Overview

Amazon Web Services (AWS) is a great platform for running Windows workloads, and offers several options for deploying and managing both off-the-shelf and custom applications. When planning to deploy a .NET web application to AWS for the first time, new customers are sometimes unclear as to which method will allow them to quickly get their application up and running. This document provides guidance for deploying .NET web applications quickly and easily using either AWS-provided or self-managed IIS web servers.

The following sections assume basic knowledge of Amazon Elastic Compute Cloud (Amazon EC2), Microsoft Windows, .NET application development, load balancing, and Microsoft SQL server databases.

General Best Practices

The size and installation complexity of web applications can vary greatly, therefore there is rarely a one-size-fits-all solution for deploying and hosting .NET applications. However, there are some universal best practices to consider when deploying any web application:

- Understand the deployment, installation, and configuration characteristics of the application.
- Understand application expectations from initial deployment to future scalability, availability, and backup and recovery requirements.
- Use automation whenever possible for deployment and other tasks where consistency is important.
- Leverage source code or application repositories to protect your application.

Application on the AWS Platform

AWS offers several tools and services to enable both AWS-managed and customer-managed .NET application deployment. The table below is a high-level reference to help identify the most appropriate option for a specific scenario. The following sections describe these different approaches and their applicable use cases in more detail.

<table>
<thead>
<tr>
<th>Application Characteristics</th>
<th>Packaging Tools</th>
<th>Deployment Mechanism</th>
<th>Deployment Method/ Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom .NET applications developed in Visual Studio</td>
<td>Visual Studio</td>
<td>Single-click deployment from within Visual Studio</td>
<td>AWS Toolkit for Visual Studio</td>
</tr>
<tr>
<td>Stand-alone .NET web applications deployed as an IIS Default Website, and requiring minimal OS changes</td>
<td>Visual Studio, Microsoft Build Engine (MSBuild), or IIS Web Deploy</td>
<td>Automated deployment of packaged application using AWS Elastic Beanstalk</td>
<td>AWS Elastic Beanstalk</td>
</tr>
<tr>
<td>Any .NET application or server configuration, especially those needing customized OS or third-party installers</td>
<td>Existing MSI or customer installers, application archive (ZIP), manual file copy, etc.</td>
<td>Existing software deployment tools and processes or automated deployment services, such as AWS CodeDeploy or AWS OpsWorks.</td>
<td>EC2 Windows Instances</td>
</tr>
</tbody>
</table>
AWS Toolkit for Visual Studio

The AWS Toolkit for Visual Studio provides the easiest and most straightforward way to deploy custom .NET applications to AWS. This approach allows developers to deploy .NET code directly to AWS, and to create supporting resources, such as AWS-managed RDS SQL Server databases, directly from within Microsoft Visual Studio. For production deployments, customers can use the Visual Studio Publish Web wizard to package .NET applications that are then versioned and deployed through more controlled change management processes using the AWS Elastic Beanstalk (Elastic Beanstalk) console, Elastic Beanstalk Command Line Interface (EB CLI), or Elastic Beanstalk API calls.

AWS Elastic Beanstalk

Elastic Beanstalk is an easy-to-use service for deploying and scaling .NET web applications. Elastic Beanstalk automatically manages capacity provisioning, load balancing, and Auto Scaling. This approach is appropriate for companies deploying .NET applications that include the following criteria:

- Are standalone web applications
- Are deployed as the IIS Default Website
- Require minimal OS changes

Additionally, the web application must be packaged for Elastic Beanstalk deployment. Elastic Beanstalk supports the following packaging and deployment mechanisms:

- Custom applications developed and deployed directly to Elastic Beanstalk using Microsoft Visual Studio and the AWS Toolkit for Visual Studio
- Custom applications developed and packaged using the Visual Studio Publish Web wizard and deployed with the Elastic Beanstalk console, EB CLI, or Elastic Beanstalk API calls.
- .NET projects packaged with MSBuild and deployed with the Elastic Beanstalk console, EB CLI, or Elastic Beanstalk API calls.
- Website projects packaged with IIS Web Deploy tool and deployed with the Elastic Beanstalk console, EB CLI, or Elastic Beanstalk API calls.

EC2 Windows Instances

Amazon EC2 Windows instances allow customers to deploy .NET applications to AWS using their existing application deployment tools and processes, or to integrate .NET application deployment with automated deployment tools and services such as AWS CodeDeploy or AWS OpsWorks. AWS CodeDeploy and AWS OpsWorks make it easier to rapidly release new features, and automate the configuration, deployment, management, and updates of your services and applications on both EC2 instances and on-premises servers.

Amazon EC2 offers preconfigured Amazon Machine Images (AMIs) that enable customers to launch fully supported Windows Server virtual machines in minutes. Customers can select from different Windows OS versions and have the option to include SQL Server. This option is best suited for companies with the following requirements:

---

1 Note that Elastic Beanstalk configuration files support advanced IIS environment and OS configuration options. However this requires additional Elastic Beanstalk packaging effort and expertise.

• Want to leverage their existing Microsoft Windows expertise and standards
• Need to integrate their AWS environment into existing on-premises development and deployment processes
• Require the most flexibility in OS version and configuration
• Plan to deploy third-party .NET applications with sophisticated installers, manual license configuration, or extensive external dependencies

This approach offers tremendous flexibility, but also requires advanced AWS knowledge and service configuration, such as creating and configuring Elastic Load Balancing load balancers, Auto Scaling groups, and self-managing Windows servers as well as implementing custom application-deployment processes.

Resources

- **Web Application Hosting in the AWS Cloud: Best Practices**
  AWS whitepaper on how to create a highly available, scalable web application

- **Overview of Deployment Options**
  [http://d0.awsstatic.com/whitepapers/overview-of-deployment-options-on-aws.pdf](http://d0.awsstatic.com/whitepapers/overview-of-deployment-options-on-aws.pdf)
  AWS whitepaper on deployment services, features, and strategies

- **AWS Elastic Beanstalk Documentation**
  AWS webpage with links to Elastic Beanstalk technical documentation including the *Developer Guide* and detailed information about getting started with web applications.

- **AWS Toolkit for Visual Studio**

- **AWS CodeDeploy**

- **AWS OpsWorks**
  [https://aws.amazon.com/opsworks/](https://aws.amazon.com/opsworks/)

- **AWS and Microsoft**
  AWS webpage “on how Windows runs in the AWS cloud”