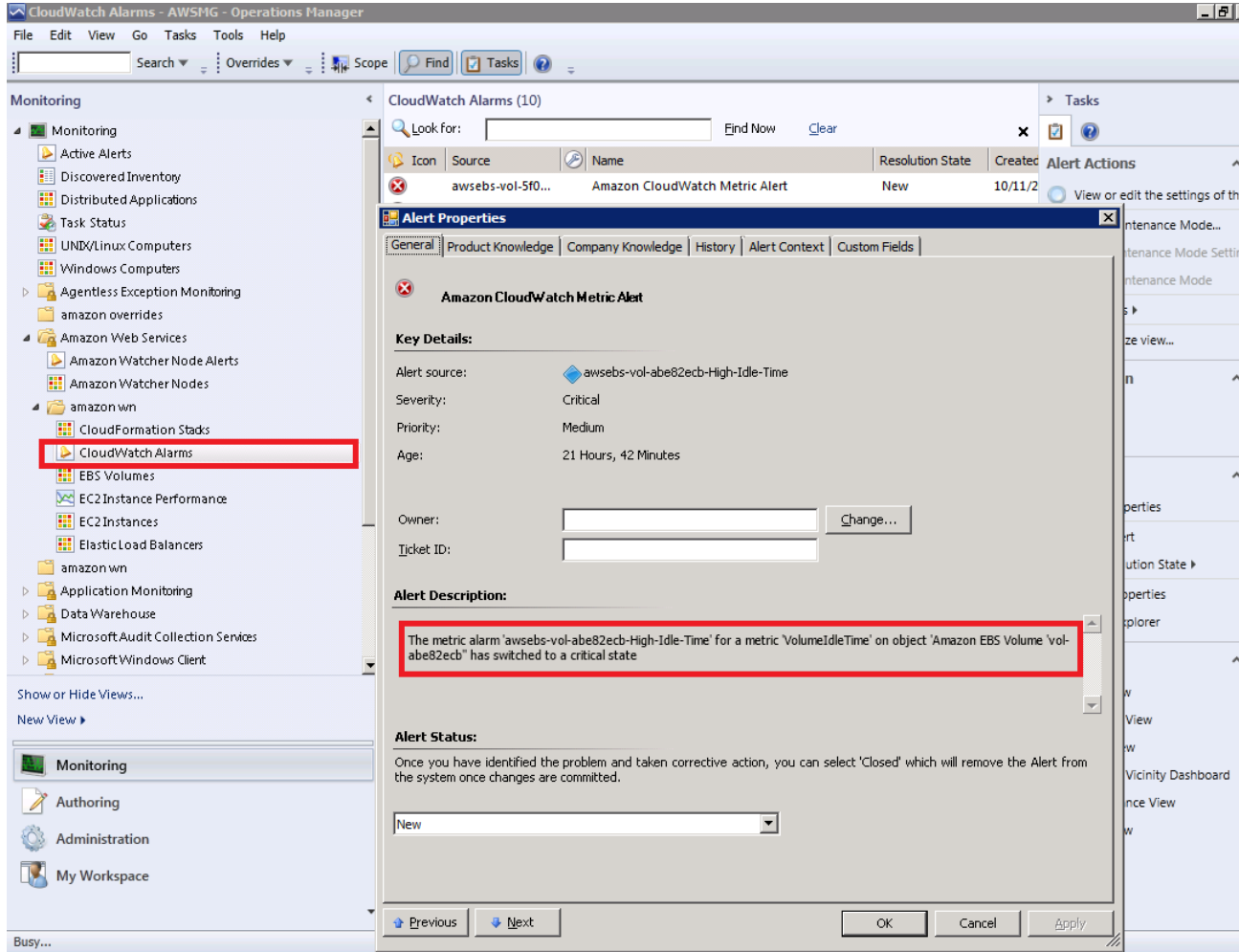
- **CloudFormation Stacks State View**

  Shows the health state of all the AWS CloudFormation stacks for a particular AWS account, from all regions.
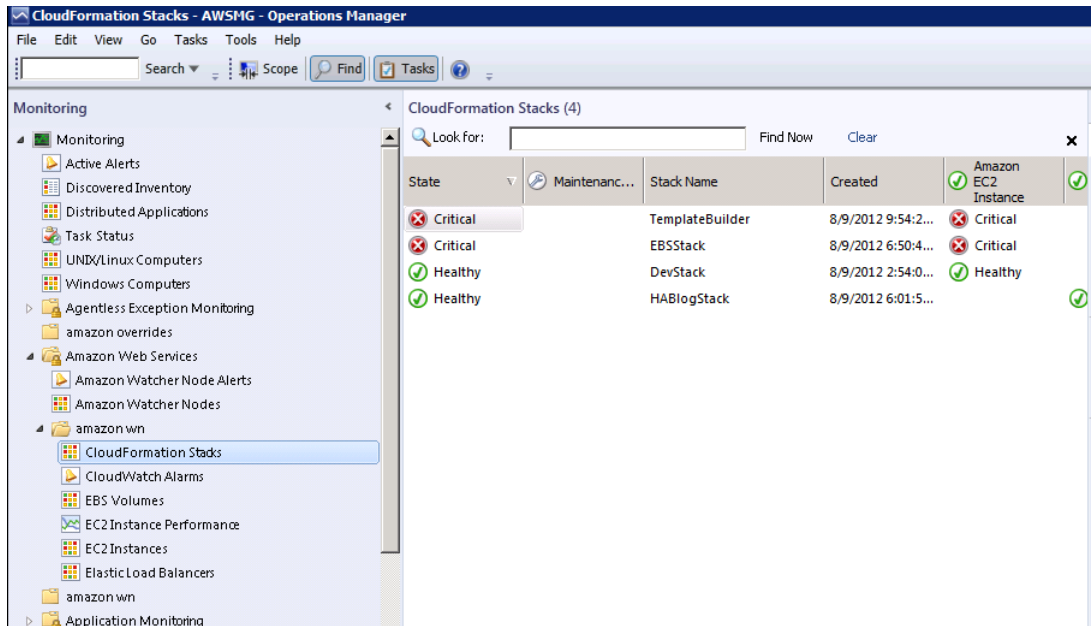
- **CloudFormation Stack Diagram View**

  Shows the AWS CloudFormation stack relationship with other components. An AWS CloudFormation stack may contain Amazon EC2 or Elastic Load Balancing resources. The following illustration shows an example:
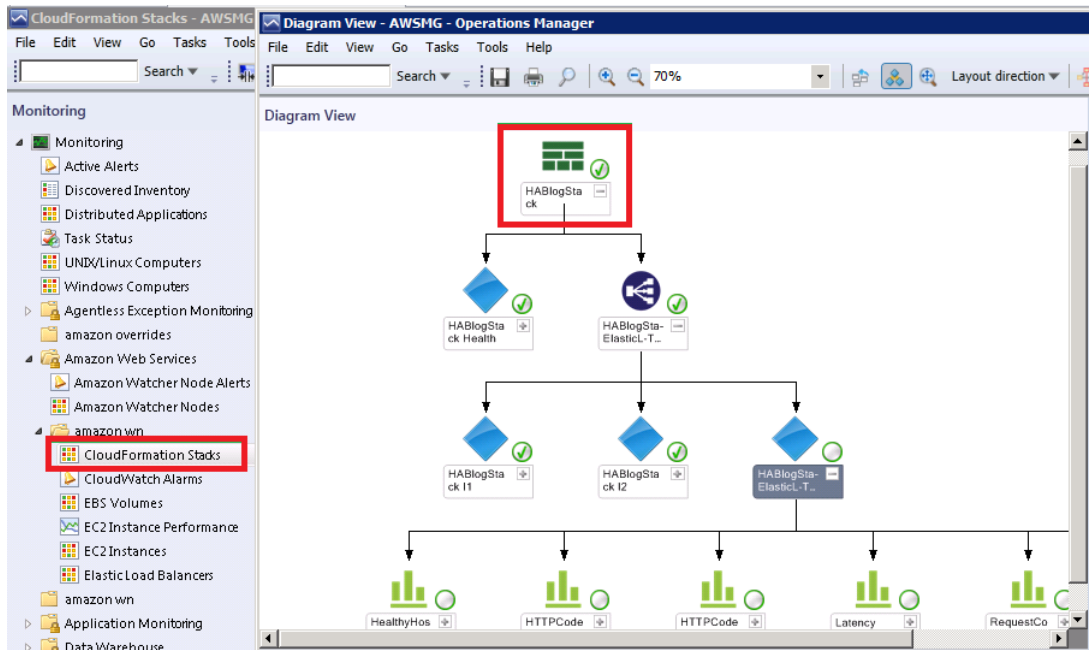
- **EBS Volumes State View**

  Shows the health state of all the Amazon EBS volumes for a particular AWS account, from all Availability Zones and regions.

- **EBS Volume Diagram View**

Shows an Amazon EBS volume and its default Amazon CloudWatch metrics. If an Amazon CloudWatch metric shows as Not Monitored, check to see if at least one Amazon CloudWatch alarm has been defined for that Amazon CloudWatch metric. The following illustration shows an example:

- **Elastic Load Balancers State View**

  Shows the health state of all the load balancers for a particular AWS account, from all regions.
- Elastic Load Balancer Diagram View

  Shows the Elastic Load Balancing relationship with other components. The following illustration shows an example:
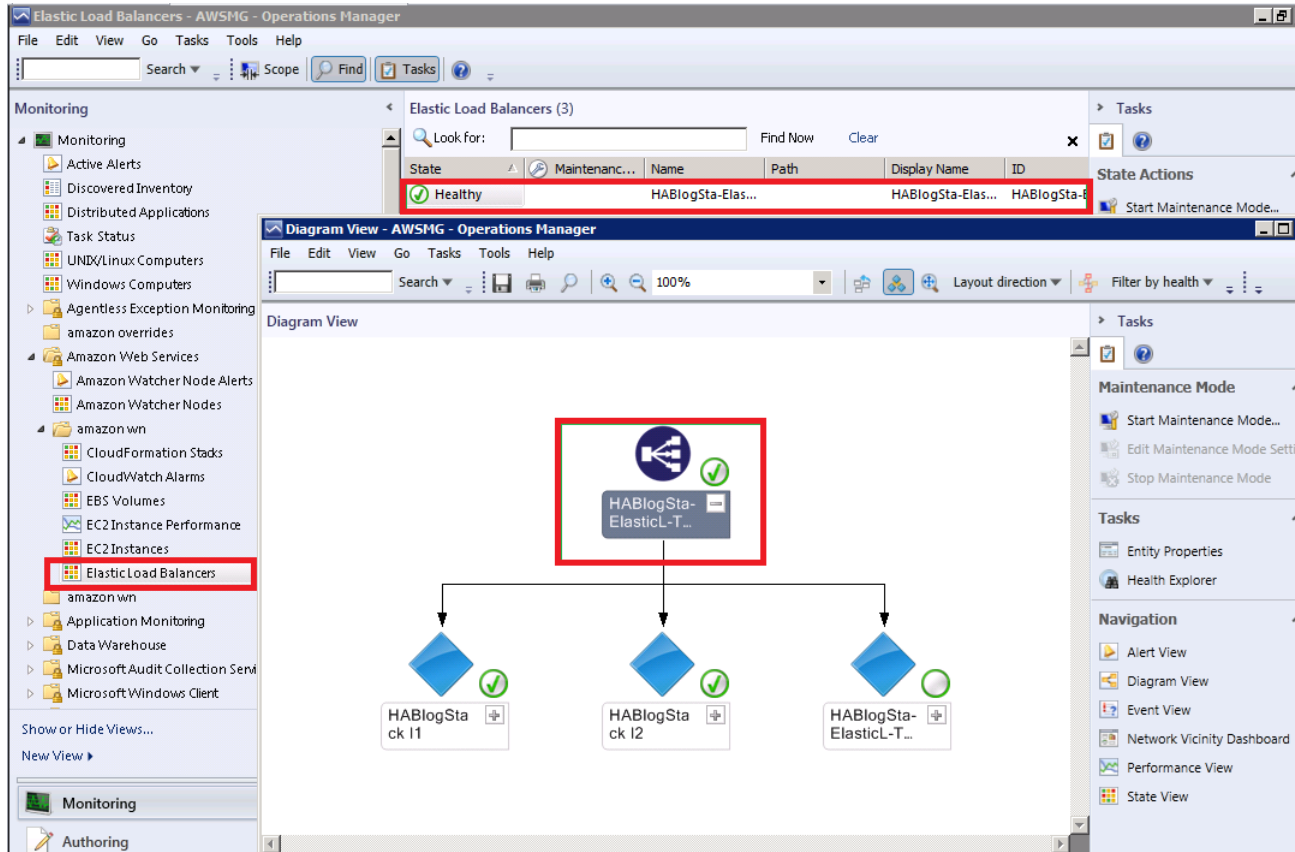
- **AWS Elastic Beanstalk Application State View**

  Shows state of all discovered AWS Elastic Beanstalk applications.
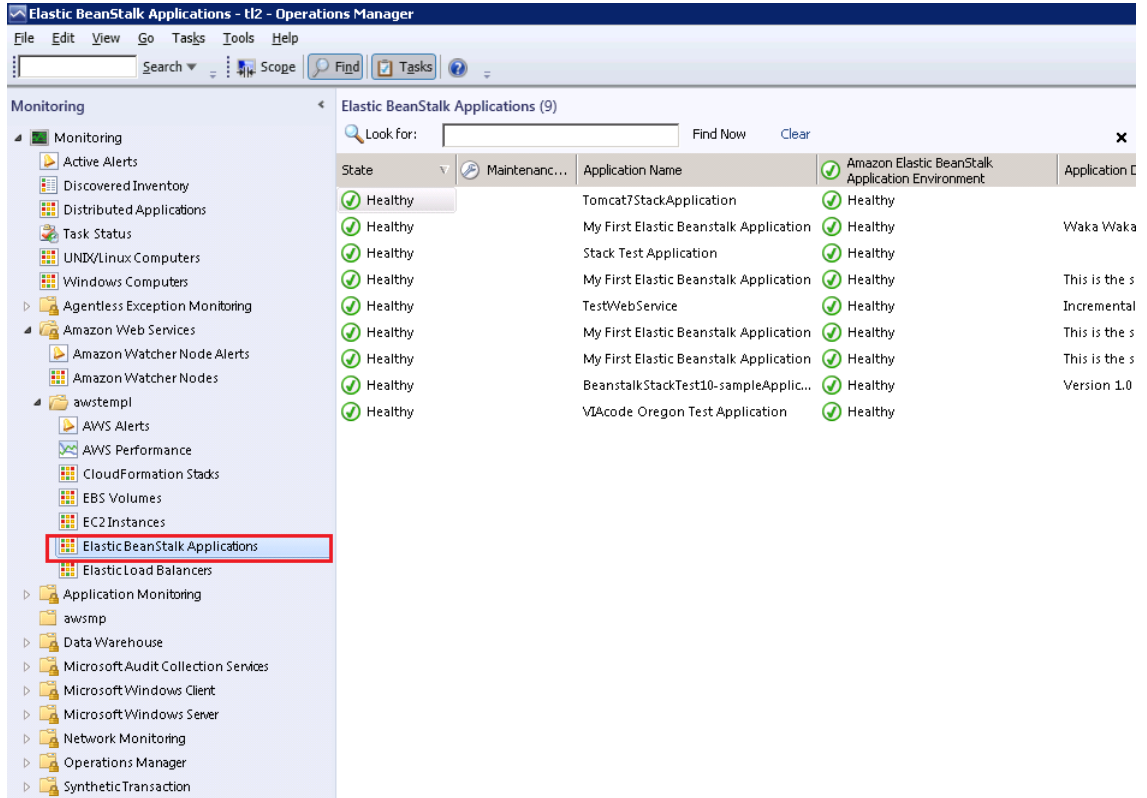
- **AWS Elastic Beanstalk Application Diagram View**

  Shows the AWS Elastic Beanstalk application, application environment, application configuration, and application resources objects.
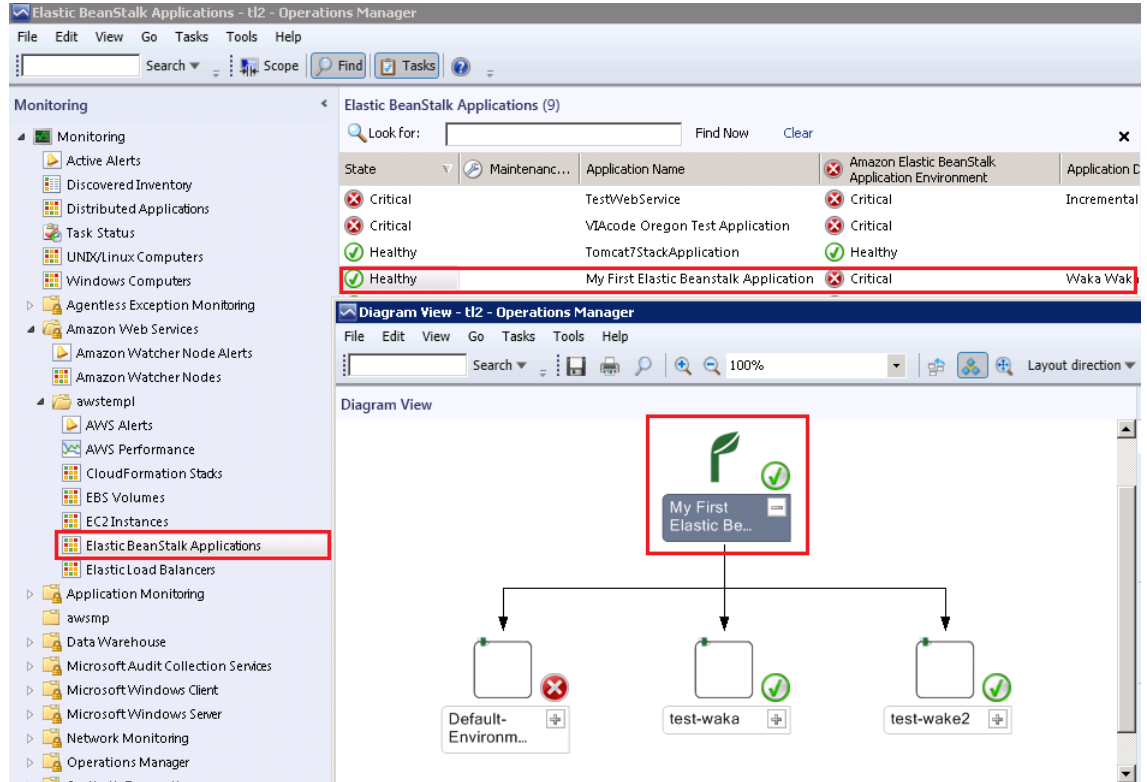
Tasks

You can use the AWS Management Pack to do many tasks with your Amazon EC2 instances.

Amazon EC2 Instance Tasks

When you select an Amazon EC2 instance in the EC2 Instance State View, you can perform instance health tasks.

- Connect to AWS Management Console: Launches the AWS Management Console in a web browser.
- Open RDP to Amazon EC2 Instance: Opens an RDP connection to the selected Amazon EC2 instance for Windows.

Amazon EC2 Instance Health Tasks:

The following tasks are available when you select an Amazon EC2 instance health entity in the diagram view:

- Reboot Amazon EC2 Instance: Remotely reboots the Amazon EC2 instance.
- Start Amazon EC2 Instance: Remotely starts the Amazon EC2 instance if it's stopped.
- Stop Amazon EC2 Instance: Remotely stops the Amazon EC2 instance if it's running.

Understanding the AWS Management Pack

The discoveries (objects and relationships) and health model of the AWS Management Pack are described in the following sections.

Discoveries

The AWS Management Pack discovers the following objects:

- Amazon EC2 instances
- Amazon EBS volumes
- Elastic Load Balancing
- AWS CloudFormation stacks
- Amazon CloudWatch metrics (the default metrics for the discovered Amazon EC2, Amazon EBS, and Elastic Load Balancing resources)
- Amazon CloudWatch alarms (defined for the discovered metrics)
- AWS Elastic Beanstalk applications
- Auto Scaling groups and Availability Zones

For Amazon CloudWatch metrics discovery, the following guidelines apply:

- Amazon CloudWatch metrics in the diagram views appear as Not Monitored if no Amazon CloudWatch alarms are defined for that metric.
- Only default Amazon CloudWatch metrics appear in Operations Manager. Custom Amazon CloudWatch metrics do not appear in Operations Manager.
- AWS CloudFormation stacks do not have any default Amazon CloudWatch metrics.
- Stopped Amazon EC2 instances or unused Amazon EBS volumes do not generate data for their default Amazon CloudWatch metrics.
- After starting an Amazon EC2 instance, it can take up to 30 minutes for the Amazon CloudWatch metrics to appear in Operations Manager.
- Amazon CloudWatch retains the monitoring data for two weeks, even if your AWS resources have been terminated. This data appears in Operations Manager.

The AWS Management Pack also discovers the following relationships:

- AWS CloudFormation stack and its Elastic Load Balancing or Amazon EC2 resources
- Elastic Load Balancing load balancer and its Amazon EC2 instances
- Amazon EC2 instance and its Amazon EBS volumes
- Amazon EC2 instance and its Windows/Linux operating system
- AWS Elastic Beanstalk application and its environment, configuration, and resources

The AWS Management Pack automatically discovers the relationship between an Amazon EC2 instance and the operating system running on it. To discover this relationship, the Operations Manager Agent must be installed and configured on the Amazon EC2 instance and the corresponding operating system management pack must be imported in Operations Manager.

For more information about these discoveries, the order in which they happen, and their default intervals, see .
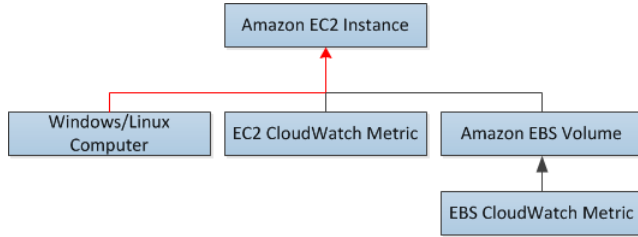
Health Model

The following illustration shows how the health states roll up in the AWS Management Pack.

Amazon EC2 Instance

EC2 CloudWatch Metric

Amazon EBS Volume

EC2 CloudWatch Alarm

EBS CloudWatch Metric

EBS CloudWatch Alarm
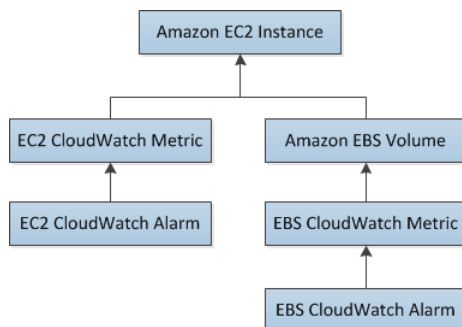
The health state for an Amazon CloudWatch alarm rolls up to the corresponding Amazon CloudWatch metric. So, the Amazon CloudWatch metrics for Amazon EC2 roll up their health state to the Amazon EC2 instance. Similarly, the Amazon CloudWatch metrics for Amazon EBS roll up their health state to the Amazon EBS volume. The Amazon EBS volumes used by an Amazon EC2 instance roll up their health state to the Amazon EC2 instance.

When the relationship between an Amazon EC2 instance and its operating system has been discovered, the operating system health state rolls up to the Amazon EC2 instance.

The health state of an AWS CloudFormation stack depends on the status of the AWS CloudFormation stack itself and the health states of its resources, namely the Elastic Load Balancing load balancers and Amazon EC2 instances.

The following table illustrates how the status of the AWS CloudFormation stack corresponds to its health state.

Health State	AWS CloudFormation Stack Status	Notes
Error	CREATE_FAILED DELETE_IN_PROGRESS DELETE_FAILED UPDATE_ROLLBACK_FAILED	Most likely usable
Warning	UPDATE_ROLLBACK_IN_PROGRESS UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS UPDATE_ROLLBACK_COMPLETE	Recovering after some problem
Healthy	CREATE_COMPLETE UPDATE_IN_PROGRESS UPDATE_COMPLETE_CLEANUP_IN_PROGRESS UPDATE_COMPLETE	Usable

The full health roll up model for an AWS CloudFormation stack is as follows:
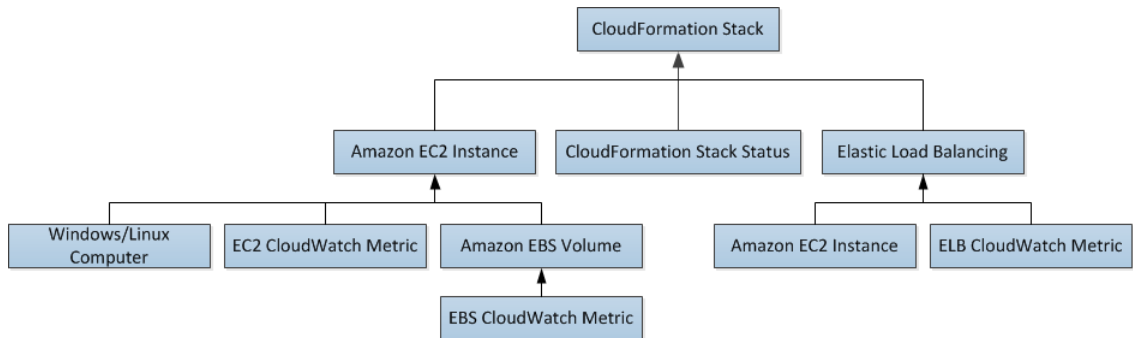
Customizing the AWS Management Pack

This section shows how you can customize the AWS Management Pack.

You can configure the time intervals for discoveries, monitors, and rules implemented in the AWS Management Pack. For more information about the default time intervals, see Discoveries, Monitors, Rules, and Events (p. 86). You can customize the AWS Management Pack by overriding the defaults in the Authoring workspace of the Operations Console.

For more information on how to create overrides, see Tuning Monitoring by Using Targeting and Overrides at the *Microsoft TechNet* website.

For more information on how to create custom rules and monitors, see Authoring for System Center 2012 - Operations Manager or System Center Operations Manager 2007 R2 Management Pack Authoring Guide at the *Microsoft TechNet* website.

Troubleshooting the AWS Management Pack

This section lists some troubleshooting tips that you might find helpful.

- Ensure that you have installed the latest Update Roll up for System Center 2012 – Operations Manager.
- The AWS Management Pack requires at least Update Roll up 1.
- Ensure that you have configured the AWS Management Pack after importing it, by running the Add Monitoring Wizard. For more information, see Step 1: Installing the AWS Management Pack (p. 67).
- Ensure that you give enough time for the AWS resources to be discovered (10-20 minutes).
- Ensure that the watcher node is configured properly.
  - The proxy agent is enabled. For more information, see Step 2: Configuring the Watcher Node (p. 68).
  - The watcher node has Internet connectivity.
  - The action account for the watcher node has local administrator privileges.
  - The watcher node must have .NET framework 3.5.1. or newer installed
- Ensure that the watcher node is healthy and resolve all alerts. For more information, see Views (p. 73).
- Ensure that the AWS Run As account is valid.
  - The values for the Access Key ID and the Secret Access Key are accurate.
  - The Access Key is active (check the My Account -> Security Credentials page of the AWS Management Console).
  - The IAM user has at least read-only access permission.
    - If an Amazon CloudWatch metric shows as Not Monitored, check whether at least one Amazon CloudWatch alarm has been defined for that Amazon CloudWatch metric.
    - For further troubleshooting, use the information from the event logs.
    - Check the Operations Manager event log on the management server as well as the watcher node. For more information, see Events (p. 90) for a list of all the events that the AWS Management Pack writes to the Operations Manager event log.

Discoveries, Monitors, Rules, and Events

This topic covers the discoveries, monitors, and rules implemented by the AWS Management Pack, along with the list of events that it writes to the Operations Manager event log on the management server and the watcher node.

Discoveries

Discoveries are the AWS resources that are monitored by the AWS Management Pack.

Discovery	Runs On	Interval (seconds)
Watcher Node Discovery Targets the root management server and creates the watcher node objects.	Management server	14400
UNIX and Windows Computer Discovery Finds UNIX and Windows computers that are running on Amazon EC2 instances. As a result, a simple URL-querying script is executed on the computers to identify the Amazon EC2 instance ID that can be used for linking Amazon EC2 instance objects to Windows and UNIX computers. This discovery populates the properties of the AmazonComputerLink objects.	UNIX / Windows computer	14400
Amazon EC2 Instance to Windows or UNIX Computer Relation Discovery Discovers the relationship between the Amazon EC2 instance and the Windows or UNIX computer.	Management server	14400

Discovery	Runs On	Interval (seconds)
AWS Elastic Beanstalk Discovery Discovers AWS Elastic Beanstalk and its relationship with environment, resources, and configuration.	Watcher node	14400

Monitors

Monitors are used to measure the health of your AWS resources.

Monitor	Runs On	Interval (seconds)
AWS CloudFormation Stack Status	Watcher node	900
Amazon CloudWatch Metric Alarm	Watcher node	900
Amazon EBS Volume Status	Watcher node	900
Amazon EC2 Instance Status	Watcher node	900
Amazon EC2 Instance System Status	Watcher node	900
Watcher Node to Amazon Cloud Connectivity	Watcher node	900

Rules

Rules create alerts (based on Amazon CloudWatch metrics) and collect data for analysis and reporting.

Rule	Runs On	Interval (seconds)
AWS Resource Discovery Rule Targets the watcher node and uses the AWS API to discover objects for following AWS resources: Amazon EC2 instances, Amazon Elastic Block Store volumes, Elastic Load Balancing, and AWS CloudFormation stacks. This discovery does not include discovery of Amazon CloudWatch metrics or alarms. After this discovery is complete, you see the objects for AWS resources in the Not Monitored state.	Watcher node	14400
Amazon CloudWatch Metrics and Alarms Discovery Rule Targets the objects for already discovered AWS resources and discovers the default Amazon CloudWatch metrics and alarms, if any, associated with those metrics.	Watcher node	14400
Amazon Elastic Block Store Volume Performance Metrics Data Collection Rule	Watcher node	900

Rule	Runs On	Interval (seconds)
Amazon EC2 Instance Performance Metrics Data Collection Rule	Watcher node	900
Elastic Load Balancing Balancing Performance Metrics Data Collection Rule	Watcher node	900

Events

Events report on activities that involve the monitored resources. Events are written to the Operations Manager event log.

Event ID	Description
4101	Amazon EC2 Instance Discovery (General Discovery) finished
4102	Elastic Load Balancing Metrics Discovery, Amazon EBS Volume Metrics Discovery, Amazon EC2 Instance Metrics Discovery finished
4103	Amazon CloudWatch Metric Alarms Discovery finished
4104	Amazon Windows Computer Discovery finished
4105	Collecting Amazon Metrics Alarm finished
4106	EC2 Instance Computer Relation Discovery finished
4107	Collecting AWS CloudFormation Stack State finished
4108	Collecting Watcher Node Availability State finished
4109	Amazon Metrics Collection Rule finished
4110	Task to change Amazon Instance State finished
4111	EC2 Instance Status Monitor State finished
4112	Amazon EBS Volume Status Monitor State finished
4113	Amazon EC2 Instance Scheduled Events Monitor State calculated
4114	Amazon EBS Scheduled Events Monitor State calculated
4115	AWS Elastic Beanstalk Discovery finished
4116	AWS Elastic Beanstalk Environment Status State calculated
4117	AWS Elastic Beanstalk Environment Operational State calculated

Event ID	Description
4118	AWS Elastic Beanstalk Environment Configuration State calculated

Configuring a Secondary Private IP Address for Your Windows Instance in a VPC

In EC2-VPC, you can specify multiple private IP addresses for your instances. After you assign a secondary private IP address to an instance in a VPC, you must configure the operating system on the instance to recognize the secondary private IP address.

Configuring the operating system on a Windows instance to recognize a secondary private IP address requires the following:

- Step 1: Configure Static IP Addressing on Your Windows Instance (p. 93)
- Step 2: Configure a Secondary Private IP Address for Your Windows Instance (p. 94)
- Step 3: Configure Applications to Use the Secondary Private IP Address (p. 95)

> **Note**
> These instructions are based on Windows Server 2008 R2. The implementation of these steps may vary based on the operating system of the Windows instance.

Prerequisites

- As a best practice, launch your Windows instances using the latest AMIs. If you are using an older Windows AMI, ensure that it has the Microsoft hot fix referenced in http://support.microsoft.com/kb/2582281.
- After you launch your instance in your VPC, add a secondary private IP address. For more information, see Multiple IP Addresses in the *Amazon Elastic Compute Cloud User Guide*.
- To allow Internet requests to your website after you complete the tasks in these steps, you must configure an Elastic IP address and associate it with the secondary private IP address. For more information, see Assigning a Elastic IP Address to the Secondary Private IP Address in the *Amazon Elastic Compute Cloud User Guide*.

Step 1: Configure Static IP Addressing on Your Windows Instance

To enable your Windows instance to use multiple IP addresses, you must configure your instance to use static IP addressing rather than a DHCP server.

> **Important**
> When you configure static IP addressing on your instance, the IP address must match exactly what you have assigned it as shown in the AWS console, CLI, or API. If you enter these IP addresses incorrectly, the instance could become unreachable.

You'll lose RDP connectivity to the Windows instance for a few seconds while the instance converts from using DHCP to static addressing. The instance retains the same IP address information as before, but now this information is static and not managed by DHCP.

To configure static IP addressing on a Windows instance

1. Connect to your instance.
2. Click **Start**, and then click **Control Panel**.
3. Click **Network and Internet**, and then click **Network and Sharing Center**.
4. Click the network interface (Local Area Connection).
5. Click **Properties**.
6. Click **Internet Protocol Version 4 (TCP/IPv4)**, and then click **Properties**.
7. In the **Properties** dialog box, click **Use the following IP address**.
8. Click **Start**. In the **Search** box, type `cmd`, and then press **Enter**. This opens the Command Prompt window.
9. At the command prompt, run the following command: **ipconfig /all**.
10. Note the current IPv4 address, default gateway, and DNS server for the network interface.
11. In the **Internet Protocol Version 4 (TCP/IPv4) Properties** dialog box, under **Use the following IP address**, in the **IP address** box, type the IPv4 address shown in the Command Prompt window.
12. In the **Subnet mask** box, type the subnet mask shown in the Command Prompt window.
13. In the **Default gateway** box, type the IP address of the default gateway shown in the Command Prompt window, and then click **OK**.
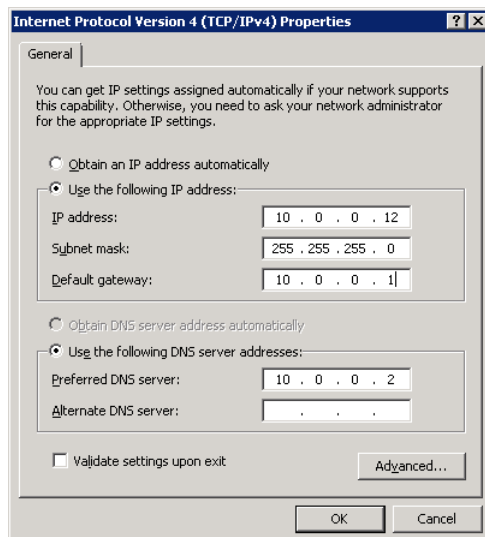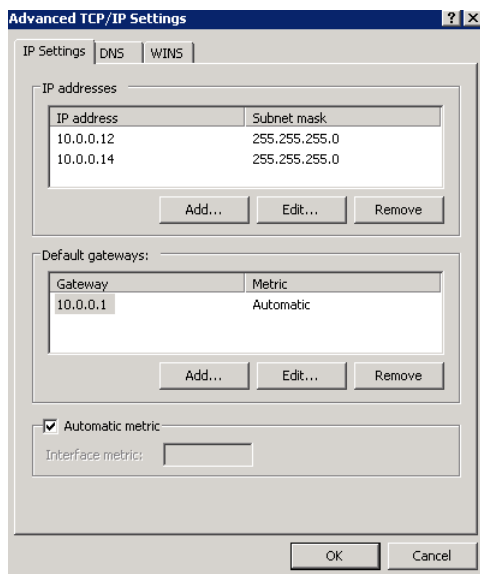
Step 2: Configure a Secondary Private IP Address for Your Windows Instance

To configure a secondary IP address for a Windows instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, click **Instances**.
3. Select your instance.
4. On the **Description** tab in the lower pane, note the secondary IP address.
5. Connect to your instance.
6. On your Windows instance, click **Start**, and then click **Control Panel**.
7. Click **Network and Internet**, and then click **Network and Sharing Center**.
8. Click the network interface (Local Area Connection).
9. Click **Properties**.
10. In the **Local Area Connection Properties** page, click **Internet Protocol Version 4 (TCP/IPv4)**, click **Properties**, and then click **Advanced**.
11. Click **Add**.
12. In the **TCP/IP Address** dialog box, type the secondary private IP address in the **IP address** box. In the **Subnet mask** box, type the same subnet mask that you entered for the primary private IP address in Step 1: Configure Static IP Addressing on Your Windows Instance (p. 93), and then click **Add**.
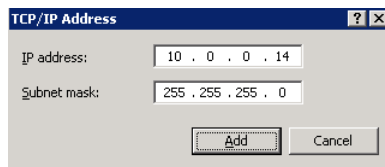
13. Verify the IP address settings, and then click **OK**.

14. Click **OK** again, and then click **Close**.

15. To confirm that the secondary IP address has been added to the operating system, at a command prompt, run the command **ipconfig /all**.

Step 3: Configure Applications to Use the Secondary Private IP Address

You can configure any applications to use the secondary private IP address. For example, if your instance is running a website on IIS, you can configure IIS to use the secondary private IP address.

To configure IIS to use the secondary private IP address

1. Connect to your instance.
2. Open Internet Information Services (IIS) Manager.
3. In the **Connections** pane, expand **Sites**.
4. Right-click your website, and then click **Edit Bindings**.
5. In the **Site Bindings** dialog box, under **Type**, click **http**, and then click **Edit**.
6. In the **Edit Site Binding** dialog box, in the **IP address** box, click the secondary private IP address. (By default, each website accepts HTTP requests from all IP addresses.)

7. Click **OK**, and then click **Close**.

Setting Up a Windows HPC Cluster on Amazon EC2

This section steps you through how to launch a scalable Microsoft Windows High Performance Computing (HPC) cluster using only Amazon Elastic Compute Cloud (Amazon EC2) instances. A Windows HPC cluster requires an Active Directory domain controller and a DNS server, a head node, and one or more compute nodes. By following the steps in this section, you can assemble each of these components and launch a Windows HPC cluster. For more information on High Performance Computing, see 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 AWS account, if you haven't already.
- 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 Interface Tools on Windows (p. 106).
- 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. 103), 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, see 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. 104), 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
```

Installing the Amazon EC2 Command Line Interface Tools on Windows

The Amazon EC2 command line interface tools (also called the *CLI tools*) wrap the Amazon EC2 API actions. These tools are written in Java and include shell scripts for both Windows and Linux/UNIX/Mac OSX.

Note

Alternatively, you can use the AWS Command Line Interface (CLI), which provides commands for a broad set of AWS products, including Amazon EC2. To get started, see AWS Command Line Interface User Guide. For more information about the AWS CLI commands for Amazon EC2, see ec2.

Before you can use the Amazon EC2 CLI tools, you need to download them and configure them to use your AWS account. You can set up the tools on your own computer or on an Amazon EC2 instance.

Complete the following tasks to set up your Amazon EC2 environment:

1. Download the CLI Tools (p. 107)

2. Set the JAVA_HOME Environment Variable (p. 107)

3. Set the EC_HOME Environment Variable (p. 108)

4. Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables (p. 109)

5. (Optional) Set the Region (p. 110)

6. (Optional) Use a Proxy (p. 110)

7. Download Remote Desktop (p. 111)

Note

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

Task 1: Download the Command Line Interface Tools (CLI Tools)

The CLI tools are available as a ZIP file on this site: Amazon EC2 CLI Tools. The tools are written in Java and include shell scripts for both Windows and Linux/UNIX/Mac OSX. The ZIP file is self-contained; no installation is required. You can simply download the file and unzip it.

Task 2: Set the JAVA_HOME Environment Variable

The Amazon EC2 CLI tools require Java. They read the JAVA_HOME environment variable to locate the Java runtime. This environment variable should specify the full path of the directory that contains a subdirectory named bin that contains the Java executable you installed (java.exe).

To set the JAVA_HOME environment variable on your computer or instance

1. If you don't have Java 1.6 or later installed, download and install Java. Either a JRE or JDK installation is acceptable. To view and download JREs for a range of platforms, see Free Java Download.

2. Set JAVA_HOME to the full path of the Java home directory. For example, if your Java executable is in C:\Program Files (x86)\Java\jre7\bin, set JAVA_HOME to C:\Program Files (x86)\Java\jre7.

   Important
   These steps don't update the environment variables in your current Command Prompt windows. The Command Prompt windows that you open after you complete these steps will contain the updates. This is why it's necessary for you to open a new Command Prompt window to verify that your environment is set up properly.

   a. Click **Start**, right-click **Computer**, and then click **Properties**.

   b. Click **Advanced system settings**.

   c. Click **Environment Variables**.

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

   e. In **Variable name**, type **JAVA_HOME**.

   f. In **Variable value**, type the path to your Java home directory (for example, C:\Program Files (x86)\Java\jre7).

      Important
      Don't include the bin directory in JAVA_HOME. This is a common mistake, but the CLI tools won't work if you do this.

   g. Click **OK**.

3. Open a new Command Prompt window and verify your JAVA_HOME setting using this command.

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

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

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

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

4. Add the bin directory that contains the Java executable 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**.

5. Open a new Command Prompt window and verify your update to the Path environment variable using this command.

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

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

Task 3: Set the EC2_HOME Environment Variable

The Amazon EC2 CLI tools read the EC2_HOME environment variable to locate supporting libraries. You'll need to set this environment variable to the path where you unzipped the CLI tools. This directory is named ec2-api-tools-*w.x.y.z* (where *w*, *x*, *y*, and *z* are components of the version number). It contains sub-directories named bin and lib.

To set the EC2_HOME environment variable on your computer or instance

1. Set EC2_HOME to the path of the directory into which you unzipped the CLI tools.

    Important
    These steps don't update the environment variables in your current Command Prompt windows. The Command Prompt windows that you open after you complete these steps will contain the updates. This is why it's necessary for you to open a new Command Prompt window to verify that your environment is set up properly.

    a. Click **Start**, right-click **Computer**, and then click **Properties**.

    b. Click **Advanced system settings**.

    c. Click **Environment Variables**.

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

    e. In **Variable name**, type **EC2_HOME**.

    f. In **Variable value**, type the path to the directory where you installed the CLI tools. For example, **C:\AWS\EC2\ec2-api-tools-1.6.7.2**.

2. Open a new Command Prompt window and verify your EC2_HOME setting using this command.

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

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

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

   You can update your `Path` as follows:

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

   b. In **Variable values**, add `;%EC2_HOME%\bin`.

Task 4: Set the AWS_ACCESS_KEY and AWS_SECRET_KEY Environment Variables

Your access keys identify you to the Amazon EC2 CLI tools. There are two types of access keys: access key IDs and secret access keys. You should have stored your access keys in a safe place when you created them. Although you can retrieve your access key ID from the Your Security Credentials page, you can't retrieve your secret access key. Therefore, if you can't find your secret access key, you'll need to create new access keys before you can use the CLI tools.

Every time you issue a command, you must specify your access keys using the **--aws-access-key** and **--aws-secret-key** (or **-O** and **-W**) options. Alternatively, you might find it easier to store your access keys using the following environment variables:

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

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

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

To set up your environment variables on your computer or instance

1. Click **Start**, right-click **Computer**, and then click **Properties**.
2. Click **Advanced system settings**.
3. Click **Environment Variables**.
4. Under **System variables**, click **New**.
5. In **Variable name**, type `AWS_ACCESS_KEY`.
6. In **Variable value**, specify your access key ID.
7. Under **System variables**, click **New**.
8. In **Variable name**, type `AWS_SECRET_KEY`.
9. In **Variable value**, specify your secret access key.

To verify that all your environment variables are set up correctly, open a new Command Prompt window and run the following command.

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

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

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

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

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

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

Task 5: Set the Region (Optional)

By default, the Amazon EC2 CLI tools use the `us-east-1` region with the `ec2.us-east-1.amazonaws.com` service endpoint URL. If your instances are in a different region, you must specify the region where your instances reside. For example, if your instances are in Europe, you must specify the eu-west-1 region by using the `--region eu-west-1` option or by setting the `EC2_URL` environment variable.

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

To specify a different region on your computer or instance

1. To view available regions, see Regions and Endpoints in the *Amazon Web Services General Reference*.

2. To change the service endpoint, set the `EC2_URL` environment variable.

   The following example sets `EC2_URL`.

   a. Click **Start**, right-click **Computer**, and then click **Properties**.

   b. Click **Advanced system settings**.

   c. Click **Environment Variables**.

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

   e. In **Variable name**, type **EC2_URL**.

   f. In **Variable value**, type **https://ec2.eu-west-1.amazonaws.com**.

Task 6: Use a Proxy (Optional)

If the computer you have installed the CLI tools on requires the use of a proxy server, you must tell the CLI tools to use the proxy server with the `EC2_JVM_ARGS` environment variable.

The following table contains the proxy configuration properties that can be set for the `EC2_JVM_ARGS` variable. The properties that are required will depend on the type of proxy server being used. For example, the `http.proxyDomain` and `http.proxyWorkstation` properties are only used with a Windows NTLM proxy.

Property	Description
`https.proxyHost`	HTTPS proxy host. Use when `EC2_URL` specifies an HTTPS host.
`https.proxyPort`	HTTPS proxy port. Use when `EC2_URL` specifies an HTTPS host.
`http.proxyHost`	HTTP proxy host. Use when `EC2_URL` specifies an HTTP host.
`http.proxyPort`	HTTP proxy port. Use when `EC2_URL` specifies an HTTP host.
`http.proxyDomain`	Proxy domain (HTTPS and HTTP)
`http.proxyWorkstation`	Proxy workstation (HTTPS and HTTP)
`http.proxyUser`	Proxy user name (HTTPS and HTTP)
`http.proxyPass`	Proxy password (HTTPS and HTTP)
`http.nonProxyHosts`	A list of hosts that should be reached directly, bypassing the proxy. Each item in the list is separated by '|'.

To set up the `EC2_JVM_ARGS` environment variable on your computer or instance

1. Click **Start**, right-click **Computer**, and then click **Properties**.
2. Click **Advanced system settings**.
3. Click **Environment Variables**.
4. Under **System variables**, click **New**.
5. In **Variable name**, type `EC2_JVM_ARGS`.
6. In **Variable value**, specify the proxy configuration properties. For example, `"-Dhttps.proxyHost=my.proxy.com -Dhttps.proxyPort=8080"`.

Task 7: Download Remote Desktop

To connect to a Windows instance, you'll need a Remote Desktop client. The most recent versions of Windows include a Remote Desktop client already. To check whether you have one, open a Command Prompt window and type **mstsc**. If this command displays the Remote Desktop Connection window, you're set. Otherwise, go to the Microsoft Windows home page and search for the download for Remote Desktop Connection.

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

AWS Diagnostics for Microsoft Windows Server

AWS Diagnostics for Microsoft Windows Server is a simple and easy to use tool that can be run on an Amazon EC2 Windows Server instance to diagnose and troubleshoot possible issues. It is a very valuable tool not just for collecting log files and troubleshooting issues, but also proactively searching for possible areas of concern. This tool can, for example, be used to diagnose configuration issues between the Windows Firewall and the AWS security groups that may affect your applications. It can even examine EBS boot volumes from other instances and collect relevant logs for troubleshooting Windows Server instances on that volume.

One use case is diagnosing problems with Key Management Service (KMS) activations. KMS activation can fail if you have changed the DNS server, added instances to a domain, or if the server time is out of sync. In this case, instead of trying to examine your configuration settings manually and debugging the issue, run the AWS Diagnostics for Microsoft Windows Server tool to give you all the information you need on possible issues.

Another use case is a difference between the rules in an Amazon EC2 security group and the Windows Firewall. If you provide your AWS user credentials to describe your security groups, the AWS Diagnostics for Microsoft Windows Server tool is able verify if the ports listed in a security group are allowed through the Windows Firewall. You will not have to look at firewall rules manually and verify them against security group rules.

The AWS Diagnostics for Microsoft Windows Server tool is available free of charge and can be downloaded and installed from
https://s3.amazonaws.com/ec2-downloads-windows/AWSDiagnostics/AWSDiagnostics.zip.

AWS Diagnostics for Microsoft Windows Server is comprised of two different modules: a data collector module that collects data from all different sources, and an analyzer module that parses the data collected against a series of predefined rules to identify issues and provide suggestions.

The AWS Diagnostics for Microsoft Windows Server tool can only be run on Windows Server running in an EC2 instance. When the tool starts, it checks to determine if it is running in an EC2 instance. If the check fails, the tool displays the `EC2InstanceCheckFailed` error message in a message box.

Analysis Rules

AWS Diagnostics for Microsoft Windows Server provides the following analysis rules:

- Check for activation status and KMS settings
- Check for proper route table entries for meta data and KMS access
- Comparison of Amazon EC2 security group rules against Windows Firewall rules
- Check the version of the PV driver (RedHat or Citrix)
- Check if the `RealTimeIsUniversal` registry key is set
- Check the default gateway settings if using multiple NICs
- Bug check code in mini dump files

Even if the analyzer doesn't report any problems, the data collected by the tool may still be useful. You can view the data files created by the tool to look for problems, or provide these files to AWS Premium Support to help resolve a support case.

Analyzing the Current Instance

To analyze the current instance, run the AWS Diagnostics for Microsoft Windows Server tool, and select **Current Instance** for the type of instance. In the **Data to Collect** section of the main window, you specify the data that AWS Diagnostics for Microsoft Windows Server collects.
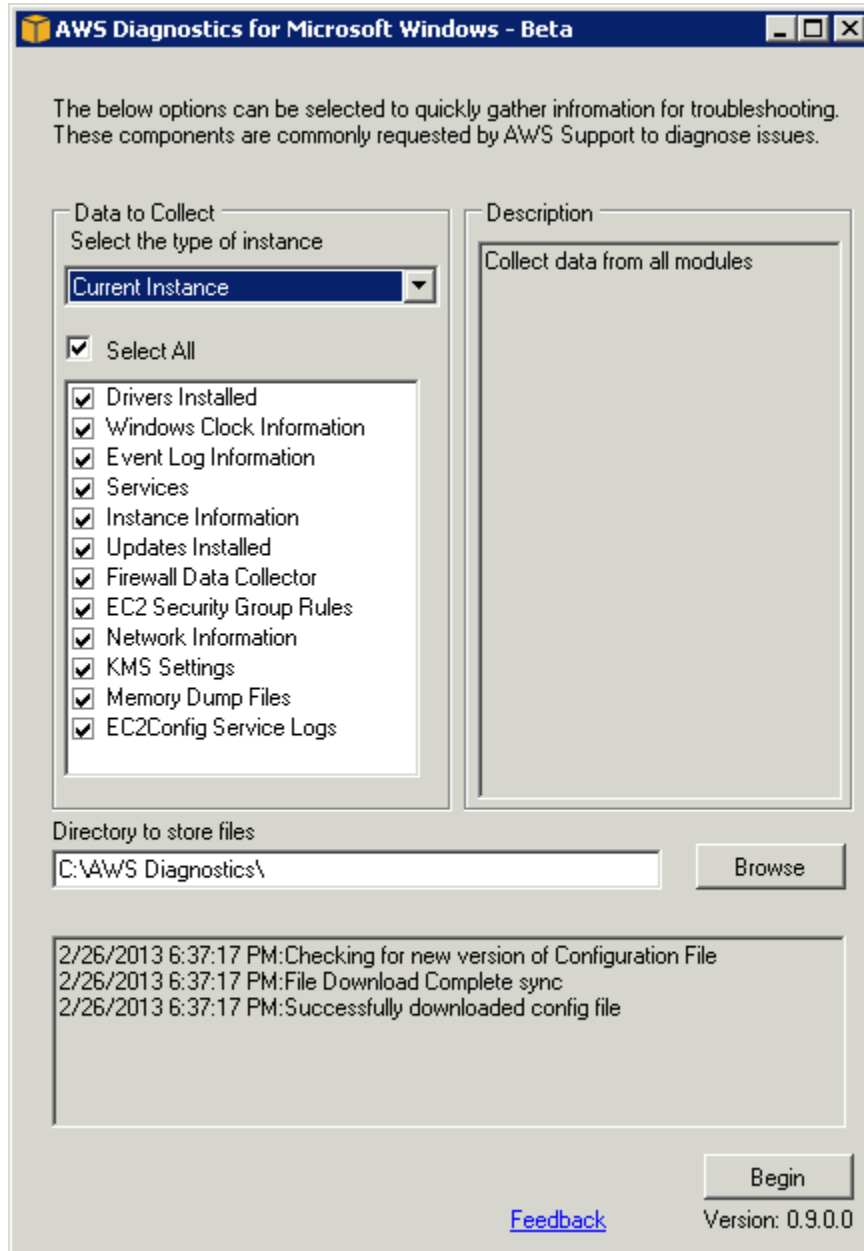
Data	Description
Drivers Installed	Collects information about all of the drivers installed on the instance.
Windows Clock Information	Collects current time and time zone information for the instance.
Event Log Information	Collects critical, error, and warning messages from the event logs.
Services	Collects information about the services that are installed on the instance.

Data	Description
Instance Information	Collects information from metadata and local environment variables.
Updates Installed	Collects information about the updates that are installed on the instance.
Firewall Data Collector	Collects information about the Windows Firewall settings.
EC2 Security Group Rules	Collects information about the rules in the Amazon EC2 security groups associated with the instance.
Network Information	Collects route table and IP address information for the instance.
KMS Settings	Collects Key Management Service settings.
Memory Dump Files	Collects any memory dump files that exist on the instance.
EC2Config Service Logs	Collects log files generated by the EC2Config service.

Collecting Data From an Offline Instance

The **Offline Instance** option is useful when you want to debug an issue with an EC2 Windows Server instance that is either unable to boot up or is impaired such that you are not able to run the AWS Diagnostics for Microsoft Windows Server tool on it. In such a case, you can detach the EBS boot volume from that instance and attach the EBS volume to another EC2 Windows Server instance.

To collect data from an offline instance

1. Stop the faulty instance, if it is not stopped already.
2. Detach the EBS boot volume from the faulty instance.
3. Attach the EBS boot volume to another working Windows Server instance that has AWS Diagnostics for Microsoft Windows Server installed on it.
4. Mount the volume in the working instance, assigning it a drive letter (e.g., "F:").
5. Run the AWS Diagnostics for Microsoft Windows Server tool on the working instance and select the **Offline Instance** option.
6. Choose the drive letter of the newly-mounted volume (e.g., "F:").
7. Click **Begin**.

The AWS Diagnostics for Microsoft Windows Server tool scans the volume and collects troubleshooting information based on the log files that are on the volume. For offline instances, the data collected is a fixed set, and no analysis of the data is performed.

Data File Storage

By default, the AWS Diagnostics for Microsoft Windows Server tool places its data files under the directory that the tool is launched from. You can choose where to save the data files that are collected by the AWS Diagnostics for Microsoft Windows Server tool. Within the chosen directory, a directory named

`DataCollected` is created, if it doesn't already exist. A separate directory with the current date and time stamp is created for each run of the application. Each data collection module produces an XML file that contains information for that data set. A ZIP file archive is also created that contains copies of all of the data files generated. This archive can be provided to a AWS Premium support engineer if needed.

Upgrading Your PV Drivers on Your Windows AMI

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.

If your Windows AMI uses RedHat drivers, you can upgrade to Citrix drivers, or, if you are already using Citrix drivers, you can upgrade the Citrix paravirtualized (PV) guest agent driver.

Topics

Upgrading Your Windows Server 2003 Instance

This section explains how to upgrade the RedHat drivers to Citrix drivers on your Windows Server 2003 instance.

Before you start upgrading your drivers, make sure you do the following:

- Back up your important information on the instance, or create an AMI from the instance. For more information about creating an AMI, see Creating an Amazon EBS-Backed Windows AMI (p. 51). If you create an AMI, make sure you do the following:
  - Do not enable the Sysprep tool in the EC2Config service.
  - Write down your password.
  - Set your ethernet adapter to DHCP.
- Install the latest version of EC2Config by going to Amazon Windows EC2Config Service. For more information about the EC2Config service, see Configuring a Windows Instance Using the EC2Config Service (p. 40).

To upgrade a Windows Server 2003 AMI

1. Connect to your instance and log in as the local administrator. For more information about connecting to your instance, see Connecting to Windows Instances.

2. In your instance, download the Citrix upgrade package by going to Amazon EC2 Windows Paravirtual Driver Upgrade Script.

3. Extract the contents of the upgrade package to a location of your choice.

4. Double-click the **Upgrade.bat** file. If you get a security warning, click **Run**.

5. In the **Upgrade Drivers** dialog box, review the information and click **Yes** if you're ready to start the upgrade.

6. In the **Red Hat Paravirtualized Xen Drivers for Windows (R) uninstaller** dialog box, click **Yes** to remove the RedHat software. Your instance will reboot.

    Note
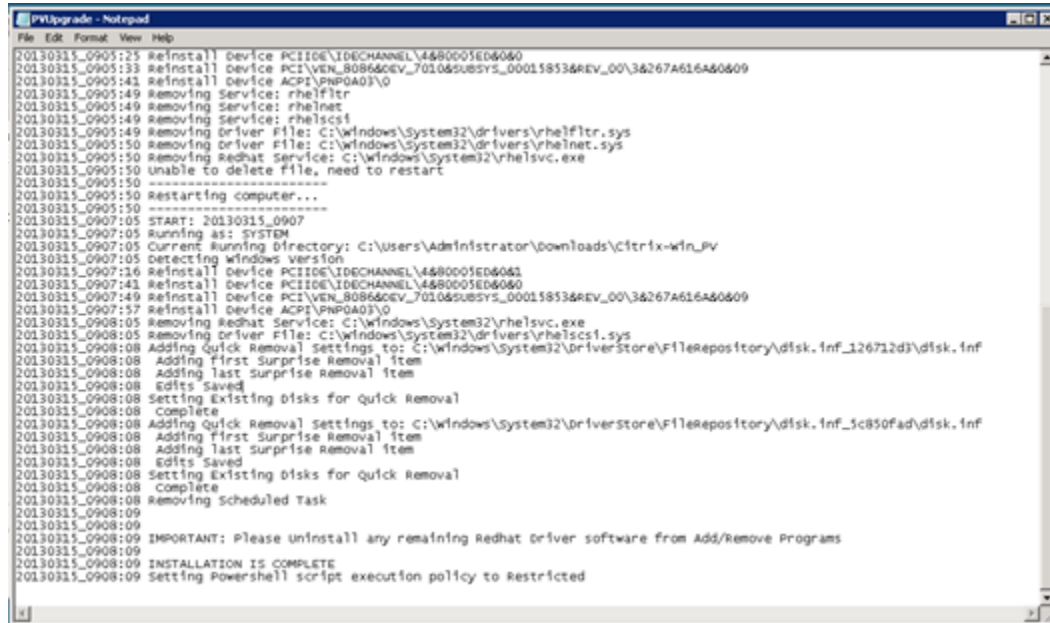   If you do not see the uninstaller dialog box, click **Red Hat Paravirtualiz…** in the Windows taskbar.

7. Check that the instance has rebooted and is ready to be used.

    a. Open the EC2 console.

    b. Under **Instances**, right-click your instance and select **Get System Log**.

    c. Check the end of the log message. It should read `Windows is Ready to use`.

8. Connect to your instance and log in as the local administrator. The upgrade will continue by opening four applications: PowerShell, RedHat uninstaller, PVUpgrade.log and the Windows Device Manager.

9. Uninstall the PCI BUS.

    a. In the **Device Manager** window, expand **System devices**, right-click **PCI bus** and select **Uninstall**.
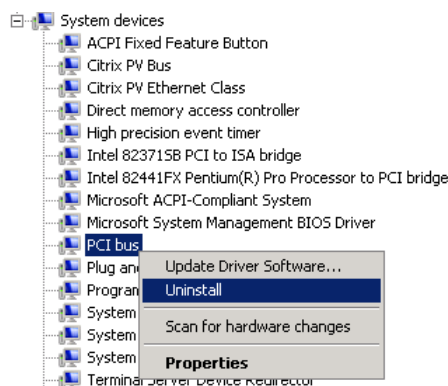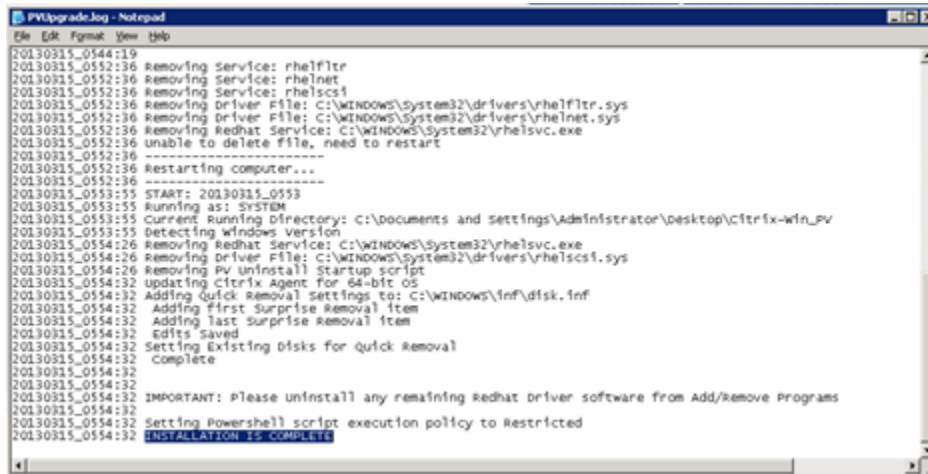
    b. In the **Confirm Device Removal** dialog box, click **OK**.

    c. In the **System Settings Change** dialog, click **No** as you do not want to restart your instance immediately.

    d. Close **Device Manager**. The upgrade script reboots your instance.

10. Check that the instance is ready by repeating the procedure in step 7. After you've confirmed it is ready, log in as the administrator.

11. Confirm that the installation is complete. Navigate to the **Citrix-WIN_PV** folder that you extracted earlier, open the **PVUpgrade.log** file, and then check for the text INSTALLATION IS COMPLETE.

Upgrading Your Windows Server 2008 and Windows Server 2008 R2 Instances

This section explains how to upgrade the RedHat drivers to Citrix drivers on your Windows Server 2008 or Windows Server 2008 R2 instance.

Before you start upgrading your drivers, make sure you do the following:

- Install the latest version of EC2Config by going to Amazon Windows EC2Config Service. For more information about the EC2Config service, see Configuring a Windows Instance Using the EC2Config Service (p. 40).
- Back up your important information on the instance, or create an AMI from the instance. For more information about creating an AMI, see Creating an Amazon EBS-Backed Windows AMI (p. 51). If you create an AMI, make sure you do the following:
  - Do not enable the Sysprep tool in the EC2Config service.
  - Write down your password.
  - Set your ethernet adapter to DHCP.

To upgrade a Windows Server 2008 or Windows Server 2008 R2 AMI

1. Connect to your instance and log in as the local administrator. For more information about connecting to your instance, see Connecting to Windows Instances.
2. In your instance, download the Citrix upgrade package by going to Amazon EC2 Windows Paravirtual Driver Upgrade Script.
3. Extract the contents of the upgrade package to a location of your choice.
4. Double-click the **Upgrade.bat** file. If you get a security warning, click **Run**.
5. In the **Upgrade Drivers** dialog box, review the information and click **Yes** if you are ready to start the upgrade.
6. In the **Red Hat Paravirtualized Xen Drivers for Windows (R) uninstaller** dialog box, click **Yes** to remove the RedHat software. Your instance will reboot.

Note

If you do not see the uninstaller dialog box, click **Red Hat Paravirtualiz…** in the Windows taskbar.

7. Check that the instance has rebooted and is ready to be used.

   a. Open the EC2 console.
   b. Under **Instances**, right-click your instance and select **Get System Log**.
   c. The upgrade operations should have restarted the server 3 or 4 times. You can see this in the log file by the number of times `Windows is Ready to use` is displayed.

8. Connect to your instance and log in as the local administrator.
9. Close the **Red Hat Paravirtualized Xen Drivers for Windows (R) uninstaller** dialog box.
10. Confirm that the installation is complete. Navigate to the **Citrix-WIN_PV** folder that you extracted earlier, open the **PVUpgrade.log** file, and then check for the text `INSTALLATION IS COMPLETE`.

Upgrading your Citrix PV Guest Agent Driver

If you are using Citrix drivers on your Windows server, you can upgrade the Citrix PV guest agent driver. This driver runs as a Windows service, and handles tasks such as time synchronization at boot, and shutdown and restart events from the API. You can run this upgrade package on any version of Windows Server, including Windows Server 2012.

Before you start upgrading your drivers, make sure you back up your important information on the instance, or create an AMI from the instance. For more information about creating an AMI, see Creating an Amazon EBS-Backed Windows AMI (p. 51). If you create an AMI, make sure you do the following:

• Do not enable the Sysprep tool in the EC2Config service.
• Write down your password.
• Set your ethernet adapter to DHCP.

To upgrade your Citrix PV guest driver

1. Connect to your instance and log in as the local administrator. For more information about connecting to your instance, see Connecting to Windows Instances.
2. In your instance, download the Citrix upgrade package by going to Amazon EC2 Windows Paravirtual Driver Upgrade Script.
3. Extract the contents of the upgrade package to a location of your choice.
4. Double-click the **Upgrade.bat** file. If you get a security warning, click **Run**.
5. In the **Upgrade Drivers** dialog box, review the information and click **Yes** if you are ready to start the upgrade.
6. When the upgrade is complete, the PVUpgrade.log file will open and contain the text UPGRADE IS COMPLETE.
7. Reboot your instance.

Document History

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

Change	Description	Release Date
Added new section covering the AWS Management Pack	The AWS Management Pack links Amazon EC2 instances and the Microsoft Windows or Linux operating systems running inside them. The AWS Management Pack is an extension to Microsoft System Center Operations Manager. For more information, see AWS Management Pack for Microsoft System Center Operations Manager (p. 64).	May 8, 2013
Added content	The topic Upgrading Your PV Drivers on Your Windows AMI (p. 117) explains how to upgrade the paravirtualized (PV) drivers on your Windows AMI.	March 2013
Added content	The topic AWS Diagnostics for Microsoft Windows Server (p. 112) describes how to diagnose and troubleshoot possible issues using the AWS Diagnostics for Microsoft Windows Server.	March 2013
Added content	The topic Getting Started with Amazon EC2 Windows Instances (p. 10) helps you launch and connect to your first Windows instance. The topic Controlling Access to Amazon EC2 Resources (p. 30) provides an overview of controlling access to your instances. The topic Deploying a WordPress Blog on Your Amazon EC2 Instance (p. 21) shows how to create and deploy a WordPress blog on your Amazon EC2 instance.	December 2011
Added content	The topic Setting Up a Windows HPC Cluster on Amazon EC2 (p. 96) explains how to configure a Windows HPC Cluster on Amazon Elastic Compute Cloud.	November 2011

Change	Description	Release Date
	This guide provides information about using Amazon EC2 Windows instances. For information about the basic infrastructure components of Windows instances, see What is Amazon EC2? (p. 3). For information about using Windows AMIs, see Windows Amazon Machine Images (AMI) (p. 35). For information about setting up your command line interface, see Installing the Amazon EC2 Command Line Interface Tools on Windows (p. 106).	September 2011