
Getting Started with AWS

Web Application Hosting for Microsoft Windows



Getting Started with AWS: Web Application Hosting for Microsoft Windows

Copyright © 2014 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

The following are trademarks of Amazon Web Services, Inc.: Amazon, Amazon Web Services Design, AWS, Amazon CloudFront, Cloudfront, Amazon DevPay, DynamoDB, ElastiCache, Amazon EC2, Amazon Elastic Compute Cloud, Amazon Glacier, Kindle, Kindle Fire, AWS Marketplace Design, Mechanical Turk, Amazon Redshift, Amazon Route 53, Amazon S3, Amazon VPC. In addition, Amazon.com graphics, logos, page headers, button icons, scripts, and service names are trademarks, or trade dress of Amazon in the U.S. and/or other countries. Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon.

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

Table of Contents

Overview	1
How Does AWS Help?	2
Web Application Hosting Architecture	4
Getting Started	6
Step 1: Sign Up for the Service	7
Step 2: Install the Command Line Tools	7
Step 3: Create an Amazon S3 Bucket	7
Where You're At	9
Step 4: Create a CloudFront Distribution	10
Where You're At	13
Step 5: Create an Elastic Load Balancer	13
Where You're At	17
Step 6: Create and Configure Your Amazon EC2 Security Group	18
Step 7: Create a Key Pair	19
Step 8: Launch Amazon EC2 Instances Using Auto Scaling	20
Where You're At	22
Step 9: Create a CloudWatch Alarm	23
Where You're At	27
Step 10: Add Amazon RDS	27
Create a DB Security Group	28
Authorize Access	28
Launch an Instance	29
Step 11: Deploy Your Application	33
Connect to Your Amazon EC2 Instance	34
Deploy Sample Application	36
Modify the Application	37
Step 12: Create a Custom AMI	40
Step 13: Launch New Environments Using AWS CloudFormation	41
Create an AWS CloudFormation Template	42
Modify an AWS CloudFormation Template	45
Create an AWS CloudFormation Stack	47
Step 14: Clean Up	50
Delete Your CloudWatch Alarm	51
Delete Your Elastic Load Balancer	51
Terminate Your Amazon EC2 Instances in Your Auto Scaling Group	52
Terminate Your DB Instance	54
Delete a Key Pair	54
Delete an Amazon EC2 Security Group	55
Delete an Amazon CloudFront Distribution	55
Delete Objects and an Amazon S3 Bucket	55
Amazon Route 53	57
Pricing	58
Amazon S3 Cost Breakdown	58
Amazon CloudFront Cost Breakdown	60
Amazon EC2 Cost Breakdown	62
Amazon RDS Cost Breakdown	65
Summing It All Up	67
Related Resources	68
Document History	70

Overview

If you purchase hardware to run your website, you might find that highly available and scalable web hosting can be a complex and expensive proposition. Your website would likely experience dense peak traffic periods and significant fluctuations in traffic patterns. This would result in low utilization rates of your expensive hardware, and you could incur high operating costs to maintain mostly idle hardware. Amazon Web Services (AWS) provides the reliable, scalable, secure, and high performance infrastructure required for the most demanding web applications. AWS enables an elastic, scale-out and scale-in infrastructure model that matches IT costs with real-time shifts in customer traffic patterns.

This guide will help you use AWS to create scalable, robust web applications that handle sophisticated demands and workloads. We'll review an example architecture of a web application hosted on AWS, and we'll walk through the process of deploying a sample .NET application using several key AWS services and following best practices. You can adapt this sample to your specific needs if you want. By the end of this walkthrough, you should be able to do the following:

- Sign up for AWS.
- Create a location to store static files for your web application.
- Create a content delivery network.
- Launch, connect, secure, and deploy a sample .NET web application to a computer in the cloud.
- Create a custom template of a computer containing the hardware, software, and configuration you need.
- Set up a load balancer to distribute traffic across multiple computers in the cloud.
- Scale your fleet of computers in the cloud.
- Monitor the health of your application and computers.
- Create a database instance and use it with a sample .NET application.
- Create a template for the AWS resources you created.
- Clean up your AWS resources.

For a deeper understanding of AWS best practices and the various options that AWS provides, we recommend that you read *Web Application Hosting: Best Practices* at [AWS Cloud Computing Whitepapers](#).

If you are looking for a quicker and easier way to deploy your web applications, you can use an application management service. AWS application management services help you leverage other AWS services without having to manage each of them separately and manually:

- [AWS Elastic Beanstalk](#) lets you focus on the code while the service manages the rest.

Getting Started with AWS Web Application Hosting for Microsoft Windows How Does AWS Help?

- [AWS OpsWorks](#) gives you the flexibility to define your own software stack and deploy, operate, and automate a variety of applications and architectures.

For additional information about deployment and resource management on AWS, go to [Deployment and Management on AWS](#).

If this guide is not exactly what you are looking for, you may want to check out the following documents:

- [Getting Started with AWS](#) — Provides information about Amazon Web Services, with helpful links for learning more.
- [Getting Started with AWS Free Usage Tier](#) — Provides information about how to get started with the free usage tier.
- [Hosting Websites on Amazon S3](#) in the *Amazon Simple Storage Service Developer Guide* — Provides a walkthrough in just a few steps of a static website deployment that does not require running an application.
- [Getting Started with AWS CloudFormation](#) in the *AWS CloudFormation User Guide* — Helps you quickly get started using an AWS CloudFormation WordPress blog sample template without needing to figure out the order in which AWS services need to be provisioned or worry about the subtleties of how to make those dependencies work.
- [Getting Started with AWS Computing Basics for Windows](#) - Introduces you to several key AWS services and components—what these services are, why they are important, and how to use them. The guide also provides a simple example architecture on a Windows platform and walks you through a deployment that uses this architecture.
- [Amazon Elastic Compute Cloud Microsoft Windows Guide](#) - Provides information that helps you get started using Amazon EC2 instances that run the Microsoft Windows Server operating system.

How Does AWS Help?

If you are responsible for running a web application then there are a variety of infrastructure and architecture issues that you face for which AWS can give you easy, seamless, and cost-effective solutions. This section provides a list of Amazon Web Services and components, and it explains the value they add in meeting the challenges you'll face in this example solution.

Challenges	Amazon Web Services	Benefits
Servers need to be provisioned to handle peak capacity and the unused cycles are wasted at other times.	<ul style="list-style-type: none">• Amazon Elastic Compute Cloud (EC2)• Amazon Elastic Load Balancing• Auto Scaling• Amazon CloudWatch	<ul style="list-style-type: none">• Amazon EC2 runs the web server and application servers.• Elastic Load Balancing supports health checks on hosts, distribution of traffic to Amazon EC2 instances across multiple Availability Zones, and the dynamic addition and removal of Amazon EC2 hosts from the load-balancing rotation.• Auto Scaling creates capacity groups of servers that can grow or shrink on demand.• Amazon CloudWatch reports metrics data for Amazon EC2 instances, and the metrics it gathers are used by Auto Scaling.

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
How Does AWS Help?**

Challenges	Amazon Web Services	Benefits
Need a content delivery network (CDN) to provide low-latency, high data transfer speeds so end users don't experience unnecessary delays.	<ul style="list-style-type: none"> Amazon CloudFront Amazon Simple Storage Service (Amazon S3) 	<ul style="list-style-type: none"> Amazon CloudFront speeds up the loading of streaming or downloaded static content by caching the content via a local edge cache at a location with the lowest latency. Amazon S3 stores data backups from the relational database, web, and application servers, and for Amazon CloudFront distribution.
Applications may require a database, file system, or access to raw block-level storage.	Amazon Elastic Block Store (Amazon EBS)	Amazon EBS provides a persistent file system for web and application servers.
Maintaining a database can be expensive and time-consuming.	Amazon Relational Database Service (Amazon RDS)	Amazon RDS provides cost-efficient and resizable capacity while managing time-consuming database administration tasks.
Developers and businesses need a reliable and cost-effective way to route end users to Internet applications.	Amazon Route 53	Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. It is designed to give developers and businesses an extremely reliable and cost effective way to route end users to Internet applications by translating human readable names like www.example.com into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other.
Need to plan the order in which Amazon Web Services will be provisioned, keeping in mind dependencies among the services.	AWS CloudFormation	AWS CloudFormation gives developers and systems administrators an easy way to create a collection of related AWS resources and provision them in an orderly and predictable fashion.

Challenges	AWS Components	Benefits
Need to provide security to protect application servers from outside malicious users.	Amazon Security Group	An Amazon Security Group lets you specify the protocols, ports, and source IP address ranges that are allowed to reach your Amazon EC2 instances.

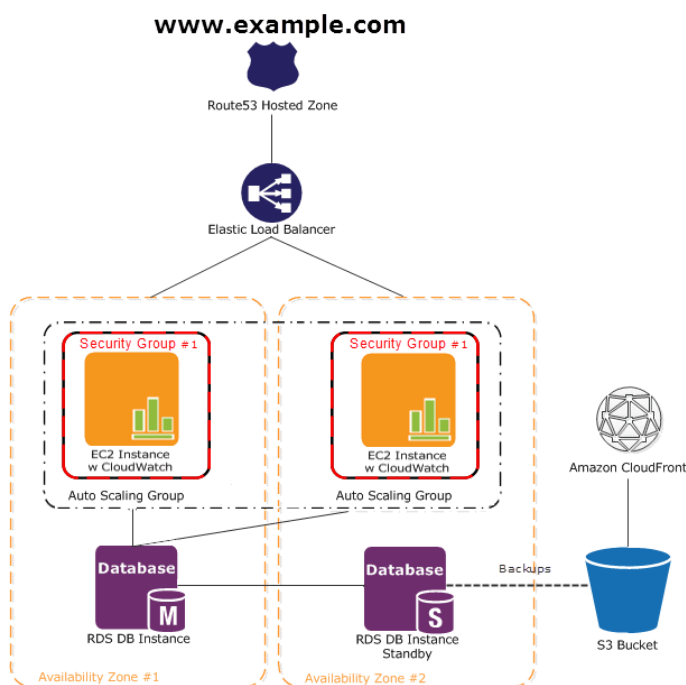
Getting Started with AWS Web Application Hosting for Microsoft Windows

Web Application Hosting Architecture

Challenges	AWS Components	Benefits
Need to design with failover in mind.	Availability Zones	Availability Zones are distinct locations engineered to be insulated from failures in other Availability Zones. Each Availability Zone provides inexpensive, low latency network connectivity to other Availability Zones in the same region.

Web Application Hosting Architecture

The following diagram shows an example architecture of a web application using the AWS resources mentioned in the previous section.



In this diagram, Amazon EC2 instances run the application and web server and belong to an Amazon EC2 Security Group. The Amazon EC2 Security Group acts as an exterior firewall for the Amazon EC2 instances. An Auto Scaling group is used to maintain a fleet of Amazon EC2 instances that can handle the presented load. This Auto Scaling group spans over multiple Availability Zones to protect against potential failures if an Availability Zone becomes unavailable. To ensure that traffic is distributed evenly among the Amazon EC2 instances, an Elastic Load Balancer is associated with the Auto Scaling group. If the Auto Scaling group launches or terminates instances based on load, then the Elastic Load Balancer will automatically adjust accordingly. The database tier consists of Amazon RDS database instances, including master and local slave, located in multiple Availability Zones for failover protection. Amazon RDS provides automated backups to Amazon S3. Amazon S3 stores backups and static content. Since the consumers of this application may be globally distributed or a large number may visit the site at one time, high volume static content is edge cached using Amazon CloudFront for better performance. Amazon Route 53 can be used to provide secure and reliable routing to your infrastructure that uses Amazon Web Services.

For a step-by-step walkthrough of how to build out this architecture, see [Getting Started \(p. 6\)](#). This walkthrough will teach you how to do the following:

Getting Started with AWS Web Application Hosting for Microsoft Windows Web Application Hosting Architecture

- Sign up for AWS.
- Create an Amazon S3 bucket and store files in the bucket.
- Create an Amazon CloudFront distribution.
- Launch, connect, secure, and deploy a .NET sample application to an Amazon EC2 instance.
- Create a Custom AMI.
- Set up an Elastic Load Balancer to distribute traffic across your Amazon EC2 instances.
- Scale your fleet of instances automatically using Auto Scaling.
- Monitor your AWS resources using Amazon CloudWatch.
- Create a database instance and use it with a sample .NET application.
- Create an AWS CloudFormation template based on the resources you created.
- Clean up your AWS resources.

For more information on how to use Amazon Route 53 in this architecture, see [Amazon Route 53 \(p. 57\)](#).

Getting Started

Topics

- [Step 1: Sign Up for the Service \(p. 7\)](#)
- [Step 2: Install the Command Line Tools \(p. 7\)](#)
- [Step 3: Create an Amazon S3 Bucket \(p. 7\)](#)
- [Step 4: Create a CloudFront Distribution \(p. 10\)](#)
- [Step 5: Create an Elastic Load Balancer \(p. 13\)](#)
- [Step 6: Create and Configure Your Amazon EC2 Security Group \(p. 18\)](#)
- [Step 7: Create a Key Pair \(p. 19\)](#)
- [Step 8: Launch Amazon EC2 Instances Using Auto Scaling \(p. 20\)](#)
- [Step 9: Create a CloudWatch Alarm \(p. 23\)](#)
- [Step 10: Add Amazon RDS \(p. 27\)](#)
- [Step 11: Deploy Your Application \(p. 33\)](#)
- [Step 12: Create a Custom AMI \(p. 40\)](#)
- [Step 13: Launch New Environments Using AWS CloudFormation \(p. 41\)](#)
- [Step 14: Clean Up \(p. 50\)](#)

Let's suppose you want to build an ASP.NET web application where customers can upload and browse photos. You want to leverage the reliable, scalable, secure, and high performance infrastructure that AWS offers. It's easy to get started, and for most of the tasks we can use the [AWS Management Console](#). In this topic, we'll walk through a series of steps to deploy your web application to AWS. There are many deployment variations for web applications, but this walkthrough focuses on one example that follows best practices and uses many of the Amazon Web Services so you can see how the services work together. Let's begin!

Note

In this example, we are going through the steps in a specific order to minimize the time for billable services. However, when you deploy your application you will likely start by launching your Amazon EC2 instance, configuring your application and database, creating a custom AMI, and then scaling your application.

Step 1: Sign Up for the Service

If you don't already have an AWS account, you'll need to get one. Your AWS account gives you access to all services, but you will be charged only for the resources that you use. For this example walkthrough, the charges will be minimal.

To sign up for AWS

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

AWS notifies you by email when your account is active and available for you to use.

You use your AWS account to deploy and manage resources within AWS. If you give other people access to your resources, you will probably want to control who has access and what they can do. AWS Identity and Access Management (IAM) is a web service that controls access to your resources by other people. In IAM, you create users, which other people can use to obtain access and permissions that you define. For more information about IAM, go to [Using IAM](#).

Step 2: Install the Command Line Tools

We'll need to install some command line tools for Auto Scaling. Do this first to minimize your usage of billable services.

To install the Auto Scaling command line tools to your local computer go to [Using the Command Line Tools](#) in the *Auto Scaling Developer Guide*. The package has a README that includes instructions on how to install the command line tools. After you have installed the command line tools, try a couple of commands to make sure they work.

After you have installed the command line tools, you can start creating your AWS resources. First you are going to need a place to store your photos when you upload them to the website. [Amazon Simple Storage Service](#) is a perfect solution for storing static objects such as photos. Let's proceed to the next step to create our first Amazon S3 bucket.

Step 3: Create an Amazon S3 Bucket

We'll use Amazon Simple Storage Service (Amazon S3) to store our photos and other static objects. We'll make the objects fully public so you can see how Amazon S3 acts as an image server in the cloud. And because Amazon S3 storage can be redundant across multiple data centers, it provides very high levels of durability for only a few cents per gigabyte per month.

Every object in Amazon S3 is stored in a bucket. Before we can store data in Amazon S3 we must create a bucket.

Note

You are not charged for creating a bucket; you are only charged for storing objects in the bucket and for transferring objects in and out of the bucket. The charges you will incur working through this example are minimal. For more information, go to [Amazon S3 Pricing](#).

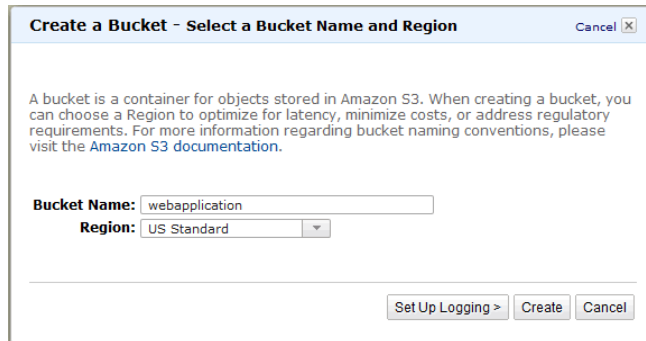
In this step we will create an Amazon S3 bucket for static content and enable logging to hold application data.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 3: Create an Amazon S3 Bucket

To create a bucket

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. On the Amazon S3 console, click **Create Bucket**.



Create a Bucket - Select a Bucket Name and Region Cancel X

A bucket is a container for objects stored in Amazon S3. When creating a bucket, you can choose a Region to optimize for latency, minimize costs, or address regulatory requirements. For more information regarding bucket naming conventions, please visit the [Amazon S3 documentation](#).

Bucket Name:

Region:

[Set Up Logging >](#) [Create](#) [Cancel](#)

3. Enter a bucket name in the **Bucket Name** field.

The bucket name you choose must be unique across all existing bucket names in Amazon S3. One way to help ensure uniqueness is to prefix your bucket names with the name of your organization. For this example, we'll use `webapplication`; however, you should choose a unique name.

Bucket names must comply with the following requirements. Bucket names:

- Can contain lowercase letters, numbers, underscores (`_`), and dashes (`-`)
- Must start with a number or letter
- Must be between 3 and 255 characters long
- Must not be formatted as an IP address (e.g., `265.255.5.4`), and do not use periods.

In some AWS regions, there might be additional restrictions on bucket names. If you intend to access objects by using a URL, we recommend that you create bucket names that are DNS-compliant. For more information, see the [Amazon Simple Storage Service Console User Guide](#).

Note

After you create a bucket, you cannot change its name. In addition, the bucket name is visible in the URL that points to the objects stored in the bucket. Ensure that the bucket name you choose is appropriate.

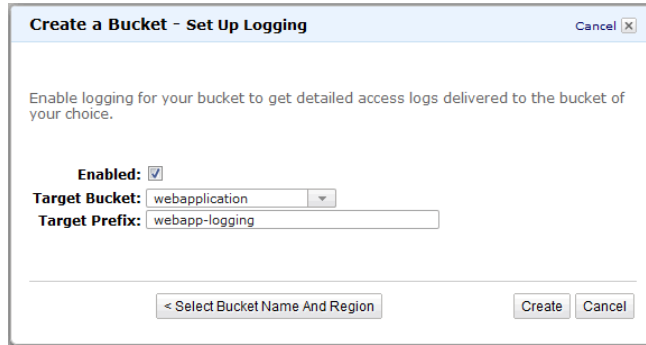
4. In the **Region** box, select a region.

By default, Amazon S3 creates buckets in the US Standard region. You can choose a region to optimize latency, minimize costs, or address regulatory requirements. Objects stored in a region never leave that region unless you explicitly transfer them to another region. For more information about regions, see [Introduction to Amazon S3](#) in the *Amazon Simple Storage Service Console User Guide*.

Click **Set Up Logging**. Log files will be created and stored in your Amazon S3 bucket. The log files will store application data.

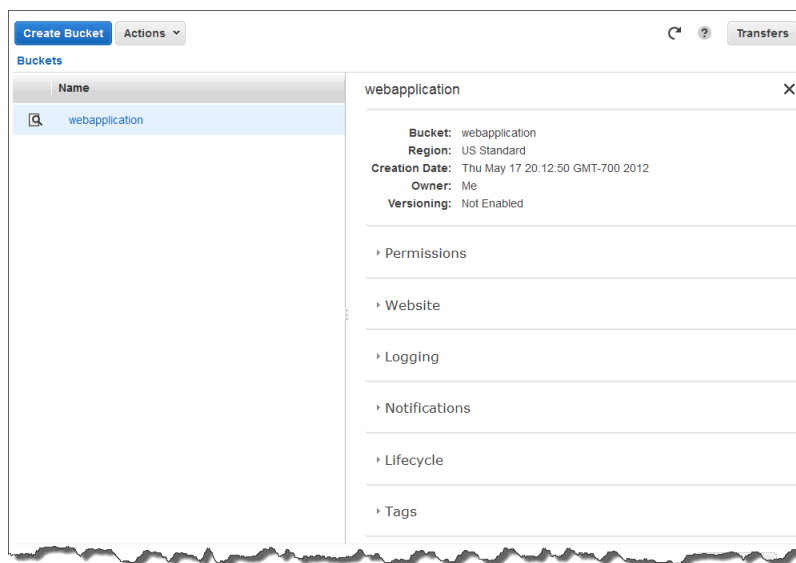
5. In the **Create a Bucket — Set Up Logging**, do the following:
 - Select the **Enabled** check box.
 - In the **Target Bucket** box, select `webappapplication`.
 - In the **Target Prefix** box, type `webapp-logging`.

Getting Started with AWS Web Application Hosting for Microsoft Windows Where You're At



6. When the settings are as you want them, click **Create**.

When Amazon S3 successfully creates your bucket, the console displays the properties of your empty bucket.



After this step, you would normally upload your content and set permissions. However, for this exercise we will do so after we deploy our web application. For more information about Amazon S3, see [Amazon Simple Storage Service \(S3\) Documentation](#).

Where You're At

Here's where you are at in building out your architecture.



Now that you have finished creating your bucket, you need to think about how you deliver the content. Let's suppose you have people viewing your website from other parts of the world. The experience for these people may be unsatisfactory if they experience a lot of latency. [Amazon CloudFront](#) is a good solution for the latency problem. Let's move on to the next step to create an Amazon CloudFront distribution.

Step 4: Create a CloudFront Distribution

Amazon CloudFront is a content delivery service from Amazon Web Services that helps you improve the performance, reliability, and availability of your websites and applications. The content you deliver with Amazon CloudFront will be stored on an *origin server*. Amazon CloudFront works by distributing your web content (such as images, video, and so on) using a network of *edge locations* around the world. Your content is served from your configured Amazon S3 bucket or custom origin, to the edge location that has the lowest latency to the user who requests it. Amazon CloudFront is a good choice if you have users that are globally distributed, expect high volume traffic, or even expect at least one hit per day. In this example, the origin is the Amazon S3 bucket where you store static content. Let's create a CloudFront distribution.

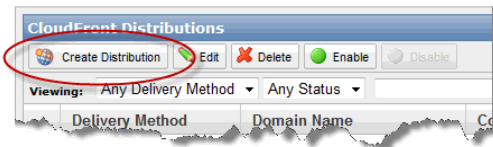
Note

This step is optional. You can skip this step and still deploy your web application at the end of this tutorial.

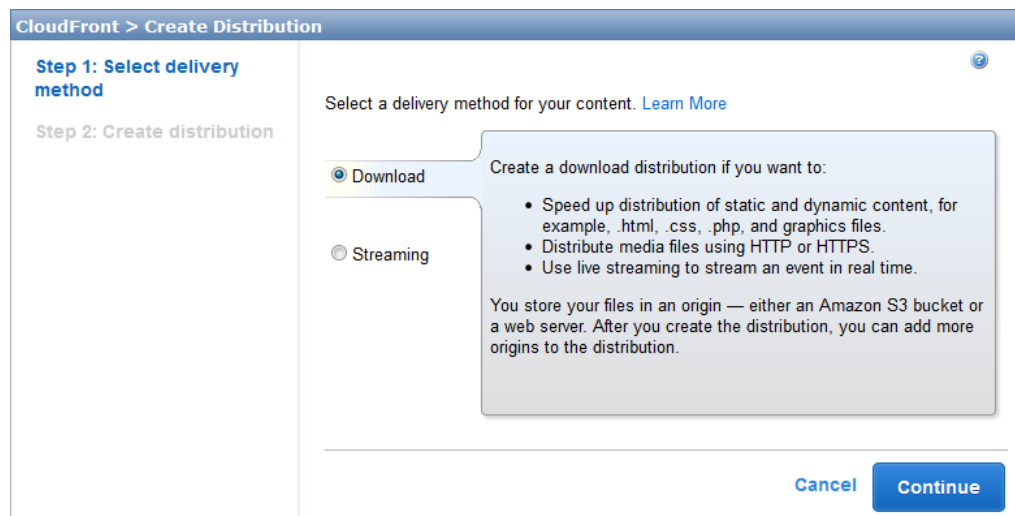
To create a distribution with an Amazon S3 origin, we will use the AWS Management Console.

To create an Amazon CloudFront distribution

1. Open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Make sure that **US East (N. Virginia)** is selected in the region selector of the navigation bar.
3. In the **CloudFront** console click **Create Distribution**.

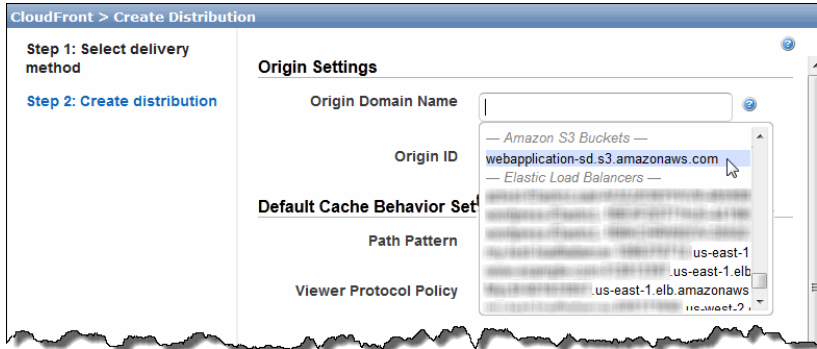


4. On the first page of the **Create Distribution** wizard, accept the default selection, **Download**, and click **Continue**.



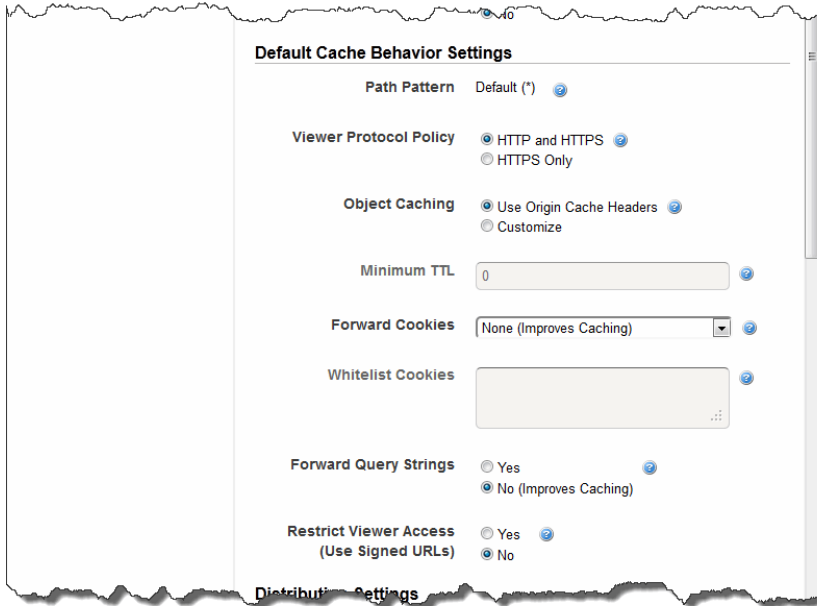
5. On the **Create Distribution** page under **Origin Settings**, select the Amazon S3 bucket that you created earlier.

Step 4: Create a CloudFront Distribution



- Forward all requests that use the CloudFront URL for your distribution (for example, `http://d1111111abcdef8.cloudfront.net/index.html`) to the Amazon S3 bucket that you specified in Step 4.
- Allow end users to use either HTTP or HTTPS to access your objects.
- Cache your objects at CloudFront edge locations for 24 hours.
- Exclude query string parameters, if any, when forwarding requests for objects to your origin.

For more information about cache behavior options, go to [Cache Behaviors](#) in the *Amazon CloudFront Developer Guide*.



- a. If your distribution will use a CNAME, for **CNAMEs**, enter the CNAME alias you want to associate with this distribution. You must own the domain name and have created the CNAME record with your registrar. For this example, leave CNAMEs blank.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 4: Create a CloudFront Distribution

- b. If you want to use a default root object with your distribution, for **Default Root Object** enter the default root object to associate with the distribution. For example, `index.html`. For this example, leave this field blank.
- c. If you want to enable logging, for **Logging** select **On**, and then from **Bucket for Logs**, select the Amazon S3 bucket to which you want to save your logs. For this example, we select **webapplication.s3.amazonaws.com**, however you should select the bucket that you created. Type the log file prefix in the **Log Prefix** field. For this example, we type **webapp-logging/**.
- d. In **Comments**, you can enter any comments you want to include about the distribution. We will leave this blank for this example.
- e. For the **Distribution Status**, select **Enabled** if you want the distribution to accept end-user requests for content as soon as it is deployed. Otherwise, if you prefer to enable the distribution later, choose **Disabled**. For this example, select **Enabled**.

Distribution Settings

Price Class: Use All Edge Locations (Best Performance)

Alternate Domain Names (CNAMEs):

Default Root Object:

Logging: ☒ On ☐ Off

Bucket for Logs: webapplication.s3.amazonaws.com

Log Prefix: webapp-logging/

Cookie Logging: ☐ On ☒ Off

Comment:

Distribution State: ☒ Enabled ☐ Disabled

Buttons: Cancel, Back, Create Distribution

9. If you are satisfied with the distribution settings, click **Create Distribution**.

After creating the distribution, it might take a few minutes to deploy. The distribution's current status is displayed in the console under **Status**. **InProgress** indicates that the distribution is not yet fully deployed.

CloudFront: Distributions								
Create Distribution Distribution Settings Create Enable Disable Show/Hide Refresh Help								
Viewing: Any Delivery Method Any Status								
	Delivery Method	ID	Domain Name	Comment	Origin	CNAMEs	Status	Last Modified
<input checked="" type="checkbox"/>	Download	E183T1ABMQT2DX	d1j6kxrgjehco.cloudfront.net	-	webapplication.s3.amazonaws.com	-	Deployed	2013-03-19 14:25 UTC

Note

Even if the distribution's status is **Deployed**, you still must enable the distribution for use before end users can retrieve content.

When your distribution is deployed, you are ready to reference your content with your new Amazon CloudFront domain name or CNAME.

For more information about Amazon CloudFront, see [Amazon CloudFront Documentation](#). Since the deployment step can take a few minutes to complete, we'll move on to the next step and check our progress later on.

Where You're At

Here's where you are at in building out your architecture.



While we are waiting for the Amazon CloudFront distribution to be deployed, you are ready to start thinking about launching your Amazon EC2 instances. Even though for the purposes of this tutorial, you only have one Amazon EC2 instance up and running, you'll want to have multiple Amazon EC2 instances running across multiple Availability Zones eventually. This way if one Availability Zone goes down, the traffic will be rerouted to another Availability Zone. To prepare for the eventuality of maintaining multiple Amazon EC2 instances, we'll go ahead and create our Elastic Load Balancer resource. In the AWS CloudFormation step, we can scale out to make use of our Elastic Load Balancer. Let's move on to the next step to create our [Elastic Load Balancer](#).

Step 5: Create an Elastic Load Balancer

Elastic Load Balancing is a cost-effective and easy-to-use web service to help you improve the availability and scalability of your application. It makes it easy for you to distribute application loads between two or more Amazon EC2 instances. Elastic Load Balancing enables availability through redundancy and supports traffic growth of your application.

Elastic Load Balancing lets you automatically distribute and balance the incoming application traffic among all the instances you are running. The service also makes it easy to add new instances when you need to increase the capacity of your application. You can dynamically register or deregister instances from the load balancer as the capacity requirements of your application change with time.

As soon as your load balancer becomes available, you're billed for each hour or partial hour that you keep the load balancer running. For more information about Elastic Load Balancing, see the [Elastic Load Balancing](#) details page.

In this step, we will create a load balancer for an HTTP service. We'll specify that the load balancer listens on port 80 and distributes traffic to port 80 on the instances.

Note

We'll go ahead and create our Elastic Load Balancer resource so that in the future when you have multiple instances running, your traffic will be load balanced between your instances. Elastic Load Balancing is a small cost relative to instance hours. In [Step 13: Launch New Environments Using AWS CloudFormation \(p. 41\)](#) we'll use AWS CloudFormation to create a template for our resources and add instances to our Auto Scaling Group.

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Step 5: Create an Elastic Load Balancer**

For more information about elastic load balancers, go to the [Elastic Load Balancing Documentation](#).

To create a load balancer

1. Define a load balancer.
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. In the top navigation bar, select **US East (N. Virginia)** from the region selector.
 - c. In the left navigation pane, click **Load Balancers**.
 - d. Click **Create Load Balancer**.
 - e. In the **Create a New Load Balancer** wizard, on the **Define Load Balancer** page, enter a name for your load balancer. For this example, type **MyLB**.

The screenshot shows the 'Create Load Balancer' wizard in the AWS Management Console. The title is 'Create Load Balancer'. Below the title are four tabs: '1. Define Load Balancer' (selected), '2. Configure Health Check', '3. Add EC2 Instances', and '4. Review'. A paragraph of text explains the wizard's purpose: 'This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name to identify it from other load balancers you might create. You will also need to configure ports and protocols for your load balancer so that traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've configured the load balancer with a standard web server on port 80.' Below this text are several input fields: 'Load Balancer name:' with the value 'MyLB', 'Create LB Inside:' with a dropdown menu showing 'EC2-Classical', and 'Create an internal load balancer:' with an unchecked checkbox and a link '(what's this?)'. Below these is the 'Listener Configuration:' section, which contains a table with four columns: 'Load Balancer Protocol', 'Load Balancer Port', 'Instance Protocol', and 'Instance Port'. The first row shows 'HTTP' for the Load Balancer Protocol, '80' for the Load Balancer Port, 'HTTP' for the Instance Protocol, and '80' for the Instance Port. There is an 'Add' button below the table and a close button (X) in the top right corner of the table area.

- f. Leave **Create LB Inside** set to **EC2-Classical**.
 - g. Leave the **Listener Configuration** set to the default values.
 - h. Click **Continue**.
2. Configure the health check for your load balancer. Elastic Load Balancing routinely checks the health of each load-balanced Amazon EC2 instance. If Elastic Load Balancing finds an unhealthy instance, it stops sending traffic to the instance and reroutes traffic to healthy instances
 - a. For this example, leave **Ping Protocol** set to its default value of **HTTP**. When you deploy your application in the future, you can specify **HTTPS**. For information on using HTTPS with Elastic Load Balancing, see [Elastic Load Balancing Security Features](#) in *Elastic Load Balancing Developer Guide*.
 - b. For this example, leave **Ping Port** set to its default value of **80**.

Elastic Load Balancing uses the **Ping Port** to send health check queries to your Amazon EC2 instances.

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Step 5: Create an Elastic Load Balancer**

Note

If you specify a port value, your Amazon EC2 instances must accept incoming traffic on the port that you specified for the health check. You can set a different port value other than 80, and you can come back and set this value at a later time. However, for this example, set it to 80.

- c. In the **Ping Path** field, replace the default value with a single forward slash ("/").

Elastic Load Balancing sends health check queries to the path you specify in **Ping Path**. This example uses a single forward slash so that Elastic Load Balancing sends the query to your HTTP server's default home page, whether that default page is named `index.html`, `default.html`, or a different name. When you deploy your application, consider creating a special lightweight file that only responds to the health check. This helps differentiate between traffic that is hitting your site and responses to the load balancer.

- d. Set the **Healthy Threshold** to **2**. Leave the rest of the **Advanced Details** set to their default values.

Create Load Balancer

1. Define Load Balancer **2. Configure Health Check** 3. Add EC2 Instances 4. Review

Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances and only route traffic to instances that pass the health check. If an instance fails the health check, it is automatically removed from the load balancer. Configure the health check to meet your specific needs.

Ping Protocol

Ping Port

Ping Path

Advanced Details

Response Timeout ⓘ seconds

Health Check Interval ⓘ seconds

Unhealthy Threshold ⓘ

Healthy Threshold ⓘ

- e. Click **Continue**.

3. On the **Add Instances to Load Balancer** page, click **Continue**.
4. Review your settings. You can make changes to the settings by clicking the **Edit** link for a specific step in the process.

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Step 5: Create an Elastic Load Balancer**

Create Load Balancer

1. Define Load Balancer 2. Configure Health Check 3. Add EC2 Instances 4. Review

Review

Please review the load balancer details before continuing

▼ Define Load Balancer

Edit load balancer definition

Load Balancer name: MyLB

Scheme: internet-facing

Port Configuration: 80 (HTTP) forwarding to 80 (HTTP)

▼ Configure Health Check

Edit health check

Ping Target: HTTP:80/

Timeout: 5 seconds

Interval: 30 seconds

Unhealthy Threshold: 2

Healthy Threshold: 2

▼ Add EC2 Instances

Edit instances

Cross-Zone Load Balancing: Enabled

Connection Draining: Enabled, 300 seconds

Instances:

Back Create

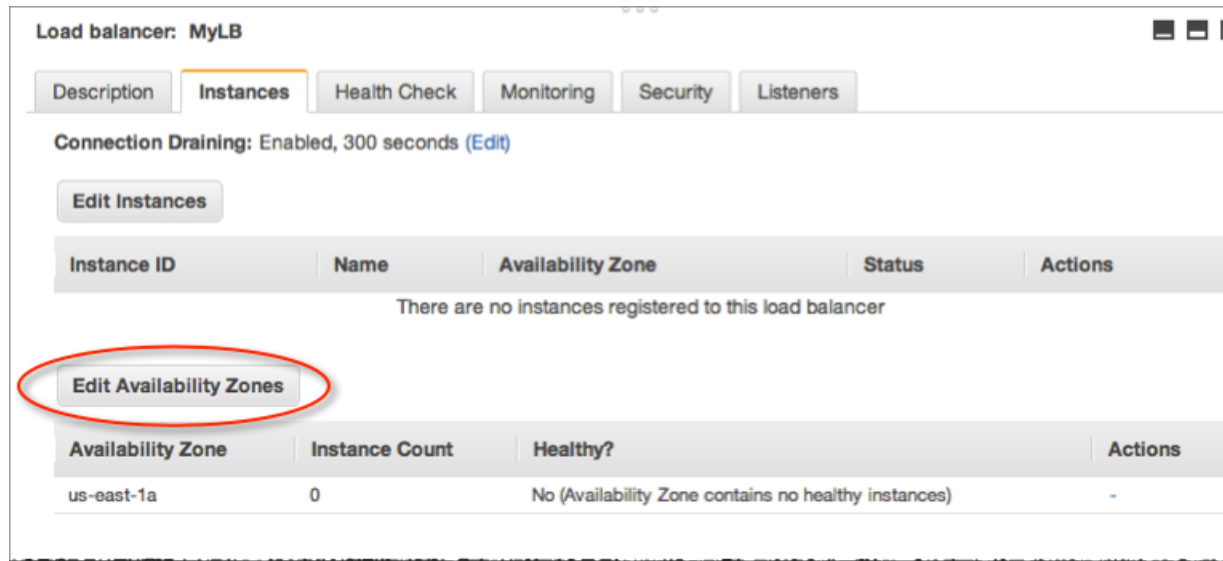
5. Click **Create**. On the **Create Load Balancer** confirmation page, click **Close**.

Your new load balancer now appears in the list.

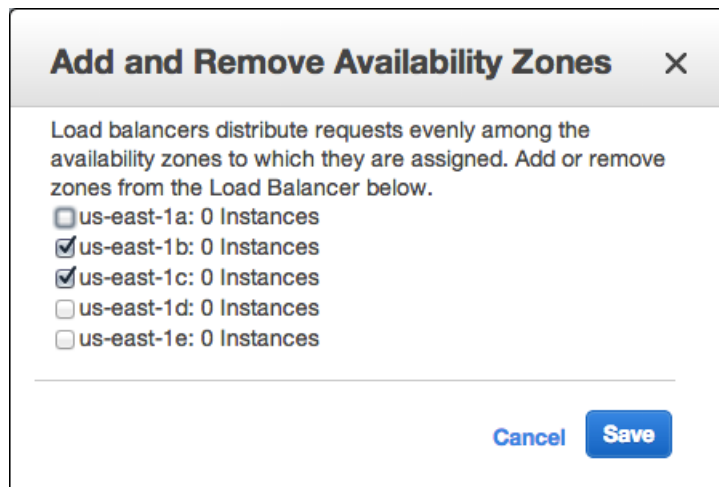
As a best practice, you should have sufficient instances across Availability Zones to survive the loss of any one Availability Zone. In the next step, you'll ensure that your load balancer points to multiple Availability Zones.

6. Add an Availability Zone.
 - a. In the **Load Balancers** list, click **MyLB**.
 - b. Click the **Instances** tab.
 - c. Click **Edit Availability Zones**.

Getting Started with AWS Web Application Hosting for Microsoft Windows Where You're At



- d. In the **Add and Remove Availability Zones** dialog box, do the following:
- Click **us-east-1b: 0 instances**.
 - Click **us-east-1c: 0 instances**.
 - Click **Save**.



The Availability Zones column for the load balancer now shows the Availability Zones you selected.

Where You're At

Here's where you are at in building your architecture.



Let's move on to the next topic to create your Amazon EC2 security group. You will need to create an Amazon EC2 security group in order to open up ports on your instance. Your security group is essentially acting as a firewall.

Step 6: Create and Configure Your Amazon EC2 Security Group

An Amazon EC2 security group acts as a firewall that controls the traffic allowed into a group of instances. When you launch an Amazon EC2 instance, you can assign it to one or more security groups. For each security group, you add rules that govern the allowed inbound traffic to instances in the group. All other inbound traffic is discarded. You can modify rules for a security group at any time. The new rules are automatically enforced for all existing and future instances in the group.

In this step, we will do the following:

- Create an Amazon EC2 security group
- Configure an Amazon EC2 security group

To create and configure your security group

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Make sure **US East (N. Virginia)** is selected in the region selector of the navigation bar.

Note

For this purposes of this walkthrough, we will always use the US East (N. Virginia) region. However when you deploy your application, select the region that is closest to you.

3. In the navigation pane, click **Load Balancers**.
4. Select the load balancer that you created earlier, and click the **Security** tab. In the **Source Security Group** field, copy or write down the name of the security group that's associated with the load balancer. You will need the name to configure your instance's security group rules.
5. In the navigation pane, click **Security Groups**, and then click **Create Security Group**.
6. In the **Create Security Group** dialog box, type `webappsecuritygroup` in the **Security group name** field, and a description of your choice in the **Description** field.
7. On the **Inbound** tab, click **Add Rule**, and select **HTTP** from the **Type** list.
8. Select **Custom IP** from the **Source** field, and enter the name of the security group that's associated with your load balancer, for example, `amazon-elb/amazon-elb-sg`. When you select this source,

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 7: Create a Key Pair

this means that only traffic that comes through the Elastic Load Balancer can connect to your Amazon EC2 instance.

9. Click **Add Rule**.
10. Select **RDP** from the **Type** list to connect to your Amazon EC2 instances. Select **Anywhere** from the Source list.

Create Security Group

Security group name: webappsecuritygroup

Description: My security group

VPC: No VPC

Security group rules:

Inbound

Type	Protocol	Port Range	Source
HTTP	TCP	80	Custom IP: amazon-elb/amazon-
RDP	TCP	3389	Anywhere: 0.0.0.0/0

Add Rule

Cancel Create

Important

The security group settings are configured to allow access from everywhere: 0.0.0.0/0. This is not good practice, and it is only for the purposes of this exercise that we are setting it up this way. Best practice should be to set rules that restrict access to only those computers or networks that require access to this service. The number after the "/" indicates a range of addresses.

11. Click **Create**.

Your Amazon EC2 security group is not yet enforced. We will enforce this when we create our Auto Scaling group. However, you can also apply an Amazon EC2 security group to an Amazon EC2 instance. For more information, see [Using Security Groups](#) in the *Amazon Elastic Compute Cloud User Guide*.

Now that we have created our Amazon EC2 security group, we will need a way to access our Amazon EC2 instance to deploy our application. Public AMI instances use a public/private key pair to login rather than a password. Let's move on to the next section to create your key pair.

Step 7: Create a Key Pair

You can create your key pair so that you can connect to your Amazon EC2 instances. Public AMI instances use a public/private key pair to log in rather than a password. The public key half of this pair is embedded in your instance, allowing you to use the private key to log in securely without a password. After you create your own AMIs, you can choose other mechanisms to securely log in to your new instances. In this step we will use AWS Management Console to create a key pair.

To generate a key pair

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the top navigation bar, in the region selector, click **US East (N. Virginia)**.
3. In the left navigation pane, under **Network and Security**, click **Key Pairs**.
4. Click **Create Key Pair**.
5. Type `mykeypair` in the new **Key pair name** box, and then click **Yes**.
6. Download the private key file, which is named `mykeypair.pem`, and keep it in a safe place. You will need it to access any instances that you launch with this key pair.

Important

If you lose the key pair, you cannot connect to your Amazon EC2 instances.

For more information about key pairs, see [Amazon EC2 Key Pairs](#) in the *Amazon Elastic Compute Cloud User Guide*.

Since your traffic may vary, you want AWS to scale the number instances appropriately. To do this you'll want to use [Auto Scaling](#) to create an Auto Scaling group. Let's move on to the next step to create our Auto Scaling group and associate our Auto Scaling group with our Elastic Load Balancer.

Step 8: Launch Amazon EC2 Instances Using Auto Scaling

Auto Scaling is designed to launch or terminate Amazon EC2 instances automatically based on user-defined policies, schedules, and alarms. You can use Auto Scaling to maintain a fleet of Amazon EC2 instances that can handle any presented load. As its name implies, Auto Scaling responds automatically to changing conditions. All you need to do is specify how it should respond to those changes. For example, you can instruct Auto Scaling to launch an additional instance whenever CPU usage exceeds 60 percent for ten minutes, or you could tell Auto Scaling to terminate half of your website's instances over the weekend when you expect traffic to be low. You can also use Auto Scaling to ensure that the instances in your fleet are performing optimally, so that your applications continue to run efficiently. Auto Scaling groups can even work across multiple Availability Zones—distinct physical locations for the hosted Amazon EC2 instances—so that if an Availability Zone becomes unavailable, Auto Scaling will automatically redistribute applications to a different Availability Zone. With Auto Scaling, you can ensure that you always have at least one healthy instance running. For more information, see [Auto Scaling](#).

In this example, we will set up the basic infrastructure that must be in place to get Auto Scaling started for most applications. We'll set up an Amazon EC2 application to be load-balanced and auto-scaled with a minimum number of one instance and maximum number of one instance so you are only charged for one instance. However, when you create your actual website you should follow the best practice of having sufficient instances across Availability Zones to survive the loss of any one Availability Zone. Additionally, increase your maximum number of instances to be greater than your minimum to make use of the Auto Scaling feature. You can also specify the maximum number of instances to control your fleet size. Auto Scaling in this example is configured to scale out by one when there is a change in capacity. We define the policy in this topic and then create a CloudWatch alarm in the next section to take action on the policy when the average CPU usage exceeds a threshold of 60 percent for 10 minutes. Auto Scaling and Amazon CloudWatch work together to launch or terminate instances based on the policies you create. To save time, we will create just one policy, however, you can create more policies, such as a scale-in policy.

If you haven't already installed the Auto Scaling command line tools, you need to do that now at [Using the Command Line Tools](#) in the *Auto Scaling DeveloperGuide*. We will use the command line tools to set up Auto Scaling.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 8: Launch Amazon EC2 Instances Using Auto Scaling

To set up an auto-scaled, load-balanced Amazon EC2 application

1. Open a command prompt window. In Microsoft Windows, start the Command Prompt application (from the **Start** menu, click **Programs**, click **Accessories**, and then click **Command Prompt**).
2. Use the Auto Scaling `as-create-launch-config` command.

In this example, we use a publicly available Windows AMI running Microsoft Windows Server 2008 and Microsoft Internet Information Services (IIS). We use an `t1.micro` instance type, and use our security group and our key pair we created in the previous steps. In this example, the key pair file is located in the directory in which we are creating our Auto Scaling group. We will not specify a region because we want to use the default region, US East (N. Virginia).

Note

The AMI that is used in this example is part of the [AWS Free Usage Tier](#). If you are eligible for the free usage tier, then you will not be charged for launching the Amazon EC2 instance. If you are not eligible for the AWS Free Usage Tier, the charges in this example are minimal. For more information about Amazon EC2 pricing, see the [Amazon Elastic Compute Cloud \(Amazon EC2\)](#) details page.

```
PROMPT>as-create-launch-config MyLC --image-id ami-c082e3a9 --instance-type t1.micro --group webappsecuritygroup --key mykeypair
```

Auto Scaling returns the following:

```
OK-Created launch config
```

Note

You can copy and paste the commands from the document into the command line window. To paste the contents in the command line window, use right-click. If you have trouble getting the commands to work, make sure the command was pasted correctly.

3. Use the Auto Scaling `as-create-auto-scaling-group` command. In this example, we use two Availability Zones. This is a good practice for building fault-tolerant applications. If one Availability Zone experiences an outage, traffic will be routed to another Availability Zone. The number of instances that are launched in the Auto Scaling group will be evenly distributed across the Availability Zones.

```
PROMPT>as-create-auto-scaling-group MyAutoScalingGroup --launch-configuration MyLC --availability-zones us-east-1b, us-east-1c --min-size 1 --max-size 1 --load-balancers MyLB
```

Auto Scaling returns the following:

```
OK-Created AutoScalingGroup
```

4. Use the Auto Scaling `as-put-scaling-policy` command to create a policy to enlarge your fleet of instances.

```
PROMPT>as-put-scaling-policy MyScaleUpPolicy --auto-scaling-group MyAutoScalingGroup --adjustment=1 --type ChangeInCapacity --cooldown 300
```

Auto Scaling returns output similar to the following example output:

Getting Started with AWS Web Application Hosting for Microsoft Windows Where You're At

```
POLICY-ARN arn:aws:autoscaling:us-east-1:012345678901:scalingPolicy:cbe7da4e-5d00-4882-900a-2f8113431e30:autoScalingGroupName/MyAutoScalingGroup:policyName/MyScaleUpPolicy
```

Note

To save time, we only created a scale-out policy. However, you typically would want to create a scale-in policy as well. Auto Scaling decreases the number of instances when your application doesn't need the resources, saving you money. To create a scale-in policy, change the policy name and change the adjustment from 1 to -1.

5. Verify that your Auto Scaling group exists by using the `as-describe-auto-scaling-groups` command.

```
PROMPT>as-describe-auto-scaling-groups MyAutoScalingGroup --headers
```

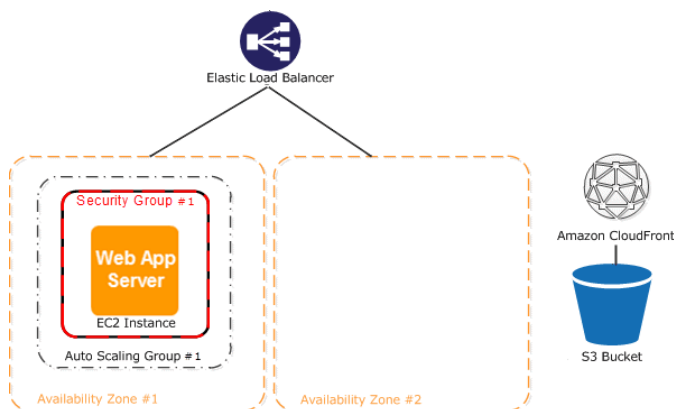
Auto Scaling returns the following:

AUTO-SCALING-GROUP	GROUP-NAME	LAUNCH-CONFIG	AVAILABILITY-ZONES		
MIN-SIZE	MAX-SIZE	DESIRED-CAPACITY			
AUTO-SCALING-GROUP	MyAutoScalingGroup	MyLC	us-east-1b,us-east-1c		
1	1	1			
INSTANCE	INSTANCE-ID	AVAILABILITY-ZONE	STATE	STATUS	LAUNCH-CONFIG
INSTANCE	i-xxxxxxx	us-east-1c	InService	Healthy	MyLC

Your Amazon EC2 application has been launched as an auto-scaled and load-balanced application. For more information about Auto Scaling, see the [Auto Scaling Documentation](#). You will continue to incur costs as long as your Amazon EC2 instances are running. If at any time you want to terminate these instances, see [Terminate Your Amazon EC2 Instances in Your Auto Scaling Group](#) (p. 52).

Where You're At

Here's where you are at in building your architecture.



Now that you have created your Auto Scaling group and your Amazon EC2 instance is up and running, you'll want a way to monitor the health of your instance. In the next step, we'll create an Amazon CloudWatch alarm so we can track the Auto Scaling policy you just created.

Step 9: Create a CloudWatch Alarm

Amazon CloudWatch is a web service that enables you to monitor, manage, and publish various metrics, as well as configure alarm actions based on data from metrics.

With Amazon CloudWatch you can collect, analyze, and view system and application metrics so that you can make operational and business decisions quickly and confidently. Amazon CloudWatch automatically collects metrics about your AWS resources—such as the performance of your Amazon EC2 instances. You can also publish your own metrics directly to Amazon CloudWatch.

Amazon CloudWatch alarms help you implement decisions more easily by enabling you to send notifications or automatically make changes to the resources you are monitoring, based on rules that you define. For example, you can create alarms that initiate Auto Scaling and Amazon Simple Notification Service (Amazon SNS) actions on your behalf.

A common use for Amazon CloudWatch is to keep your applications and services healthy and running efficiently. For example, you can use it to discover that your website runs best when network traffic remains below a certain threshold level on your Amazon EC2 instances. You can then create an automated procedure to ensure that you always have the right number of instances to match the amount of traffic you have. You can also use Amazon CloudWatch to diagnose problems by looking at system performance before and after a problem occurs. Amazon CloudWatch helps you identify the cause and verify your fix by tracking performance in real time. For example, you can set up Amazon CloudWatch to email you right away when your application slows down, to go back and discover that a particular database was being overloaded, and later to watch response times come back up to speed. For more information about creating CloudWatch alarms, go to [Creating CloudWatch Alarms](#) in the *Amazon CloudWatch Developer Guide*.

In the previous task, we created an Auto Scaling policy to scale out the number of instances. In this task, you need to associate that Auto Scaling policy with an alarm action to make changes to your resources. This topic walks you through how to create a CloudWatch alarm to alert the application when this threshold is breached. To save time during this walkthrough, we'll just create one alarm; however, you can apply the same procedure create other alarms. For example, you could create another alarm to scale in your instances. For more information about Amazon CloudWatch, see the [Amazon CloudWatch](#) details page.

To create an Amazon CloudWatch alarm

1. Select a metric for your alarm.
 - a. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
 - b. In the top navigation bar, make sure **US East (N. Virginia)** is selected in the region selector.
 - c. In the navigation pane, select **Alarm** under **Alarms**.
 - d. Click **Create Alarm**.
 - e. On the **Select Metric** page of the **Create Alarm Wizard**, select **EC2: Aggregated by Auto Scaling Group** from the **Viewing** drop-down menu.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 9: Create a CloudWatch Alarm

Create Alarm Wizard [Cancel]

SELECT METRIC | DEFINE ALARM | CONFIGURE ACTIONS | REVIEW

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

Statistic: Average
Period: 5 Minutes

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

CPUUtilization (Percent)

Viewing: EC2: Aggregated by Auto Scaling Group

MyAutoScalingGroup3	DiskReadOps
MyAutoScalingGroup3	DiskWriteBytes
MyAutoScalingGroup3	DiskWriteOps
MyAutoScalingGroup3	NetworkIn
MyAutoScalingGroup3	NetworkOut
MyAutoScalingGroup	CPUUtilization
MyAutoScalingGroup	DiskReadBytes
MyAutoScalingGroup	DiskReadOps
MyAutoScalingGroup	DiskWriteBytes
MyAutoScalingGroup	DiskWriteOps
MyAutoScalingGroup	NetworkIn
MyAutoScalingGroup	NetworkOut

Continue

- f. Click the **MyAutoScalingGroup/CPU Utilization** row.
 - g. Click **Continue**.
2. Define the alarm.
- a. On the **Define Alarm** page of the **Create Alarm** wizard, type **MyHighCPUALarm** in the **Name** box.
 - b. Type a description in the **Description** box.
 - c. In the **Define Alarm Threshold** section, select **>** and type **60** in the first box and **10** in the minutes box for this example. For your application, you can do some load testing to see what value makes the most sense for your application.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 9: Create a CloudWatch Alarm

Create Alarm Wizard Cancel

SELECT METRIC **DEFINE ALARM** **CONFIGURE ACTIONS** **REVIEW**

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

Identify Your Alarm
Assign your alarm a name and description.

Name: MyHighCPUAlarm
Description: Alarm triggered when CPU utilization is high

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

This alarm will enter the ALARM state when CPUUtilization is **>** **60** for **10** minutes.

Metric: CPUUtilization
Period: 5 Minutes
Statistic: Average

CPUUtilization (Percent)

100
75
50
25
0

3/21 3/21 3/21 3/21 3/21
14:00 15:00 16:00 17:00 18:00 19:00

[< Back](#) [Continue >](#)

- d. Click **Continue**.
3. Define your actions.
 - a. On the **Configure Actions** page of the **Create Alarm** wizard, select **Alarm** from the **When Alarm state is** drop-down menu.
 - b. Select **Auto Scaling Policy** from the **Take Action** drop-down menu.
 - c. Select **MyAutoScalingGroup** from the **Auto Scaling Group** drop-down menu.
 - d. Select **MyScaleUpPolicy (Add 1 instance)** from the **Policy** drop-down menu.
 - e. Click **Add Action**.
 - f. Select **Alarm** from the **When Alarm state is** drop-down menu.
 - g. Select **Send Notification** from the **Take Action** drop-down menu.
 - h. For topic, select **Create New Email Topic**. Then type a topic name in the **Topic** box.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 9: Create a CloudWatch Alarm

Create Alarm Wizard [Cancel X]

SELECT METRIC DEFINE ALARM **CONFIGURE ACTIONS** REVIEW

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

You can define multiple actions for a single alarm. For example, you may want to scale out your fleet and send an email to your pager when this alarm enters the ALARM state, and then send another all-clear email when it returns to the OK state.

Define Your Actions

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

When Alarm state is	Take action	Action details	
ALARM	Auto Scaling Policy	Auto Scaling Group: MyAutoScalingGroup Policy: MyScaleUpPolicy (Add 1)	REMOVE
ALARM	Send Notification	Topic: MyHighCPUAlarmTopic Email(s): <input type="text"/>	ADD ACTION

A topic is a communication channel that can be reused across Send Notification actions. Please enter a new topic name and a list of comma-separated email addresses.

< Back Continue >

- Type an email address in the **Email(s)** box.
 - Click **Add Action**.
 - Click **Continue**.
4. In the **Review** page, click **Create Alarm**.

Create Alarm Wizard [Cancel X]

SELECT METRIC DEFINE ALARM CONFIGURE ACTIONS **REVIEW**

If you want to make any changes to this alarm, click **Back** or select a step on the right to edit.

Alarm Definition [Edit Definition]

Name: MyHighCPUAlarm
Description: Alarm triggered when CPU utilization is high
In ALARM state when: the value is > 60 for 10 minutes

Metric [Edit Metric]

Namespace: AWS/EC2
MetricName: CPUUtilization
AutoScalingGroupName: MyAutoScalingGroup
Period / Statistic: 5 Minutes / Average

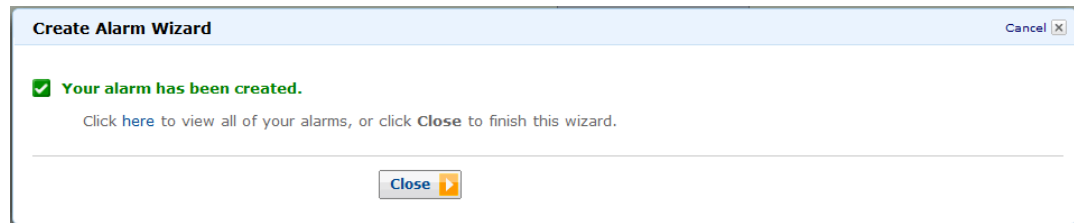
Alarm Actions [Edit Actions]

Actions:

- When alarm state is "ALARM"
Action Type: Auto Scaling Policy
Action: Use policy MyScaleUpPolicy (Add 1 instance) for group MyAutoScalingGroup
- When alarm state is "ALARM"
Action Type: Send Notification to New Topic
Action: Notify topic: MyHighCPUAlarmTopic (janedoe@example.com)

< Back Create Alarm >

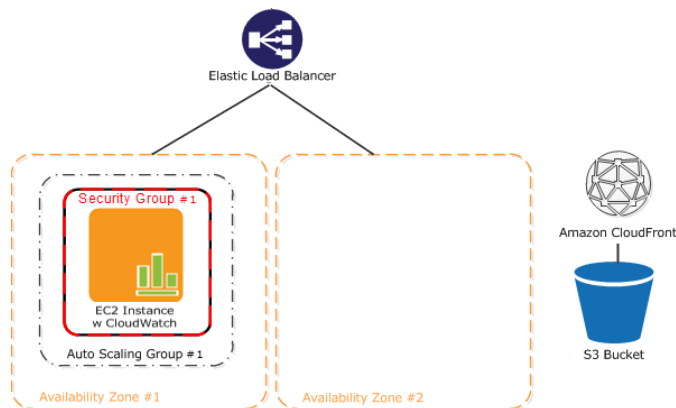
5. Click **Close**.



Your new alarm now appears in the list. When you create your MyScaleDownPolicy, you can create another alarm using the same steps.

Where You're At

Here's where you are at in building your architecture.



Next, let's add a database to the web application. Amazon provides a couple of database options, but for this example, we'll use [Amazon Relational Database Service \(Amazon RDS\)](#) because it's easy to operate and relieves us from the database administrative overhead.

Step 10: Add Amazon RDS

Topics

- [Create a DB Security Group \(p. 28\)](#)
- [Authorize Access \(p. 28\)](#)
- [Launch an Instance \(p. 29\)](#)

Now we are ready to add Amazon Relational Database (Amazon RDS) to our architecture. In this step we will launch a Multi-AZ RDS instance. When you create or modify your DB Instance to run as a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. Updates to your DB Instance are synchronously replicated across Availability Zones to the standby in order to keep both in sync and protect your latest database updates against DB Instance failure. During certain types of planned maintenance, or in the unlikely event of DB Instance failure or Availability Zone failure, Amazon RDS will automatically fail over to the standby so that you can resume database writes and reads as soon as the standby is promoted. Since the name record for your DB Instance remains the same, your application can resume database operation without the need for manual administrative intervention. With Multi-AZ deployments, replication is transparent: you do not interact directly with the standby, and it cannot be used to serve read traffic.

Important

The DB Instance you're about to launch will be live (and not running in a sandbox). You will incur the standard Amazon RDS usage fees for the instance until you terminate it. The total charges will be minimal (typically less than a dollar) if you complete the exercise described here in one sitting and terminate your DB Instance when you are finished. For more information about Amazon RDS usage rates, go to the [Amazon RDS product page](#).

Note

This is an optional step, so if you would like to skip this step, you can continue on to [Step 11: Deploy Your Application](#) (p. 33).

To set up your Amazon RDS database you need to do the following:

- Create a DB security group
- Authorize your DB instance
- Launch a DB instance

Create a DB Security Group

To create a DB Security Group, you need to provide a name and a description.

To create a DB Security Group

1. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. Make sure **US East (N. Virginia)** is selected in the region selector of the navigation bar.
3. In the left navigation pane, click **Security Groups**.
4. Click the **Create DB Security Group** button.
5. Type the name of the new DB security group. For this example, type **mydbsecuritygroup**.
6. Type a description for the new DB Security Group in the **Description** text box.
7. Click **Yes, Create**.

Now you're ready to authorize access to the Amazon EC2 security group.

Authorize Access

Now you will need to grant your Amazon EC2 security group access to your DB Security Group.

To authorize your Amazon EC2 security group for access to your DB Security Group

1. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. Make sure **US East (N. Virginia)** is selected in the region selector of the navigation bar.
3. In the left navigation pane, click **Security Groups**.
4. Select **mydbsecuritygroup**.
5. In the lower pane, select **EC2 Security Group** for the **Connection Type**.
6. Your AWS Account ID appears in the right half of the lower pane. If you want to authorize a different AWS ID to use this DB Security Group, select **Another account**, and then type your ID in the **AWS Account ID** box.

Note

Make sure to remove the hyphens when you type your account ID.

7. For the **EC2 Security Group Name**, select **webappsecuritygroup**.

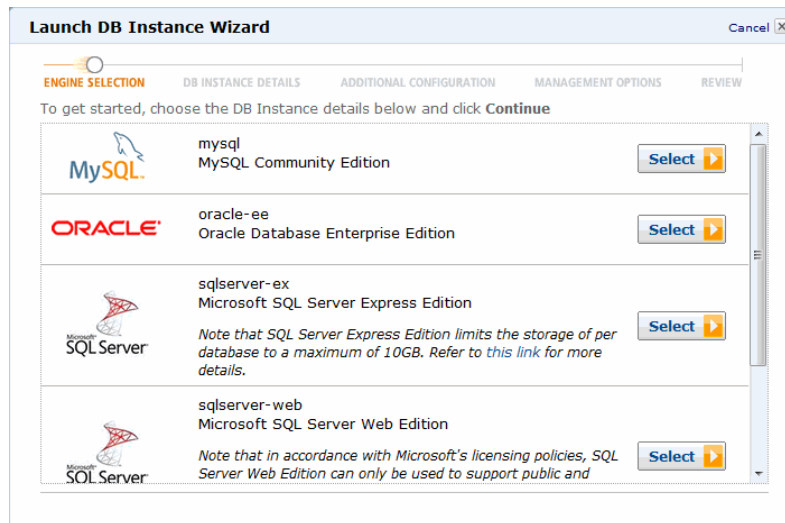
8. Click **Authorize**.

The authorization will take a few moments. After the security group has been authorized, the **Status** column in the top pane will say **authorized**. Move on to the next step to launch your first Amazon RDS database.

Launch an Instance

To launch an instance

1. Start the launch wizard:
 - a. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
 - b. Make sure **US East (N. Virginia)** is selected in the region selector in the navigation bar.
 - c. In the left navigation pane, click **DB instances**.
 - d. In the **Amazon RDS Console Dashboard**, click **Launch a DB Instance**.



2. Click **Select** next to the **MySQL Community Edition**.
3. On the **DB Instance Details** page, specify your DB instance details as shown in the following table. Then click **Continue**.

For this parameter...	Do this
License Model	Keep the default: general-public-license .
DB Engine Version	Keep the default: 5.5.27 (default) .
DB Instance Class	Select db.m1.small . The DB Instance class defines the CPU and memory capacity of your DB instance.
Multi-AZ Deployment	Choose Yes . Although the Multi-AZ deployment is more expensive, it is a best practice.

Getting Started with AWS Web Application Hosting for Microsoft Windows Launch an Instance

For this parameter...	Do this
Auto Minor Version Upgrade	Keep the default setting of Yes for this example. The Auto Minor Version Upgrade option enables your DB Instance to receive minor engine version upgrades automatically when they become available.
Allocated Storage	You can specify how much storage in gigabytes you want initially allocated for your DB Instance. For this example, type 5.
Use Provisioned IOPS	Leave the check box unselected. This option turns on Provisioned IOPS (I/O operations per second), a high-performance storage option in RDS that is optimized for I/O-intensive, transactional (OLTP) database workloads. For more information about high performance storage, see Provisioned IOPS .
DB Instance Identifier	The DB Instance is a name for your DB Instance that is unique for your account in a Region. Type <code>mydbinstance</code> in the DB Instance Identifier text box.
Master Username	Type a name for your master user in the Master Username text box. You use the master user name to log on to your DB Instance with all database privileges.
Master Password	Type a password for your master user in the Master User Password text box.

Important

You must specify a password containing from 4 to 16 alphanumeric characters only.

Launch DB Instance Wizard [Cancel]

ENGINE SELECTION **DB INSTANCE DETAILS** ADDITIONAL CONFIGURATION MANAGEMENT OPTIONS REVIEW

To get started, choose a DB engine below and click **Continue**

DB Engine: mysql
License Model: General Public License
DB Engine Version: MySQL 5.5.27 (default)
DB Instance Class: db.m1.small
Multi-AZ Deployment: Yes
Auto Minor Version Upgrade: ☒ Yes ☐ No

Provide the details for your RDS Database Instance.

Allocated Storage: 5 GB (Minimum: 5 GB, Maximum: 3072 GB) Higher allocated storage may improve IOPS performance.
Use Provisioned IOPS: ☐
DB Instance Identifier: mydbinstance (e.g. mydbinstance)
Master Username: awsuser (e.g. awsuser)
Master Password: (e.g. mypassword)

< Back **Continue**

4. Provide additional configuration information for your DB Instance:
 - a. Type `mydb` into the **Database Name** text box.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Launch an Instance

You provide a database name so that Amazon RDS will create a default database on your new DB Instance. If you skip this step, Amazon RDS will not create a database on your DB Instance.

- b. Select **mydbsecuritygroup** in the **DB Security Groups** box.

Launch DB Instance Wizard [Cancel]

ENGINE SELECTION DB INSTANCE DETAILS **ADDITIONAL CONFIGURATION** MANAGEMENT OPTIONS REVIEW

Provide the optional additional configuration details below.

Database Name: mydb (e.g. mydb)

Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.

Database Port: 3306

Choose a VPC: Not in VPC Only VPCs with a DB Subnet Group(s) are allowed

Availability Zone: No Preference MultiAZ deployment selected

Option Group: defaultmysql5-5

If you have custom DB Parameter Groups or DB Security Groups you would like to associate with this DB Instance, select them below, otherwise proceed with default settings.

Parameter Group: defaultmysql5.5

DB Security Group(s): default, mydbsecuritygroup

< Back Continue

- c. Accept the default values for the rest of the parameters available on this page, and then click **Continue**.

5. Use the **Management Options** page to specify backup and maintenance options for your DB Instance. For this example, accept the default values, and then click **Continue**.

Launch DB Instance Wizard [Cancel]

ENGINE SELECTION DB INSTANCE DETAILS ADDITIONAL CONFIGURATION **MANAGEMENT OPTIONS** REVIEW

Enabled Automatic Backups: Yes No

The number of days for which automated backups are retained.

Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to details [here](#).

Backup Retention Period: 1 days

The daily time range during which automated backups are created if automated backups are enabled

Backup Window: Select Window No Preference

The weekly time range (in UTC) during which system maintenance can occur.

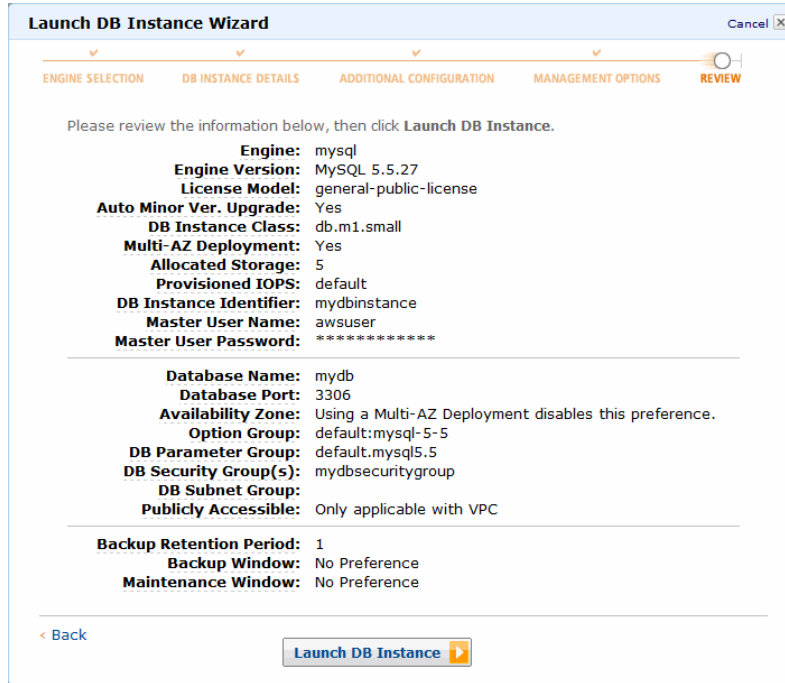
Maintenance Window: Select Window No Preference

< Back Continue

6. Review the options for your DB Instance. If you need to correct any options, click **Back** to return to previous pages and make corrections. When you're ready, click **Launch DB Instance** to launch your new DB Instance.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Launch an Instance

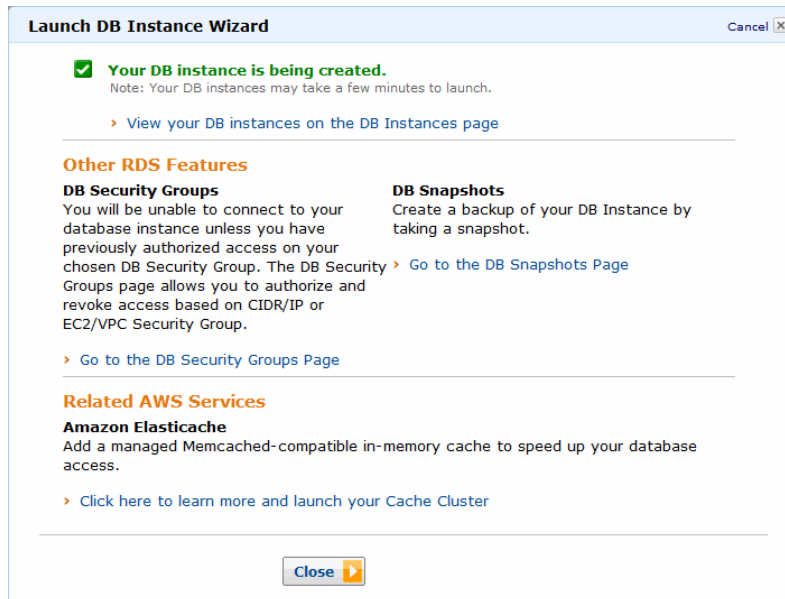


The screenshot shows the 'Launch DB Instance Wizard' in the 'REVIEW' step. The wizard has five tabs: ENGINE SELECTION, DB INSTANCE DETAILS, ADDITIONAL CONFIGURATION, MANAGEMENT OPTIONS, and REVIEW. The REVIEW tab is active, showing a summary of the configuration. The text 'Please review the information below, then click Launch DB Instance.' is at the top. The configuration details are as follows:

- Engine: mysql
- Engine Version: MySQL 5.5.27
- License Model: general-public-license
- Auto Minor Ver. Upgrade: Yes
- DB Instance Class: db.m1.small
- Multi-AZ Deployment: Yes
- Allocated Storage: 5
- Provisioned IOPS: default
- DB Instance Identifier: mydbinstance
- Master User Name: awsuser
- Master User Password: *****
- Database Name: mydb
- Database Port: 3306
- Availability Zone: Using a Multi-AZ Deployment disables this preference.
- Option Group: default:mysql-5-5
- DB Parameter Group: default:mysql5.5
- DB Security Group(s): mydbsecuritygroup
- DB Subnet Group:
- Publicly Accessible: Only applicable with VPC
- Backup Retention Period: 1
- Backup Window: No Preference
- Maintenance Window: No Preference

At the bottom, there is a '< Back' link and a 'Launch DB Instance' button with a right arrow.

7. Launching can take a few minutes to complete. When you see the notice that your instance is being created, click **Close**.



The screenshot shows the 'Launch DB Instance Wizard' after the instance has been launched. The title bar says 'Launch DB Instance Wizard' and there is a 'Cancel' button. A green checkmark icon is followed by the text 'Your DB instance is being created.' and a note: 'Note: Your DB instances may take a few minutes to launch.' Below this is a link: '> View your DB instances on the DB Instances page'. There are two sections: 'Other RDS Features' and 'Related AWS Services'. Under 'Other RDS Features', there are two columns: 'DB Security Groups' and 'DB Snapshots'. Under 'Related AWS Services', there is a section for 'Amazon ElastiCache'.

Other RDS Features

DB Security Groups You will be unable to connect to your database instance unless you have previously authorized access on your chosen DB Security Group. The DB Security Groups page allows you to authorize and revoke access based on CIDR/IP or EC2/VPC Security Group. > Go to the DB Security Groups Page	DB Snapshots Create a backup of your DB Instance by taking a snapshot. > Go to the DB Snapshots Page
--	---

Related AWS Services

Amazon ElastiCache
Add a managed Memcached-compatible in-memory cache to speed up your database access.
> Click here to learn more and launch your Cache Cluster

At the bottom, there is a 'Close' button with a right arrow.

Your DB instance appears in the list on this page with the status of **creating** until your DB Instance is created and ready for use.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 11: Deploy Your Application

Launch DB Instance



Show Monitoring

Instance Actions

Filter: All Instances

Search DB Instances...

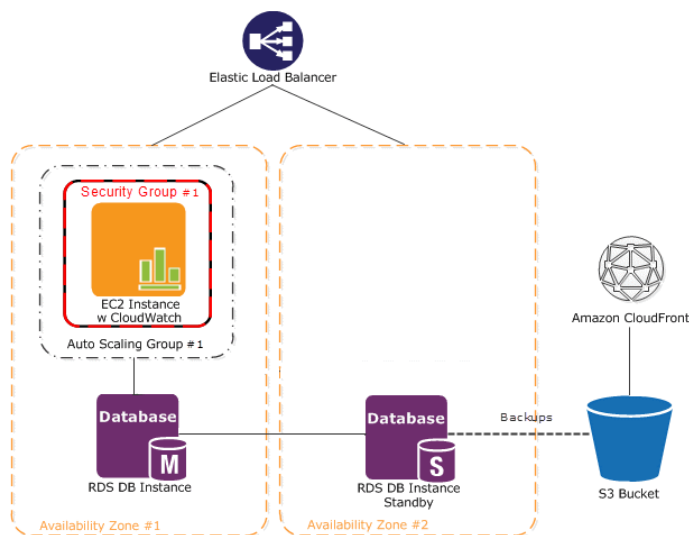
Viewing 9 of 9 DB Instances

	DB Instance	VPC	Multi-AZ	Class	Status	Storage	IOPS	Security Groups	Engine	Zone
	 mydbinstance	Yes		db.m1.small	creating	5 GB		mydbsecuritygroup (active)	mysql	us-east-1a

After your DB instance changes to the **available** state, you're billed for each hour or partial hour that you keep the DB Instance running (even if the DB Instance is idle).

Where You're At

Here's where you are at in building your architecture.



Now that you have launched your Amazon RDS database, you're ready to deploy your sample web application.

Step 11: Deploy Your Application

Topics

- [Connect to Your Amazon EC2 Instance \(p. 34\)](#)
- [Deploy Sample Application \(p. 36\)](#)
- [Modify the Application \(p. 37\)](#)

Now that you've created all of your AWS resources, it's time to deploy your application to your Amazon EC2 instance. In this step, you'll use a sample application that you can download, modify, and deploy to your Amazon EC2 instance. Amazon provides an AWS Toolkit for Visual Studio to help you build and deploy your .NET applications. For information, go to <http://aws.amazon.com/visualstudio/>.

Connect to Your Amazon EC2 Instance

To connect to a Windows instance, you must retrieve the initial administrator password first, and then use it with Remote Desktop. You'll need the contents of the private key file that you created (e.g., `mykeypair.pem`) in [Step 7: Create a Key Pair \(p. 19\)](#).

Note

It can take up to 30 minutes to get the original password from the time you launched your Amazon EC2 instance.

To connect to your Windows instance

1. Retrieve the auto-generated administrator password:
 - a. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
 - b. Make sure that **US East (N. Virginia)** is selected in the region selector of the navigation bar.
 - c. Locate the instance on the **Instances** page.
 - d. Right-click the instance and select **Get Windows Password**.

The **Retrieve Default Windows Administrator Password** dialog box is displayed (it might take a few minutes after the instance is launched before the password is available).

- e. Click **Browse** to locate the private key file.
- f. Click **Decrypt Password**.

The console displays the default administrator password for the instance.

- g. Save the password. You will need it to connect to the instance.

Note

This step only works for a new Amazon EC2 instance. Once you change the Administrator password or create a custom AMI, you need to remember the password.

2. In the EC2 console, make a note of the public DNS name of the instance. You will need it in the following steps.
3. Connect to the instance using Remote Desktop:
 - a. Start the Remote Desktop application (e.g., from the **Start** menu, point to **All Programs > Accessories**, and then click **Remote Desktop Connection**).

Note

Most modern Windows operating systems from Windows XP onward already include the Remote Desktop application. If you're using an old version of Windows, you can download the Remote Desktop application from the [Microsoft website](#).

- b. Click **Show Options** and ensure that you are not specifying a domain.
- c. Enter the public DNS name of the instance and click **Connect**.
- d. Log in using `Administrator` as the username and the administrator password you got in the previous task as the password.

Note

You can copy and paste the password using Ctrl+C and Ctrl+V.

You're now connected to your instance. You can work with it the same way you would work with any Windows server.

Getting Started with AWS Web Application Hosting for Microsoft Windows Connect to Your Amazon EC2 Instance

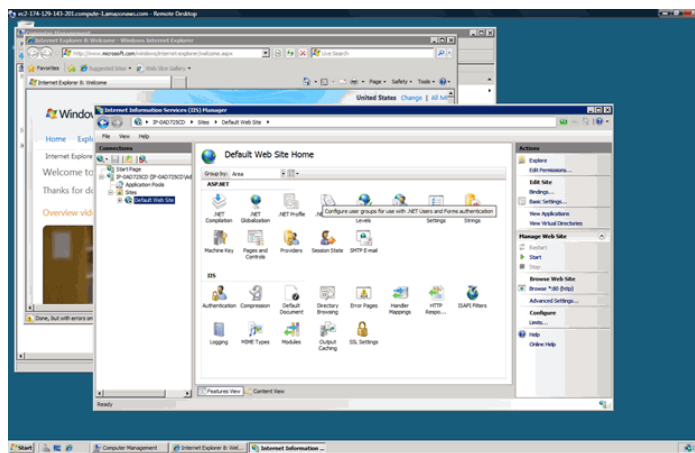
Caution

After you connect to any new Windows instance you've just launched, we recommend you change the Windows administrator password from the default value.

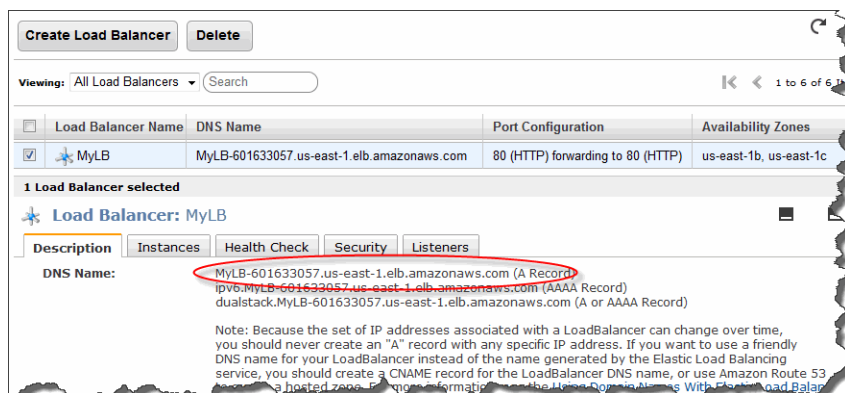
We will be updating this site; however start by making certain that you can connect to the server with Internet Information Services (IIS) Manager.

To connect to the server with IIS Manager

1. From your Amazon EC2 instance, click **Start**.
2. Click **Administrative Tools**.
3. Click **Internet Information Services (IIS) Manager**.
4. Expand the localhost node.
5. Expand the **Sites** node.



6. Right-click **Default Web Site**.
7. Select **Manage Web Site** and click **Start**.
8. In the **Load Balancers** page of the EC2 console, copy the public DNS address of your load balancer. Then paste it into your browser's address bar to make certain that you can connect to the site.



Now that we know our web server is working correctly, let's move on to the next step to deploy a sample application to our Amazon EC2 instance.

In this topic we will download a sample ASP.NET application, grant write access to the appropriate folders and files, and configure IIS to point to the new application on your Amazon EC2 instance.

1. In the Amazon EC2 instance, create a folder for your sample application in your root directory called ImageGallery, **c:\ImageGallery**.
2. Add the website location for the BaseImageGallery application to trusted sites on your Amazon EC2 instance
 - a. In the Amazon EC2 instance, run **Internet Explorer** from the Start menu.
 - b. Click the gear icon in top right corner.
 - c. Click **Internet Options**.
 - d. Click the **Security** tab.
 - e. Click **Trusted Sites** and then click **Sites**.
 - f. Add **https://s3.amazonaws.com** and click **Close**.
 - g. Click the **Advanced** tab.
 - h. Scroll down to Security.
 - i. Un-check **Do not save encrypted pages to disk** and click **OK**.
3. Download the application.
 - a. Using Internet Explorer in your Amazon EC2 instance, type **https://s3.amazonaws.com/aws-sdk-samples/.NET/BaseImageGallery.zip** in your browser's address bar.

This is a case-sensitive URL. This zip file is a .NET application; however we are not going to delve into the details of .NET. Instead we'll “paste and modify” the application.

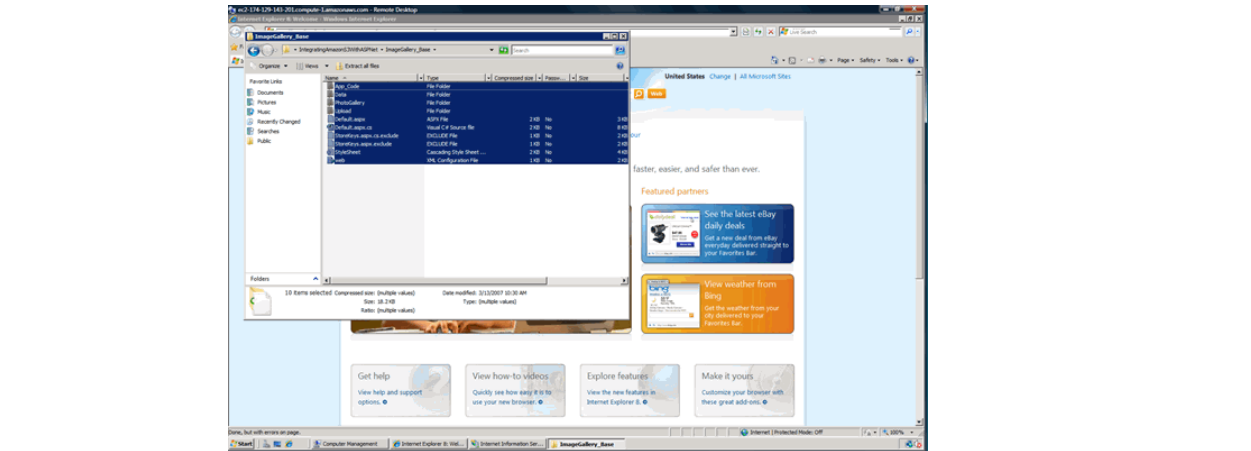
b. When the **File Download** dialog box appears, click **Save As**. Save the file to **c:\ImageGallery**.

- Note**

If you receive an error that Internet Explorer is unable to download the file, close the error dialog and try downloading the application again inside the same browser window.

- c. Paste the contents of the ImageGallery folder (found inside the zip file) into this directory, so

- that individual files sit directly under C:\ImageGallery.

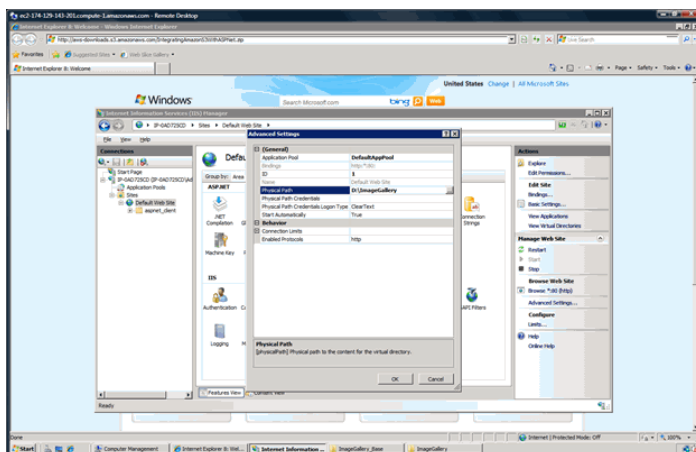


Getting Started with AWS Web Application Hosting for Microsoft Windows

Modify the Application

To modify the physical path for your default website

1. Open IIS Manager on your Amazon EC2 instance and navigate to the **Default Web Site** again.
2. Right-click **Default Web Site**.
3. Select **Manage Web Site** and click **Advanced Settings**.
4. In the **Physical Path** box, type `c:\ImageGallery` and click **OK**.



Modify the Application

Now we are ready to modify the application so that our images are published to Amazon S3 automatically and we use Amazon RDS to index our content stored in Amazon S3. We will modify the code from the web server so we don't expose the webmaster's S3 credentials to end users.

Now we'll modify the application in the following ways:

- Update your web.config file
- Modify your code to point to your Amazon CloudFront distribution or your Amazon S3 bucket
- Modify your code to point to your Amazon RDS file

To update your web.config file

- In the `c:\ImageGallery` folder on your Amazon EC2 instance, open the `web.config` file and find the `appSettings` section. Specify your AWS credentials using one of the methods described in [Using Credentials in an Application](#).

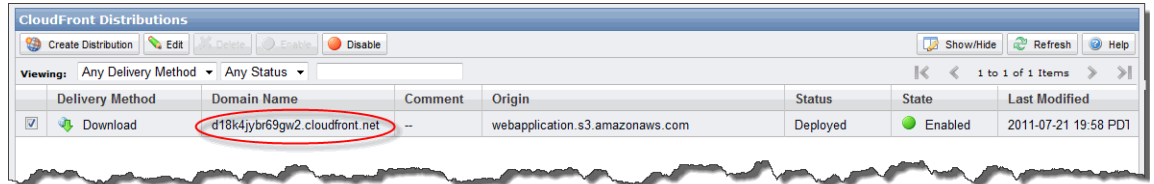
For more information about managing AWS credentials, see [Best Practices for Managing AWS Access Keys](#).

To modify code to point to an Amazon CloudFront distribution or an Amazon S3 bucket

1. In the `c:\ImageGallery` folder in your Amazon EC2 instance, open `default.aspx.cs` and replace `http://your-non-dotted-bucket-name.s3.amazonaws.com/` with the name of your Amazon CloudFront distribution or your Amazon S3 bucket. For instance, in [Step 4: Create a CloudFront Distribution \(p. 10\)](#), our Amazon CloudFront distribution is `d1j6fvxrgjehco.cloudfront.net`.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Modify the Application



2. A few lines below that, replace `your-non-dotted-bucket-name` with the name of your Amazon S3 bucket. For instance, in [Step 3: Create an Amazon S3 Bucket \(p. 7\)](#), our Amazon S3 bucket is `webapplication`.

Note

Make sure you do not forget the trailing slash for your Amazon S3 bucket or your Amazon CloudFront distribution.

```
public partial class _Default : System.Web.UI.Page
{
    public string PhotoGalleryBaseUrl = "http://d1j6fvxrgjehco.cloudfront.net/";

    ...
    private const string ImageBucketName = "webapplication";
}
```

If you skipped creating the CloudFront distribution, then you would enter your Amazon S3 bucket as in the following example:

```
public partial class _Default : System.Web.UI.Page
{
    public string PhotoGalleryBaseUrl = "http://webapplication.s3.amazonaws.com/";

    ...
    private const string ImageBucketName = "webapplication";
}
```

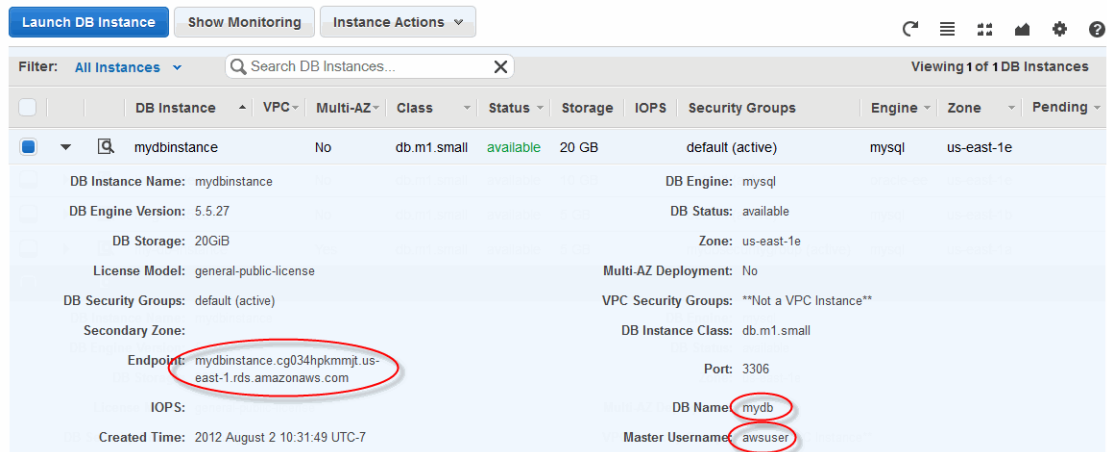
Note

This code publishes images to your Amazon S3 bucket and sets the permissions on each file to be public, that is readable by everyone.

To modify code to point to your Amazon RDS database

1. In the `c:\ImageGallery` folder in your Amazon EC2 instance, open `default.aspx.cs` if it's not open already. Replace the text for the following variables with the actual values. These values will be used to build your connection string using MySQL Connector/.NET. Many of these can be found on the **DB Instances** page of the RDS console at <https://console.aws.amazon.com/rds/>
 - Database endpoint
 - Database master user ID
 - Database master password
 - Database name

Getting Started with AWS Web Application Hosting for Microsoft Windows Modify the Application



```
public partial class _Default : System.Web.UI.Page
{
    private string dbinstance = "mydbinstance.cgwxy4tle0xb.us-east-1.rds.amazonaws.com";
    private string userid = "awsuser";
    private string password = "mypassword";
    private string database = "mydb";
}
```

2. Save and close the file.

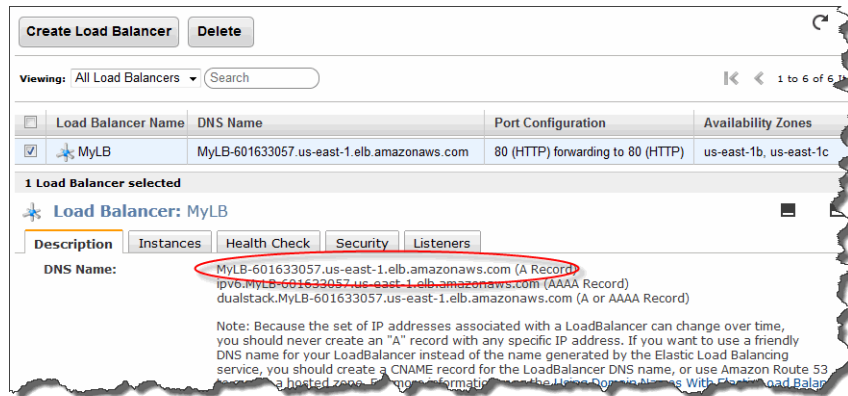
Now it's time to deploy your sample website.

To restart your website

1. In the **IIS Manager**, navigate to **Default Web Site**.
2. Right-click **Default Web Site**.
3. Select **Manage Web Site** and then click **Restart**.
4. Find the DNS address of the load balancer.
 - a. In the AWS Management Console, click the **EC2** tab.
 - b. In the navigation pane, click **Load Balancers**.
 - c. In the list of load balancers, click **MyLB** and note the public DNS in the **Description** tab.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 12: Create a Custom AMI



- d. Type the DNS address of the load balancer in your web browser and verify that your application still works.
- e. You will not see any images on the site because there are none currently in the Amazon RDS database. Upload images less than or equal to 4 MB to the website and check that they display correctly.

Note

If you have trouble displaying the images, you may not be connecting to your Amazon RDS database; make sure you typed your Amazon EC2 security group correctly when you authorized access for database security group. If you still have trouble, try re-authorizing your Amazon EC2 security group.

Congratulations! You have successfully deployed a sample web application using Amazon Web Services. Now, in the future if you decide you want to launch more instances you don't want to customize each one. Let's move on to the step to create a custom Amazon Machine Image (AMI) with all our configuration changes.

Step 12: Create a Custom AMI

Now that we have customized our Amazon EC2 instance, we can save this Amazon Machine Image (AMI) and launch future environments with this saved configuration using AWS CloudFormation. This is an optional step. If you prefer to finish the tutorial now, you can skip ahead to clean up your AWS resources in [Step 14: Clean Up \(p. 50\)](#).

To create an AMI from a running Amazon EBS-backed instance

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Make sure that **US East (N. Virginia)** is selected in the region selector of the navigation bar.
3. Click **Instances** in the navigation pane.
4. On the **Instances** page, right-click your running instance and select **Create Image**.

The **Create Image** dialog box opens.

5. Fill in a unique image name and an optional description of the image (up to 255 characters), and click **Create Image**.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Step 13: Launch New Environments Using AWS CloudFormation

Tip

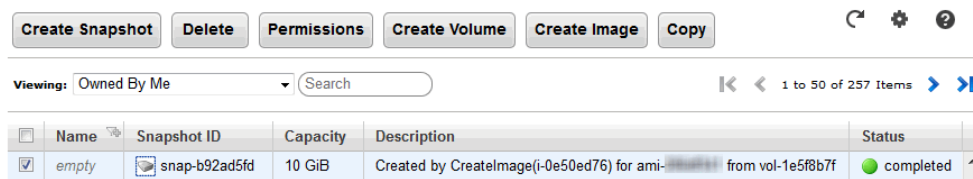
If you're familiar with Amazon EC2 instance store-backed AMIs, the image name replaces the manifest name (e.g., `s3_bucket/something_of_your_choice.manifest.xml`), which uniquely identifies each Amazon EC2 instance store-backed AMI.

Amazon EC2 powers down the instance, takes images of any volumes that were attached, creates and registers the AMI, and then reboots the instance.

6. Go to the **AMIs** page and view the AMI's status. While the new AMI is being created, its status is *pending*.

It takes a few minutes for the whole process to finish.

7. Once your new AMI's status is *available*, go to the **Snapshots** page and view the new snapshot that was created for the new AMI. Any instance you launch from the new AMI uses this snapshot for its root device volume. You could update your Auto Scaling group with the new AMI, however we will do this as part of the AWS CloudFormation step.



We've taken a lot of steps so far to create all of our AWS resources, deploy our application, and customize our AMI. Wouldn't it be great if we could save all of this information and launch new environments without having to manually configure everything again? We can! [AWS CloudFormation](#) is a way to launch environments easily. That is, when you launch an AWS CloudFormation environment, you are able to launch specific AMIs with particular key pairs, on pre-defined instance sizes, and behind load balancers. And if any portion of your environment fails to launch, the environment rolls itself back, terminating all the pieces along the way. Let's move on to the next topic to begin using AWS CloudFormation.

Step 13: Launch New Environments Using AWS CloudFormation

Topics

- [Create an AWS CloudFormation Template \(p. 42\)](#)
- [Modify an AWS CloudFormation Template \(p. 45\)](#)
- [Create an AWS CloudFormation Stack \(p. 47\)](#)

You can use AWS CloudFormation to create and provision AWS infrastructure deployments predictably and repeatedly. Use AWS CloudFormation to build highly reliable, highly scalable, cost-effective applications without worrying about creating and configuring the underlying AWS infrastructure. AWS consists of template files you use to create and delete collections of resources as a single unit (an AWS CloudFormation stack). Using AWS CloudFormation you can leverage other services such as Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Block Store (Amazon EBS), Amazon Simple Notification Service (Amazon SNS), Elastic Load Balancing, and Auto Scaling.

In this example, we'll use the CloudFormer tool to generate a template based on the AWS resources we just created. CloudFormer is intended to create a starting point for your template. After you create the template, you'll customize the template to launch a new environment with multiple instances spanning multiple Availability Zones to enable a fault-tolerant architecture.

This is an optional step. If you want to skip this step, you can move on to [Step 14: Clean Up \(p. 50\)](#) to begin deleting your resources.

Create an AWS CloudFormation Template

First, you'll need to create a template based on the resources you just created. You'll use a tool called CloudFormer that collects information about all your running resources and creates a template. CloudFormer is a prototype that helps you get started. You'll then make some tweaks to the template before you create your new stack. Visit the [AWS Forums](#) to learn more and to run the tool.

After generating the template and making a few tweaks, you may have something that looks like the following.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Resources": {
    "elbMyLB": {
      "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
      "Properties": {
        "AvailabilityZones": [
          "us-east-1b",
          "us-east-1c"
        ],
        "HealthCheck": {
          "HealthyThreshold": "2",
          "Interval": "30",
          "Target": "HTTP:80/",
          "Timeout": "5",
          "UnhealthyThreshold": "2"
        },
        "Listeners": [
          {
            "InstancePort": "80",
            "LoadBalancerPort": "80",
            "Protocol": "HTTP",
            "PolicyNames": [

            ]
          }
        ]
      }
    },
    "distd18k4jybr69gw2cloudfrontnet": {
      "Type": "AWS::CloudFront::Distribution",
      "Properties": {
        "DistributionConfig": {
          "S3Origin": {
            "DNSName": "webapplication.s3.amazonaws.com"
          },
          "Enabled": "true",
          "Logging": {
            "Bucket": "webapplication.s3.amazonaws.com",
            "Prefix": "webapp-logging/"
          }
        }
      }
    }
  }
}
```

```
    },
    "asgMyAutoScalingGroup": {
      "Type": "AWS::AutoScaling::AutoScalingGroup",
      "Properties": {
        "AvailabilityZones": [
          "us-east-1b",
          "us-east-1c"
        ],
        "Cooldown": "300",
        "DesiredCapacity": "1",
        "MaxSize": "1",
        "MinSize": "1",
        "LaunchConfigurationName": {
          "Ref": "lcMyLC"
        },
      },
    },
    "LoadBalancerNames": [
      {
        "Ref": "elbMyLB"
      }
    ]
  },
  "lcMyLC": {
    "Type": "AWS::AutoScaling::LaunchConfiguration",
    "Properties": {
      "ImageId": "ami-c082e3a9",
      "InstanceType": "t1.micro",
      "KeyName": "mykeypair",
      "SecurityGroups": [
        {
          "Ref": "sgwebappsecuritygroup"
        }
      ]
    }
  },
  "aspMyScaleUpPolicy": {
    "Type": "AWS::AutoScaling::ScalingPolicy",
    "Properties": {
      "AdjustmentType": "ChangeInCapacity",
      "AutoScalingGroupName": { "Ref": "asgMyAutoScalingGroup" },
      "Cooldown": "300",
      "ScalingAdjustment": "1"
    }
  },
  "cwCPUALarmHigh": {
    "Type": "AWS::CloudWatch::Alarm",
    "Properties": {
      "AlarmDescription": "Scale-up if CPU > 60% for 10 minutes",
      "MetricName": "CPUUtilization",
      "Namespace": "AWS/EC2",
      "Statistic": "Average",
      "Period": "300",
      "EvaluationPeriods": "2",
      "Threshold": "60",
    }
  },
}
```

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Create an AWS CloudFormation Template**

```
        "AlarmActions": [ { "Ref": "aspMyScaleUpPolicy" } ],
        "Dimensions": [
            {
                "Name": "AutoScalingGroupName",
                "Value": { "Ref": "asgMyAutoScalingGroup" }
            }
        ],
        "ComparisonOperator": "GreaterThanThreshold"
    }
},
"rdsmydbinstance": {
    "Type": "AWS::RDS::DBInstance",
    "Properties": {
        "AllocatedStorage": "5",
        "BackupRetentionPeriod": "1",
        "DBInstanceClass": "db.m1.small",
        "DBName": "MyDatabase",
        "DBParameterGroupName": "default.mysql5.1",
        "Engine": "mysql",
        "EngineVersion": "5.1.57",
        "MasterUsername": "awsuser",
        "MasterUserPassword": "awsuser",
        "Port": "3306",
        "PreferredBackupWindow": "07:00-07:30",
        "PreferredMaintenanceWindow": "sat:04:00-sat:04:30",
        "MultiAZ": "true",
        "DBSecurityGroups": [
            {
                "Ref": "dbsgmydbsecuritygroup"
            }
        ]
    }
},
"s3webapplication": {
    "Type": "AWS::S3::Bucket"
},
"sgwebappsecuritygroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
        "GroupDescription": "for web app",
        "SecurityGroupIngress": [
            {
                "IpProtocol": "tcp",
                "FromPort": "80",
                "ToPort": "80",
                "SourceSecurityGroupName": "amazon-elb-sg",
                "SourceSecurityGroupOwnerId": "amazon-elb"
            },
            {
                "IpProtocol": "tcp",
                "FromPort": "3389",
                "ToPort": "3389",
                "CidrIp": "0.0.0.0/0"
            }
        ]
    }
},
"dbsgmydbsecuritygroup": {
```

```
"Type": "AWS::RDS::DBSecurityGroup",
"Properties": {
  "GroupDescription": "security group for my web app",
  "DBSecurityGroupIngress": [
    {
      "EC2SecurityGroupName": {
        "Ref": "sgwebappsecuritygroup"
      },
      "EC2SecurityGroupOwnerId": "123456789012"
    }
  ]
},
"Description": ""
}
```

You'll want to make a couple of changes to this template before you launch your new environment. In this tutorial, you only launched one Amazon EC2 instance. However, it's a best practice to launch multiple instances across multiple Availability Zones; you'll want to update your Auto Scaling group to launch multiple instances. You'll also want to launch a new environment with your custom AMI. Finally, You'll update your database information to include your database name and password.

Modify an AWS CloudFormation Template

Now that the template has been created, let's modify it so that you can launch a new environment with the custom AMI so that you will have multiple instances spanned across multiple Availability Zones.

To launch a new stack with a modified template

1. Open the template you created using CloudFormer.
2. Update the format of the **CloudDistribution** group to include **DistributionConfig** and the **Logging** portion.

```
"distdl18k4jybr69gw2cloudfrontnet" : {
  "Type" : "AWS::CloudFront::Distribution",
  "Properties" : {
    "DistributionConfig" : {
      "S3Origin" : {
        "DNSName": "webapplication.s3.amazonaws.com"
      },
      "Enabled" : "true",
      "Logging" : {
        "Bucket" : "webapplication.s3.amazonaws.com",
        "Prefix" : "webapp-logging/"
      }
    }
  }
},
```

3. Update the **Min Size**, **Max Size**, and **Desired Capacity** in the **Auto Scaling** group to 2.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Modify an AWS CloudFormation Template

```
"asgMyAutoScalingGroup": {
  "Type": "AWS::AutoScaling::AutoScalingGroup",
  "Properties": {
    "AvailabilityZones": [
      "us-east-1b",
      "us-east-1c"
    ],
    "Cooldown": "300",
    "DesiredCapacity": "2",
    "MaxSize": "2",
    "MinSize": "2",
    "LaunchConfigurationName": {
      "Ref": "lcMyLC"
    },
    "LoadBalancerNames": [
      {
        "Ref": "elbMyLB"
      }
    ]
  }
},
```

4. Update the **Image ID** in the **Launch Configuration** group to the custom AMI that you created in [Step 12: Create a Custom AMI \(p. 40\)](#).

Note

Your AMI ID will be different than the one you see below.

```
"lcMyLC": {
  "Type": "AWS::AutoScaling::LaunchConfiguration",
  "Properties": {
    "ImageId": "ami-576ca43e",
    "InstanceType": "t1.micro",
    "KeyName": "mykeypair",
    "SecurityGroups": [
      {
        "Ref": "sgwebappsecuritygroup"
      }
    ]
  }
},
```

5. Update the following parameters in the **Database** group.
 - Update **DBName** to **mydb**.
 - Update **MasterUserPassword** to the master password you specified in [Step 10: Add Amazon RDS \(p. 27\)](#).

```
"rdsawsworkshop": {
  "Type": "AWS::RDS::DBInstance",
```

```
"Properties": {
  "AllocatedStorage": "5",
  "BackupRetentionPeriod": "1",
  "DBInstanceClass": "db.m1.small",
  "DBName": "mydb",
  "DBParameterGroupName": "default.mysql5.1",
  "Engine": "mysql",
  "EngineVersion": "5.1.57",
  "MasterUsername": "awsuser",
  "MasterUserPassword": "mypassword",
  "Port": "3306",
  "PreferredBackupWindow": "08:30-09:00",
  "PreferredMaintenanceWindow": "fri:03:30-fri:04:00",
  "MultiAZ": "true",
  "DBSecurityGroups": [
    {
      "Ref": "dbsgmydbsecuritygroup"
    }
  ]
},
```

Now that you have modified the template, let's move on to the next step to launch your new environment using your template.

Create an AWS CloudFormation Stack

Now that you have modified your AWS CloudFormation template, let's create a new stack to launch your new environment. Before you launch your new stack, you can verify that it works by cleaning up all your AWS resources *except* for your key pair and custom AMI. For instructions on how to clean up your resources see [Step 14: Clean Up \(p. 50\)](#).

Note

AWS CloudFormation is a free service. However, you are charged for the AWS resources you include in your stacks at the current rates for each. For more information about AWS pricing, go to the detail page for each product on <http://aws.amazon.com/pricing>.

To create an AWS CloudFormation stack

1. Launch the Create Stack wizard.
 - a. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
 - b. Make sure **US East (N. Virginia)** is selected in the region selector in the top navigation bar.
 - c. Click **Create Stack**.
2. Select a template.
 - a. On the **SELECT TEMPLATE** page of the **Create Stack** wizard, type a stack name in the **Stack Name** box.
 - b. Click **Provide a Template URL** and type the location where you saved your modified template.
 - c. Click **Show Advanced Options**.
 - d. Select **Create a new SNS topic** from the **Amazon SNS Topic** drop-down menu. You will receive email notifications when resources are created and when they are deleted.

Getting Started with AWS Web Application Hosting for Microsoft Windows

Create an AWS CloudFormation Stack

- e. Type **CRM** in the **New topic name** box.
- f. Type your email address in the **Email** box and accept the rest of the default values.

Create Stack Cancel

SELECT TEMPLATE SPECIFY PARAMETERS ADD TAGS REVIEW

AWS CloudFormation gives you an easier way to create a collection of related AWS resources (a stack) by describing your requirements in a template. To create a stack, fill in the name for your stack and select a template. You may chose one of the sample templates to get started quickly, or one of your own templates stored in S3 or on your local hard drive.

Stack Name:
MyCloudFormationStack

Template:

☐ Use a sample template

☐ Upload a Template File

☒ Provide a Template URL
zonaws.com/aws-templates/cloudformer.template

☒ Show Advanced Options

Notifications (optional):
Amazon SNS Topic: Create a new SNS topic
New topic name: CRM Email: janedoe@example.co

Creation Timeout (minutes): none

Rollback on Failure: ☒ Yes ☐ No

Continue

- g. Click **Continue**.

Create Stack Cancel

SELECT TEMPLATE SPECIFY PARAMETERS REVIEW

Please review the information below, then click Create Stack.

Stack Information Edit Stack

Stack Name: MyCloudFormationStack

Stack Description:

Template: https://s3.amazonaws.com/aws-templates/cloudformer.template

Notification Edit Notification

Notification: arn:aws:sns:us-east-1:[redacted]:CRM

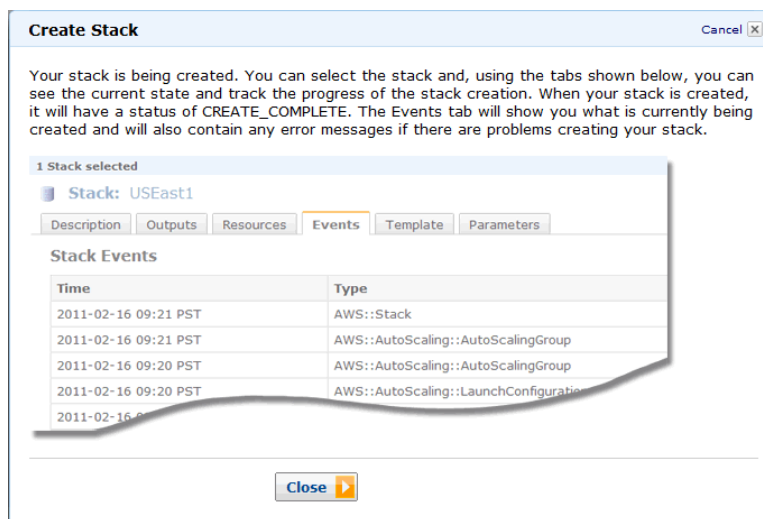
Creation Timeout: none

Rollback on Failure: true

[< Back](#) **Create Stack**

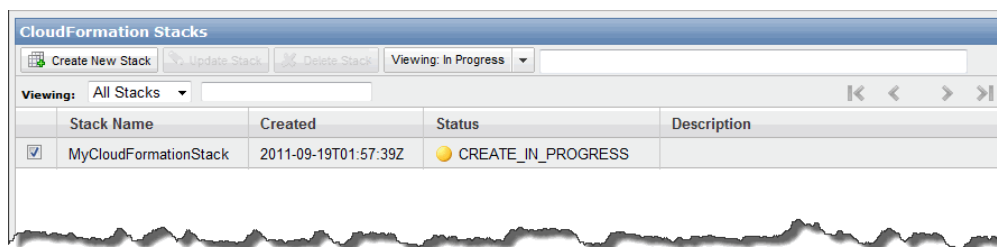
3. Review your settings. You can make changes to the settings by clicking the edit link for a specific step in the process.
4. Click **Create Stack**.

Getting Started with AWS Web Application Hosting for Microsoft Windows Create an AWS CloudFormation Stack



5. Click **Close** to close the confirmation window.

The confirmation window closes, returning you to the **CloudFormation** page. Your new AWS CloudFormation template appears in the list with the status set to **CREATE_IN_PROGRESS**.



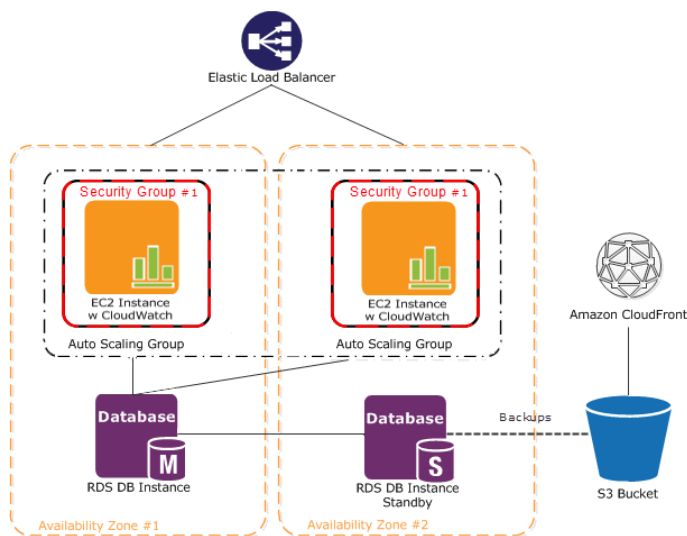
Note

Your stack will take a few minutes to create. Make sure to click **Refresh** in the Stacks page to see when the template has successfully been created.

Where You're At

Congratulations! You have just launched your new environment using the AWS resources you created in this tutorial. Your Elastic Load Balancer is now pointing to both of your Amazon EC2 instances across multiple Availability Zones.

Getting Started with AWS Web Application Hosting for Microsoft Windows Step 14: Clean Up



AWS CloudFormation creates a new endpoint for your Amazon RDS database. You'll need to update the Amazon RDS endpoint on each instance in your default.aspx.cs file just as you did in [Modify the Application \(p. 37\)](#). You can connect to your Amazon EC2 instances using the new Administrator password that you created when you first connected to your instance. You'll also notice that you don't see the images that you stored in your original Amazon RDS database because you are now pointing to a new database. To restore your original images you can create a backup of your database and then restore it from snapshot. For more information about backing up your Amazon RDS database, see [Backing Up and Restoring DB Instances](#) in the *Amazon Relational Database Service User Guide*. Now let's clean up all of your AWS resources. Deleting your AWS CloudFormation stack will delete all of the AWS resources it just created.

To delete an AWS CloudFormation stack

1. At the top of the AWS Management Console, click the AWS CloudFormation tab.
2. Click the stack you want to delete, and click **Delete Stack**.

Step 14: Clean Up

Topics

- [Delete Your CloudWatch Alarm \(p. 51\)](#)
- [Delete Your Elastic Load Balancer \(p. 51\)](#)
- [Terminate Your Amazon EC2 Instances in Your Auto Scaling Group \(p. 52\)](#)
- [Terminate Your DB Instance \(p. 54\)](#)
- [Delete a Key Pair \(p. 54\)](#)
- [Delete an Amazon EC2 Security Group \(p. 55\)](#)
- [Delete an Amazon CloudFront Distribution \(p. 55\)](#)
- [Delete Objects and an Amazon S3 Bucket \(p. 55\)](#)

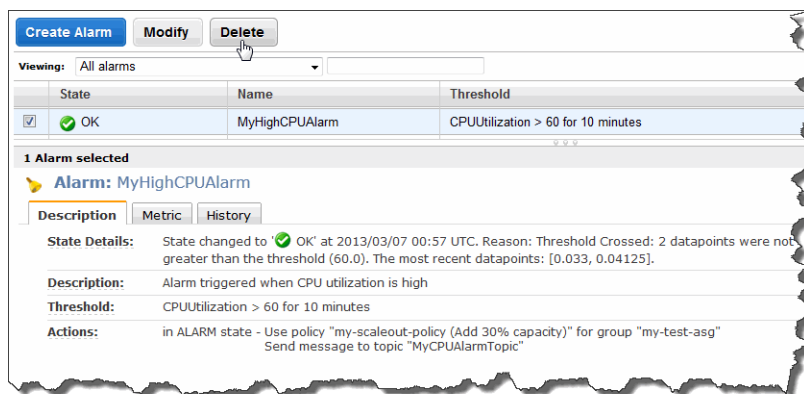
Congratulations! You have just deployed your web application. Now, to prevent accruing any further charges, let's terminate our environments and clean our resources.

Delete Your CloudWatch Alarm

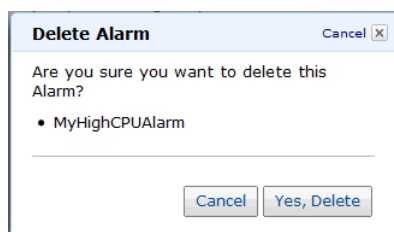
After you've decided that you no longer need the alarm, you can delete it.

To delete your alarm

1. Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the top navigation bar, click **US East (N. Virginia)** in the region selector.
3. In the left navigation pane, click **Alarms**.
4. Select the check box next to the alarm that you want to delete, and then click **Delete**.



5. When a confirmation message appears, click **Yes, Delete**.



Delete Your Elastic Load Balancer

As soon as your load balancer becomes available, AWS bills you for each hour or partial hour that you keep the load balancer running. After you've decided that you no longer need the load balancer, you can delete it.

To delete your load balancer

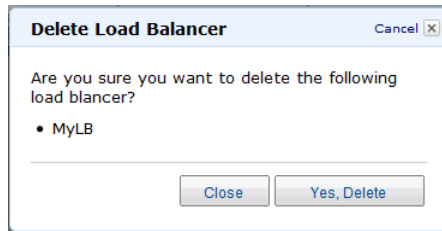
1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the top navigation bar, click **US East (N. Virginia)** in the region selector.
3. In the left navigation pane, click **Load Balancers**.
4. Select the check box next to the load balancer you want to delete and then click **Delete**.



Getting Started with AWS Web Application Hosting for Microsoft Windows

Terminate Your Amazon EC2 Instances in Your Auto Scaling Group

5. When a confirmation message appears, click **Yes, Delete**.



Elastic Load Balancing deletes the load balancer. As soon as the load balancer is deleted, you stop incurring charges for that load balancer.

Caution

Even after you delete a load balancer, the Amazon EC2 instances associated with the load balancer continue to run. You will continue to incur charges on the Amazon EC2 instances while they are running.

Terminate Your Amazon EC2 Instances in Your Auto Scaling Group

In this section you will first remove the Amazon EC2 instance, then delete the Auto Scaling group, and finally delete the launch configuration.

You must terminate all Amazon EC2 instances in an Auto Scaling group before you can delete the group. A simple way to terminate all instances in a group is to update the group so that both the minimum size and maximum size are set to zero.

To remove the Amazon EC2 instance from the Auto Scaling group

1. Open a command prompt window: From a Windows computer, click **Start**. In the Search box, type `cmd`, and then press **Enter**.
2. You'll use the `as-update-auto-scaling-group` command to update the Auto Scaling group that we created earlier. At the command prompt, type the following, and then press **Enter**:

```
PROMPT>as-update-auto-scaling-group MyAutoScalingGroup --min-size 0 --max-size 0
```

Auto Scaling returns the following:

```
OK-Updated AutoScalingGroup
```

3. Now you'll use the `as-describe-auto-scaling-groups` command to verify that Auto Scaling has removed the instance from `MyAutoScalingGroup`.

It can take a few minutes for the instance to terminate, so you might have to check the status more than once. At the command prompt, type the following, and then press **Enter**:

```
PROMPT>as-describe-auto-scaling-groups MyAutoScalingGroup --headers
```

If the instance termination is still in progress, Auto Scaling returns information similar to the following. (Your value for `INSTANCE-ID` will differ):

Getting Started with AWS Web Application Hosting for Microsoft Windows

Terminate Your Amazon EC2 Instances in Your Auto Scaling Group

AUTO-SCALING-GROUP	GROUP-NAME	LAUNCH-CONFIG	AVAILABILITY-ZONES		
LOAD-BALANCERS	MIN-SIZE	MAX-SIZE	DESIRED-CAPACITY		
AUTO-SCALING-GROUP	MyAutoScalingGroup	MyLC	us-east-1b,us-east-1c		
MyLB	0	0	0		
INSTANCE	INSTANCE-ID	AVAILABILITY-ZONE	STATE	STATUS	LAUNCH-CONFIG
INSTANCE	i-xxxxxxx	us-east-1c	InService	Healthy	MyLC

Note

You can also click **Instances** in the Amazon EC2 console to view the status of your instances.

When no instances exist in MyAutoScalingGroup, you can delete the group.

To delete the Auto Scaling group

- At the command prompt, type the following, and then press **Enter**:

```
PROMPT>as-delete-auto-scaling-group MyAutoScalingGroup
```

To confirm the deletion, type **y**, and then press **Enter**.

```
Are you sure you want to delete this MyAutoScalingGroup? [Ny]
```

Auto Scaling returns the following:

```
OK-Deleted MyAutoScalingGroup
```

All that remains now is to delete the launch configuration you created for this Auto Scaling group.

To delete the launch configuration

- At the command prompt, type the following, and then press **Enter**:

```
PROMPT>as-delete-launch-config MyLC
```

To confirm the deletion, type **y** and then press **Enter**.

```
Are you sure you want to delete this launch configuration? [Ny]
```

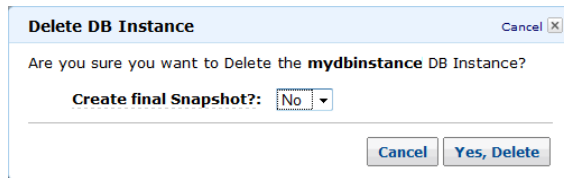
Auto Scaling returns the following:

```
OK-Deleted launch configuration
```


Terminate Your DB Instance

To terminate your DB Instance

1. Open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. Make sure **US East (N. Virginia)** is selected in the region selector in the navigation bar.
3. In the left navigation pane, click **DB Instances**.
4. Locate the DB Instance in your list of DB Instances.
5. Select the check box next to the DB Instance, and then choose **Delete** from the **Instance Actions** drop-down list at the top of the page.
6. Select **No** in the **Create final snapshot?** drop-down list.



The screenshot shows a modal dialog titled "Delete DB Instance" with a "Cancel" button in the top right corner. The main text asks, "Are you sure you want to Delete the mydbinstance DB Instance?". Below this is a label "Create final Snapshot:" followed by a dropdown menu currently showing "No". At the bottom of the dialog are two buttons: "Cancel" and "Yes, Delete".

If this weren't an exercise, you might create a final snapshot before you deleted the DB Instance so that you could restore the DB Instance later.

Note

Creating a final snapshot incurs additional storage fees.

7. Click **Yes, Delete**.

Amazon RDS begins terminating the instance. As soon as the DB Instance status changes to **deleted**, you stop incurring charges for that DB Instance.

To delete your DB Security Group

1. Open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. Click **DB Security Groups** in the navigation pane on the left.
3. Select a DB Security Group and click **Delete DB Security Group**.
4. Click **Yes, Delete**.

Delete a Key Pair

This is an optional step. You are not charged for keeping a key pair, and you may want to reuse the key pair for later use.

To delete a key pair

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the top navigation bar, click **US East (N. Virginia)** in the region selector.
3. In the left navigation pane, click **Key Pairs**.
4. Select the check box beside the key pair you want to delete, and then click **Delete**.
5. When a confirmation message appears, click **Yes**.

Delete an Amazon EC2 Security Group

To delete a security group

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the top navigation bar, click **US East (N. Virginia)** in the region selector.
3. In the left navigation pane, click **Security Groups**.
4. In the details pane, under **Security Groups**, select a security group you want to delete, and then click **Delete**.
5. Click **Yes, Delete**.

Delete an Amazon CloudFront Distribution

To disable a CloudFront distribution

1. Open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Select a CloudFront distribution and click **Disable**.

A confirmation dialog box appears.

3. Click **Yes, Disable**.

Amazon CloudFront disables the distribution.

To delete a CloudFront distribution

1. Select a CloudFront distribution and click **Delete**.

A confirmation dialog box appears.

2. Click **Yes, Delete**.

Amazon CloudFront deletes the distribution.

Delete Objects and an Amazon S3 Bucket

Before you can delete an Amazon S3 bucket, all objects within the bucket must be deleted.

You should also ensure that logging for your Amazon S3 bucket is disabled; otherwise, logs might be immediately written to your bucket after you delete your bucket's objects.

To disable logging

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** pane, right-click your bucket and then click **Properties**.
3. In the **Properties** pane, click **Logging**.
4. Clear the **Enabled** check box.

To delete an object

1. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** pane, click the bucket where the objects are stored.
3. In the list of objects, right-click the object that you want to delete and then click **Delete**.

A dialog box shows the actions you can take on the selected object(s).

Tip

You can use the **SHIFT** and **CRTL** keys to select multiple objects and perform the same action on them simultaneously.

4. In the confirmation message that appears, click **Yes, Delete**.

To delete a bucket, you must first delete all of the objects in it.

To delete a bucket

1. Continuing from the previous procedure, right-click the bucket you want to delete, and then click **Delete**.
2. In the confirmation message that appears, click **Yes, Delete**.

Amazon Route 53

Amazon Route 53 is a scalable Domain Name System (DNS) web service. It provides secure and reliable routing to your infrastructure that uses Amazon Web Services (AWS) products, such as Amazon Elastic Compute Cloud (Amazon EC2), Elastic Load Balancing, or Amazon Simple Storage Service (Amazon S3). You can also use Amazon Route 53 to route users to your infrastructure outside of AWS.

Amazon Route 53 automatically routes queries to the nearest DNS server in a global network of DNS servers, resulting in low latency. It is an authoritative DNS service, meaning it translates friendly domains names like `www.example.com` into IP addresses like `192.0.2.1`.

You can manage your DNS records through the Amazon Route 53 API, or set account-level user and access management through the Identity and Access Management (IAM) API. Like other AWS products, there are no contracts or minimum commitments for using Amazon Route 53—you pay only for the domains you configure and the number of queries that the service answers. For more information about Amazon Route 53 pricing, see [Amazon Route 53 Pricing](#).

The following procedure explains the high-level steps you need to take to use Route 53 for this example. For instructions on how to do steps one through four, go to [Amazon Route 53 Getting Started Guide](#). For information on how to create an alias in the last step, go to [How to Create an Alias Record Set](#) in the *Amazon Route 53 Developer Guide*.

To use Amazon Route 53

1. Create a hosted zone for `example.com`.
2. Create a new DNS record for your static content (e.g., `static.example.com`) that points to your CloudFront distribution (e.g., `d18k4jybr69gw2.cloudfront.net`).
3. Create a new DNS record for your website (e.g., `www.example.com`) that points to your Elastic Load Balancer CNAME.
4. Confirm your requests are complete.
5. Update the registrar's name server records.
6. Create an alias for your load balancer which responds to queries for `example.com` and `www.example.com`. You use the hosted zone ID for the load balancer (e.g., `Z3DZXEQ79N41H`).

Pricing

Topics

- [Amazon S3 Cost Breakdown \(p. 58\)](#)
- [Amazon CloudFront Cost Breakdown \(p. 60\)](#)
- [Amazon EC2 Cost Breakdown \(p. 62\)](#)
- [Amazon RDS Cost Breakdown \(p. 65\)](#)
- [Summing It All Up \(p. 67\)](#)

The [AWS Simple Monthly Calculator](#) estimates your monthly bill. It provides a per-service cost breakdown, as well as an aggregate monthly estimate. You can also use the calculator to see an estimate and breakdown of costs for common solutions. This topic walks you through an example of using the AWS Simple Monthly Calculator to estimate your monthly bill.

Note

AWS pricing you see in this documentation is current at the time of publication. For the latest pricing information, go to [AWS Service Pricing Overview](#). For more information on how AWS pricing works, go to [How AWS Pricing Works](#).

Amazon S3 Cost Breakdown

The following table shows the characteristics for Amazon S3 we have identified for this web application hosting architecture.

Characteristic	Metric	Description
Storage	0.001 GB/month	20 JPEG @ 50 KB = 1 MB Total of 1.0 MB for 20 objects = 0.001 GB

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Amazon S3 Cost Breakdown**

Characteristic	Metric	Description
Requests	PUT requests : 20/month GET requests: 2000/month	<p>We will plan to update the objects twice a month. (PUT requests include from any location including Amazon EC2) 1 PUT request x 20 objects = 20 requests</p> <p>(PUT requests include from any location including Amazon EC2)</p> <p>We will transfer the objects 10 times per month to each of the 10 Amazon CloudFront edge locations. (GET requests include from any location including Amazon EC2) 10 GET requests x 10 CloudFront Nodes x 20 objects = 2000 requests</p> <p>(GET requests include from any location including Amazon EC2)</p>
Data Transfer	Data out: 0.1 GB/month	If the average size object is 50KB, and we make about 2000 requests per month, than the average data transfer is approximately 0.1 GB.

The following image shows the cost breakdown for Amazon S3 in the AWS Simple Monthly Calculator.

Amazon S3 is storage for the Internet. It is designed to make web-scale computing easier for developers. [Clear Form](#)

Storage:

Storage: GB

Reduced Redundancy Storage: GB

Requests:

PUT/COPY/POST/LIST Requests: Requests

GET and Other Requests: Requests

Data Transfer:

Inter-Region Data Transfer Out: GB/Month

Data Transfer Out: GB/Month

Data Transfer In: GB/Month

The total monthly cost is the sum of the cost for storing the objects, updating the objects, and transferring the objects to Amazon CloudFront.

Variable	Formula	Calculation
Provisioned Storage	Storage rate	\$0.125
	X Storage Amount (GB)	x 0.001
	-----	-----
		\$0.00

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Amazon CloudFront Cost Breakdown**

Variable	Formula	Calculation
PUT Requests	Request Rate	\$0.01
	X Number of requests (per 1000)	x 1
	-----	----
		\$0.01
GET Requests	Request Rate	\$0.01
	X Number of requests (per 10000)	x 1
	-----	----
		\$0.01

We use the AWS Simple Monthly calculator to estimate this. With the calculator, the total cost for Amazon S3 is \$0.02.

Amazon CloudFront Cost Breakdown

The following table shows the characteristics for Amazon CloudFront we have identified for this web application hosting architecture.

Characteristic	Metric	Description
Traffic Distribution	50% US	Distribution of traffic across regions
	25% EU	
	10% HK	
	15% JP	
Request Type	HTTP	Type of requests that customers make to the cached locations
Data Transfer Out	30.0 GB/month	1.0 MB x 30.5 days x 1,000 hits/day

The following image shows the cost breakdown for Amazon CloudFront in the AWS Simple Monthly Calculator.

Getting Started with AWS Web Application Hosting for Microsoft Windows Amazon CloudFront Cost Breakdown

Choose region: US-East / US Standard (Northern Virginia) Inbound Data Transfer is Free and Outbound Data Transfer is 1 GB free per region per month

Amazon CloudFront is a web service for content delivery. It delivers your content using a global network of edge locations and works seamlessly with Amazon S3 which durably stores the original, definitive versions of your files. Clear Form

Data Transfer Out:
Monthly Volume: 30 GB/Month

Requests:
Average Object Size: 50 KB
Type of Requests: ☒ HTTP ☐ HTTPS
Invalidation Requests: 0 Requests

Edge Location Traffic Distribution:

United States	50 %
Europe	25 %
Hong Kong and Singapore	10 %
Japan	15 %
South America	0 %
Australia	0 %

The total monthly cost is the sum of the data transfer out plus the requests costs for each of the regions.

Variable	Formula	Calculation
Data Transfer Out for US	Monthly Volume (GB)	30.00
	Traffic Distribution (%)	0.50
	x Data Out Rate	x 0.12
	-----	-----
		\$1.80
Data Transfer Out for EU	Monthly Volume (GB)	30.00
	Traffic Distribution (%)	0.25
	x Data Out Rate	x 0.12
	-----	-----
		\$0.90
Data Transfer Out for HK/Singapore	Monthly Volume (GB)	30.00
	Traffic Distribution (%)	0.10
	x Data Out Rate	x 0.19
	-----	-----
		\$0.57

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Amazon EC2 Cost Breakdown**

Variable	Formula	Calculation
Data Transfer Out for JP	Monthly Volume (GB) Traffic Distribution (%) x Data Out Rate -----	30.00 0.15 x 0.201 ----- \$0.90
Requests for US	Request Rate Traffic Distribution (%) x (Monthly Volume/Object Size (per 10,000 requests)) -----	\$0.0075 0.50 x (30.00GB/50KB/10K) ----- \$0.24
Requests for EU	Request Rate Traffic Distribution (%) x (Monthly Volume/Object Size (per 10,000 requests)) -----	\$0.009 0.25 x (30.00GB/50KB/10K) ----- \$0.15
Requests for HK/Singapore	Request Rate Traffic Distribution (%) x (Monthly Volume/Object Size (per 10,000 requests)) -----	\$0.0075 0.10 x (30.00GB/50KB/10K) ----- \$0.05
Requests for JP	Request Rate Traffic Distribution (%) x (Monthly Volume/Object Size (per 10,000 requests)) -----	\$0.0095 0.15 x (30.00GB/50KB/10K) ----- \$0.09

We use the AWS Simple Monthly calculator to obtain this estimate. According to the calculator, the total cost for CloudFront is \$4.70.

Amazon EC2 Cost Breakdown

The following table shows the characteristics for Amazon EC2 we have identified for this web application hosting architecture.

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Amazon EC2 Cost Breakdown**

Characteristic	Metric	Description
Uptime	24 hrs/day	Assuming there is an average of 30.5 days in a month, the instance runs 732 hours/month
Machine Characteristics	t1.micro instance	Micro Instance 613 MB of memory, up to 2 ECUs (for short periodic bursts), EBS storage only, 32-bit or 64-bit platform For a list of instance types, go to http://aws.amazon.com/ec2/instance-types/ .
Additional Storage	1 EBS Volume Storage: 30 GB/Month 100 IOPS	The AMI is EBS-backed, the volume will have 30 GB provisioned storage, and 100 I/O requests made to the volume per second.
Data Transfer	Data In: 0.005 GB/day Data Out: 0.05 GB/day	There are approximately 1,000 hits per day and each response is about 50 KB and each request is about 5 KB.
Instance Scale	2	On average in a given day, there are 2 instances running.
Elastic Load Balancing	Hourly usage: 732 hrs/month Data processed: 1.525 GB/month	ELB is used 24 hrs/day, 7 days/week ELB processes a total of 0.055 GB/day (data in + data out)
Detailed Monitoring	\$3.50 per instance per month	We have setup detailed monitoring for our Amazon EC2 instances.

The following image shows the cost breakdown for Amazon EC2 in the AWS Simple Monthly Calculator.

Getting Started with AWS Web Application Hosting for Microsoft Windows Amazon EC2 Cost Breakdown

Choose region: US-East / US Standard (Northern Virginia) Inbound Data Transfer is Free and Outbound Data Transfer is 1 GB free per region per month

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. Amazon Elastic Block Store (EBS) provides persistent storage to Amazon EC2 instances. [Clear Form](#)

+ Compute: Amazon EC2 On-Demand Instances:

Description	Instances	Usage	Instance Type	Operating System	Tenancy	Detailed Monitoring
	2	24 Hours/Day	Micro	Windows	Default	<input checked="" type="checkbox"/>
<input type="checkbox"/> EBS-Optimized						

+ Compute: Amazon EC2 Reserved Instances:

Description	Instances	Usage	Instance Type	Operating System	Offering and Term	Tenancy	Detailed Monitoring
	0	0 Hours/Month	Small	Windows	Medium Utilization 3yr term	Default	<input type="checkbox"/>
<input type="checkbox"/> EBS-Optimized							

+ Storage: Amazon EBS Volumes:

Description	Volumes	Volume Type	Storage	IOPS	Snapshot Storage
	1	Standard	30 GB	100	0 GB-month of Storage

Elastic IP:

Number of Additional Elastic IPs:

Elastic IP Non-attached Time: Hours/Month

Number of Elastic IP Remaps: Per Month

Data Transfer:

Inter-Region Data Transfer Out: GB/Month

Data Transfer Out: GB/Day

Data Transfer In: GB/Day

Intra-Region Data Transfer: GB/Month

Public IP/Elastic IP Data Transfer: GB/Month

Elastic Load Balancing:

Number of Elastic LBs:

Total Data Processed by all ELBs: GB/Day

The total monthly cost is the sum of the cost of the running instances, EBS volumes and I/O requests, elastic load balancers, the data processed by the elastic load balancers, and Amazon CloudWatch metrics.

Variable	Formula	Calculation
Instance Cost	Instance cost per hour Number of instances x Uptime in hours -----	\$0.02 2 x 732 ----- \$29.28
Additional Storage	Storage rate x Storage Amount (GB) + (I/O requests rate x seconds per month x Request rate(per 1M requests)) -----	\$0.10 x 30 + (100 x ~2.6M x \$0.10)/1M ----- \$29.35
Elastic Load Balancing	Hours used x Hourly rate + (Data processed (GB) x Process rate) -----	732 x \$0.025 + 1.6775 x \$0.008 ----- \$18.31

**Getting Started with AWS Web Application Hosting for
Microsoft Windows
Amazon RDS Cost Breakdown**

Variable	Formula	Calculation
Amazon CloudWatch	Number of instances x Detailed Monitoring Rate ----- -----	2 x \$3.50 ----- \$7.00

We use the AWS Simple Monthly calculator to estimate this. With the calculator, the total cost for Amazon EC2 is \$83.94.

Amazon RDS Cost Breakdown

The following table shows the characteristics for Amazon RDS we have identified for this web application hosting architecture.

Characteristic	Metric	Description
Uptime	24 hrs/day	24 Assuming there is an average of 30.5 days in a month, the instance runs 732 hours/month
Database Characteristics	Small Amazon RDS instance	1.7 GB memory, 1 ECU (1 virtual core with 1 ECU), 64-bit platform, Moderate I/O Capacity
Provisioned Storage	5 GB/month	Amazon provides 5 GB to 1 TB of associated storage capacity for your primary data set.
Requests	2M IOP/month	We have 1,000 hits per day at a rate of 5 IOP per hit on site. Assume there are 30.5 days in a month on average. This is a total of 152,500 I/O requests per month or 1M (rounded to the nearest million), but since the write I/O request will double since data is also replicated to the standby instance, we have a total of 2M.
Deployment Type	Multi-AZ	We will run our database instance across multiple Availability Zones.
Additional Backup Storage	none	We'll use up to the provisioned amount which is 5 GB.
Data Transfer	Data in: 0 GB Data out: 0 GB	There is no data transfer from RDS to the Internet.
Database Instance Scale	1	We need one database instance.

The following image shows the cost breakdown for Amazon RDS in the AWS Simple Monthly Calculator.

Getting Started with AWS Web Application Hosting for Microsoft Windows Amazon RDS Cost Breakdown

Choose region: US-East / US Standard (Northern Virginia) Inbound Data Transfer is Free and Outbound Data Transfer is 1 GB free per region per month

Amazon RDS is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. [Clear Form](#)

+ Amazon RDS On-demand DB Instances:

Description	DB Instances	Usage	DB Engine and License	Class and Deployment	Storage	IOPS
	1	24 Hours/Day	MySQL	Small Multi-AZ	Provisioned 5 GB	01

+ Additional Backup Storage (Free backup storage up to 100% of provisioned Storage):

Backup Storage: 0 GB-month of Storage

+ Amazon RDS Reserved DB Instances:

Description	DB Instances	Usage	DB Engine and License	Class and Deployment	Offering and Term	Storage	IOPS
	0	0 Hours/Month	MySQL	Small Standard (Single-AZ) o	Medium Utilization 3 yr term	Standard 20 GB	0

Data Transfer:

Data Transfer Out: 0 GB/Month

Data Transfer In: 0 GB/Month

Intra-Region Data Transfer: 0 GB/Month

Because we do not have data transfer in or out or backup storage, the total monthly cost is the sum of the cost of the running instances, provisioned storage, and I/O requests.

Variable	Formula	Calculation
Instance Cost	Instance cost per hour	\$0.153
	Number of instances	1
	x Uptime in hours	x 732
	-----	-----
		\$112.00
Provisioned Storage	Storage rate	\$0.20
	x Storage Amount (GB)	x 5
	-----	-----
		\$1.00
I/O Requests	I/O rate	\$0.10
	x Number of requests (millions)	x ~2
	-----	-----
		\$0.22

We use the AWS Simple Monthly calculator to estimate this. With the calculator, the total cost for Amazon RDS is \$113.52.

Summing It All Up

To calculate the total cost for this scenario, we add the cost for Amazon S3, Amazon CloudFront, Amazon EC2, Amazon RDS, and the AWS Data Transfer Out, and subtract any discount that falls into the AWS Free Usage Tier.

The total AWS Transfer Out is an aggregate Data Transfer Out usage across Amazon EC2, Amazon S3, and Amazon RDS. For Amazon EC2, we have 0.05 GB per day which is approximately 1.525 GB per month. For Amazon S3, we have 0.1 GB per month, and for Amazon RDS we did not have any AWS Transferred out. Since up to 1 GB per month of data transferred out is free, we are left with a total of 0.625 GB per month.

Variable	Formula	Calculation
AWS Data Transfer	(Data in (GB) X Data In Rate)	0.1525 X \$0.00
	+ (Data out (GB) X Data Out Rate)	+ (0.625) X \$0.12
	-----	-----
		\$0.08


The following image shows an example of your monthly estimate.

Services

Estimate of your Monthly Bill (\$ 165.37)

Estimate of Your Monthly Bill

☒ Show First Month's Bill (include all one-time fees, if any)

 With AWS, You only pay for what you use. Below you will see an estimate of your monthly bill. Expand each line item to see cost breakout of each service. To save this bill and input values, click on 'Save and Share' button. To remove the service from the estimate, click on the red cross.

Save and Share

+ Amazon EC2 Service (US-East)	\$	83.94
+ Amazon S3 Service (US-East)	\$	0.00
+ Amazon RDS Service (US-East)	\$	113.25
+ Amazon CloudFront Service	\$	4.70
+ AWS Data Transfer In	\$	0.00
+ AWS Data Transfer Out	\$	0.08
+ AWS Support (Basic)	\$	0.00
Free Tier Discount:	\$	-36.59
Total One-Time Payment:	\$	0.00
Total Monthly Payment:	\$	165.37

The total cost of this web application is estimated at \$165.37 per month including the free usage tier discount.

Related Resources

The following table lists related resources that you'll find useful as you work with AWS services.

Resource	Description
AWS Products and Services	A comprehensive list of products and services AWS offers.
Documentation	Official documentation for each AWS product including service introductions, service features, and API references, and other useful information.
AWS Architecture Center	Provides the necessary guidance and best practices to build highly scalable and reliable applications in the AWS cloud. These resources help you understand the AWS platform, its services and features. They also provide architectural guidance for design and implementation of systems that run on the AWS infrastructure.
AWS Economics Center	Provides access to information, tools, and resources to compare the costs of Amazon Web Services with IT infrastructure alternatives.
AWS Cloud Computing Whitepapers	Features a comprehensive list of technical AWS whitepapers covering topics such as architecture, security, and economics. These whitepapers have been authored either by the Amazon team or by AWS customers or solution providers.
Videos and Webinars	Previously recorded webinars and videos about products, architecture, security, and more.
Discussion Forums	A community-based forum for developers to discuss technical questions related to Amazon Web Services.
AWS Support Center	The home page for AWS Technical Support, including access to our Developer Forums, Technical FAQs, Service Status page, and AWS Premium Support. (subscription required).
AWS Premium Support Information	The primary web page for information about AWS Premium Support, a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.

Resource	Description
Form for questions related to your AWS account: Contact Us	This form is <i>only</i> for account questions. For technical questions, use the Discussion Forums.
Conditions of Use	Detailed information about the copyright and trademark usage at Amazon.com and other topics.

Document History

This document history is associated with the 2011-09-30 release of Getting Started. This guide was last updated on May 18, 2012.

Change	Description	Release Date
New content	Created new document	30 September 2011
Added new section	Added new section to talk about AWS Identity and Account Management	24 October 2011
Added new Windows AMI	Added new Windows AMI for Auto Scaling	26 October 2011
Changed Windows AMI	Changed Windows AMI to a t1.micro for Auto Scaling and Pricing example	09 November 2011
Changed Windows AMI	Changed Windows AMI to a newer Windows AMI	25 November 2011
Added new section	Updated content for creating a CloudFront distribution.	18 May 2012