Sitecore on the AWS Cloud

Quick Start Reference Deployment

Brian Wagner

November 2015

This guide is also available in HTML format at
http://docs.aws.amazon.com/quickstart/latest/sitecore/.

Contents

About This Guide

This Quick Start reference deployment guide discusses architectural considerations and configuration steps for deploying the Sitecore content management system on the Amazon Web Services (AWS) cloud. It also provides links for viewing and launching AWS CloudFormation templates that automate the deployment.

The guide is for IT infrastructure architects, administrators, and DevOps professionals who are planning to implement or extend their Sitecore workloads on the AWS cloud.

Quick Starts are automated reference deployments for key workloads on the AWS cloud. Each Quick Start launches, configures, and runs the AWS compute, network, storage, and other services required to deploy a specific workload on AWS, using AWS best practices for security and availability.

Overview

Sitecore on AWS

Sitecore is a popular enterprise content management system and multichannel marketing automation software. The software architecture of the product is well suited to take advantage of the benefits of the AWS cloud.

Sitecore can be deployed on AWS as a highly available and fault-tolerant application. By using AWS services, including Amazon Relational Database Service (Amazon RDS) and Elastic Load Balancing, you can operate Sitecore in a flexible and scalable way.

This guide adds AWS context to the Sitecore Scaling Guide by providing infrastructure and corresponding software configurations. It follows the recommendations in the scaling guide with a few deviations, which will be highlighted.

Quick Links

The links in this section are for your convenience. Before you launch the Quick Start, please review the architecture, configuration, network security, and other considerations discussed in this guide.

View template

It is also in your best interest to review the Sitecore Scaling Guide before you deploy this Quick Start to become familiar with the concepts of a distributed Sitecore deployment.

Launch Quick Start

This template will deploy the application with two content management servers behind a load balancer, spread across two Availability Zones in the region that you choose.

The template includes default settings that you can customize by following the instructions in this guide.

Time to deploy: Less than 30 minutes

Cost and Licenses

You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using the Quick Start. As of the date of publication, the cost for using the Quick Start with default settings is approximately $2 an hour. Prices are subject to change. See the pricing pages for each AWS service you will be using in this Quick Start for full details.

The version of Sitecore that you will be deploying is version 7.2. You are responsible for providing your own license. You will be required to define the location of the license file in your template before it is deployed. Please make sure that the file is accessible to the resources being launched, which will need to download it from a remote location.

This Quick Start launches the Amazon Machine Image (AMI) for Microsoft Windows Server 2012 R2 and includes the license for the Windows Server 2012 R2 operating system. The AMI is updated on a regular basis with the latest service pack for the operating system, so you don't have to install any updates.

Architecture

Deploying this Quick Start with the **default parameters** builds the following Sitecore environment in the AWS cloud.

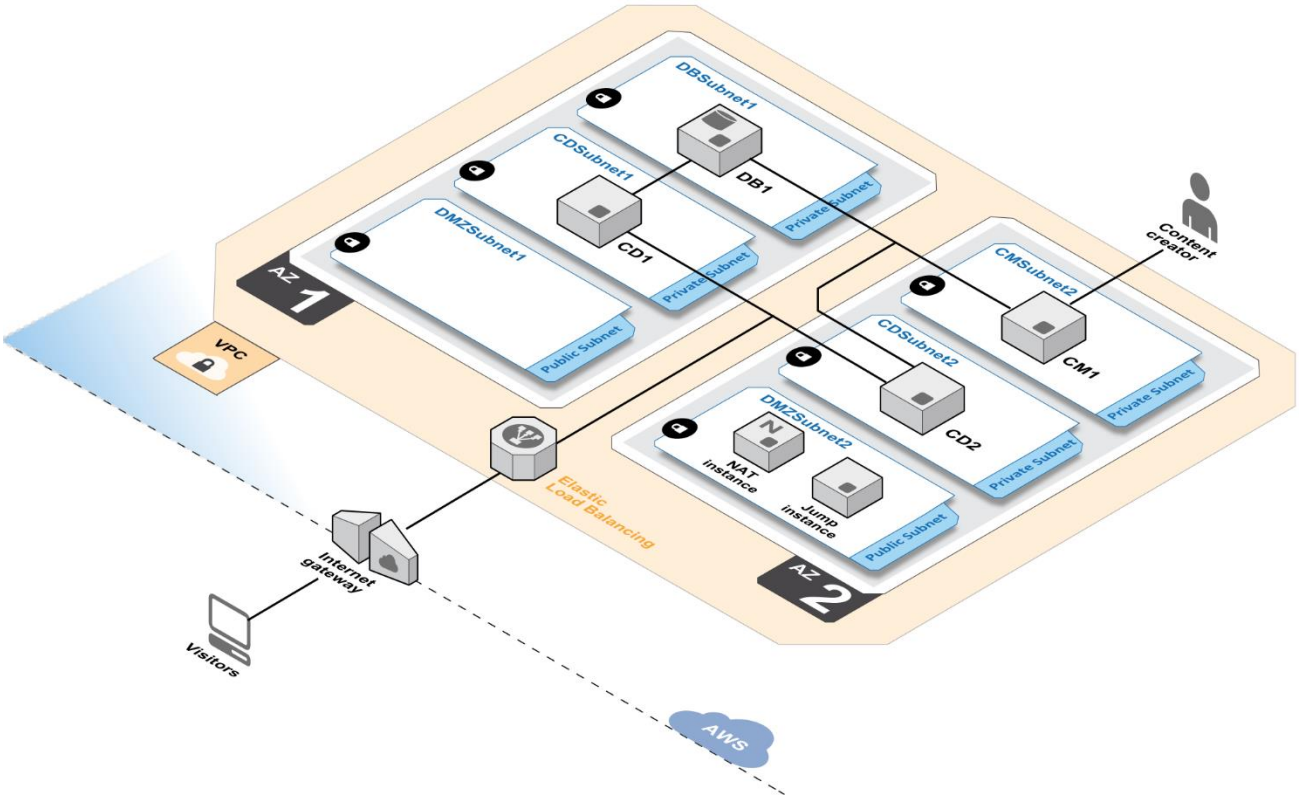

Figure 1: Quick Start Architecture for Sitecore on AWS

The AWS CloudFormation template creates a fully functional Sitecore 7.2 deployment. Sitecore includes three databases:

- Core – Stores Sitecore internal settings and configuration. All Sitecore content management and content delivery servers will need to be able to communicate with this database.
- Master – Stores all content revisions and artifacts. Sitecore content management servers will need to communicate with this database, but Sitecore content delivery servers will not.
- Web – This is where final content is stored. Once content is created by the Sitecore content management server, the creator will publish it to the web database. Sitecore content delivery servers use the web database as their source of content for rendering pages for visitors.

In this Sitecore Quick Start, we will be hosting all three databases on one Amazon Elastic Compute Cloud (Amazon EC2) instance that runs Microsoft SQL Server. For a discussion of additional database options, see the Database section later in this document.

The AWS CloudFormation template deploys the following components:

- An Amazon Virtual Private Cloud (Amazon VPC) with resources distributed across two Availability Zones.
- Public subnets in each Availability Zone that provide access to and from the Internet. One public subnet includes a network address translation (NAT) instance for outbound Internet access, and a Linux-based jump host to allow you to create an SSH tunnel to instances for inbound remote administrative access.
- Private subnets in each Availability Zone for running the application in a contained environment. These subnets will only be able to initiate outbound Internet traffic through the NAT, or receive traffic through an ELB load balancer.
- ELB load balancer in two public subnets in each Availability Zone to handle the incoming requests for the Sitecore content delivery servers.
- Security groups to tightly control the flow of traffic between your Amazon EC2 instances and the ELB load balancer.

AWS Services

The core AWS components used by this Quick Start include the following AWS services. (If you are new to AWS, see the Getting Started section of the AWS documentation.)

- Amazon VPC – The Amazon Virtual Private Cloud (Amazon VPC) service lets you provision a private, isolated section of the AWS cloud where you can launch AWS services and other resources in a virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways.
- Elastic Load Balancing – Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances in the cloud. It enables you to achieve greater levels of fault tolerance in your applications, seamlessly providing the required amount of load balancing capacity needed to distribute application traffic.
- Amazon EC2 – The Amazon Elastic Compute Cloud (Amazon EC2) service enables you to launch virtual machine instances with a variety of operating systems. You can choose from existing Amazon Machine Images (AMIs) or import your own virtual machine images.
- Amazon EBS – Amazon Elastic Block Store (Amazon EBS) provides persistent block-level storage volumes for use with Amazon EC2 instances in the AWS cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect

you from component failure, offering high availability and durability. Amazon EBS volumes provide the consistent and low-latency performance needed to run your workloads.

Design Considerations

Depending on the size and reach of your content and on the expected traffic volume, there are many ways in which you can deploy Sitecore in AWS to meet your requirements. With a decoupled Sitecore installation, you will also be able to scale the main components independently as your situation changes.

Database

The AWS CloudFormation template for this Quick Start deploys a single Amazon EC2 instance with Microsoft SQL Server installed. While this is a perfectly viable solution, there are many advantages to using Amazon Relational Database Service (Amazon RDS), such as automated backup, snapshots, automated patching and updating, and more.

It is best practice to run production database workloads in a highly available manner by leveraging multiple Availability Zones (Multi-AZ). Amazon RDS provides Multi-AZ configuration as a feature and handles replication between your master and secondary databases. In the unfortunate event of a database failure or connectivity issue, Amazon RDS will fail over to your secondary database automatically without requiring any changes to the application that uses the database.

The Sitecore installation package uses MDF and LDF files for the three databases it requires. Microsoft SQL Server needs to attach these files to create functional databases. The attach operation requires the files to be on your local file system. However, Amazon RDS is a managed service, so you will not have access to the underlying file system, and therefore you won't be able to directly utilize the MDF and LDF files.

To reap the benefits of Amazon RDS, you would have to incorporate an additional workflow of setting up a database that you do have file system access to (that is, an EC2 instance running Microsoft SQL Server) to attach the database files. When you have attached the files and created the databases, they will have to be extracted and exported to a collection of SQL statements that can be executed over the network to your RDS instance. The recommended tool for the extraction process is Microsoft SQL Server Database Publishing Wizard.

When you have established your databases on an RDS instance, you should take a snapshot before you install the Sitecore application. That way, you will always have a clean and

prepared database that you can launch and relaunch to enable tests, region migration, or additional deployments.

Instance Sizing

To select the appropriate instance type for servers in your Sitecore deployment, you should map the requirements in the Sitecore Installation Guide to compatible Amazon EC2 instance types.

The Sitecore Installation Guide lists only the *minimum* requirements, but AWS provides a variety of instance types that you can choose for larger or more complex workloads. If you intend to serve a considerable amount of traffic, you might consider instance types with features such as Amazon EBS optimization, enhanced networking, and high or 10-gigabit network performance, which results in higher performance (packets per second), lower latency, and lower jitter. For a list of EC2 instances that support enhanced networking, see Instances that Support Enhanced Networking in the AWS documentation.

Improving Performance with Amazon CloudFront

Amazon CloudFront is a web service that speeds up distribution of static and dynamic web content that Sitecore outputs. CloudFront delivers your content through a worldwide network of data centers called *edge locations*. When a user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency (time delay), so content is delivered with the best possible performance. If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. If the content is not currently in that edge location, CloudFront retrieves it from the Sitecore content delivery endpoint. CloudFront should be used as part of your production Sitecore deployment to deliver the best possible performance to your end users, but it is not included in this Quick Start.

Multi-Region Deployments

The Sitecore architecture is well suited for global distribution without having to replicate the entire environment. As stated earlier in the Architecture section, Sitecore consists of three databases: core, master, and web. For multi-region distribution, the web database is the only one that needs to be distributed.

Once you have multiple web databases in your deployment, you just have to configure remote publishing targets, one for each remote database. This is a setting on the content management servers that defines *where* content is deployed once published. It can even be set on a per-site basis, which could be useful for region-specific content.

> **Note** Multi-region distribution is done in Sitecore and is not a function of AWS. Please refer to the Sitecore Scaling Guide for details.

If you're using Amazon RDS, specifically its snapshot feature, you can easily distribute the web database by copying a snapshot to your new region and creating a new database from that, which is why it may be useful to have a snapshot with no content.

Sample Production Deployment

The following diagram represents the recommended minimal deployment for a production-ready installation of Sitecore. This setup provides high availability and resiliency against failure by spanning two Availability Zones, and manages performance by load-balancing the content delivery servers.

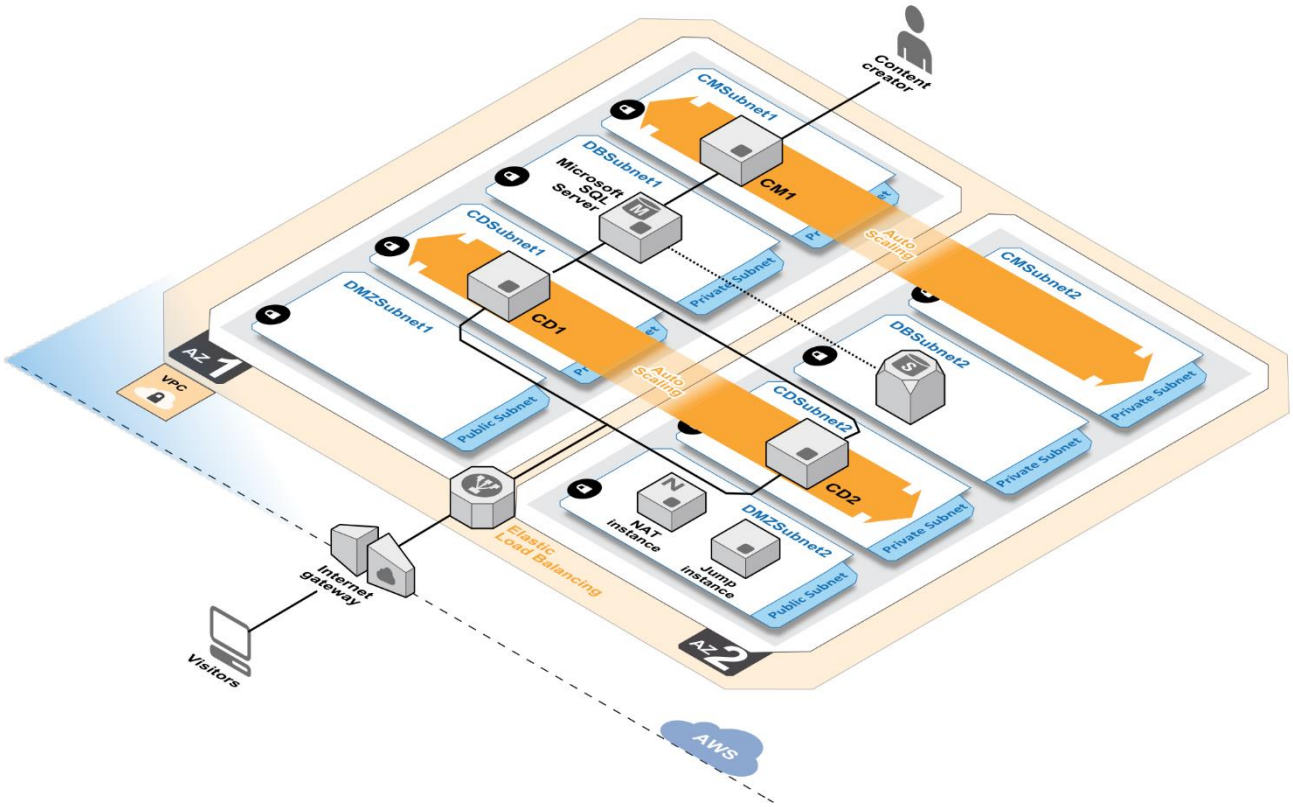

Figure 2. Minimal Production-Ready Architecture for Sitecore

Here are some key features of the architecture shown in Figure 2:

- Autoscaling the content management instances. The diagram shows only a single content management instance in an Auto Scaling group. The assumption here is that

your site will serve much more traffic from the content delivery servers than from the content management servers, so usually fewer content management instances than content delivery instances are required. In the proposed production deployment, if the content management instance were to become unreachable, the Auto Scaling group would automatically replace it in a capable subnet. Additionally, if you know that you will need more content management systems to support content creation, Auto Scaling groups can be scheduled to add instances. For example, if you know that your content creators generally work between 8 A.M. and 8 P.M., you could schedule your Auto Scaling group to add instances at 8 A.M. and to remove some at 8 P.M.

- Autoscaling the content delivery instances. The Sitecore application works very well with AWS Auto Scaling—no special handling is required for the delivery instances, which can be added and removed as traffic increases or decreases.
- Use of Amazon RDS. As mentioned in the Database section, Amazon RDS provides a significant advantage to applications running on AWS. Amazon RDS will provide this Sitecore installation with resiliency, durability, and automated backups.

What is not shown, but worth mentioning, is controlling access to your content management instance, as discussed later in the Secure Communication section. Access to the Sitecore content management application is controlled by a Sitecore-provided login page. It is best practice to restrict access to that instance by using conventional methods such as a VPN, or by AWS methods such as security group conditions that require a specific IP range.

Automated Deployment

The AWS CloudFormation template provided with this Quick Start bootstraps the AWS infrastructure and automates the deployment of Sitecore 7.2 on the AWS cloud from scratch. Follow the step-by-step instructions in this section to set up your AWS account, customize the template, and deploy the software into your account.

What We'll Cover

The procedure for deploying the Sitecore architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Prepare an AWS account

- Sign up for an AWS account, if you don't already have one.
- Choose the region where you want to deploy the stack on AWS.
- Create a key pair in the region. This key will be used to access the bastion host.

- Review account limits for Amazon EC2 instances, and request a limit increase, if needed.

Step 2. Launch the stack

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for parameters that require input.
- Review the other template parameters, and customize their values if necessary.

Step 3. Create a Sitecore website for testing

- Use the bastion host to create an SSH tunnel to your Sitecore content management instance.
- Use RDP to connect to the content management instance.
- Navigate to your Sitecore application by visiting `http://localhost/` in a browser.
- Log in to Sitecore and create your site.

Step 4. Test your Sitecore website

- Open a browser and enter the DNS name for the load balancer that was created.
- Navigate around your site and verify the content.

Step 1. Prepare an AWS Account

1. If you don't already have an AWS account, create one at http://aws.amazon.com by following the on-screen instructions. Part of the sign-up process involves receiving a phone call and entering a PIN using the phone keypad.
2. Use the region selector in the navigation bar to choose the Amazon EC2 region where you want to deploy Sitecore on AWS.

   Amazon EC2 locations are composed of *regions* and *Availability Zones*. Regions are dispersed and located in separate geographic areas. This Quick Start uses the m4.large and m4.xlarge instance types for the Sitecore portion of the deployment. M4 instances are currently available in all AWS regions except AWS GovCloud (US), China (Beijing), and South America (São Paulo).

Figure 3: Choosing an Amazon EC2 Region

> **Tip** Consider choosing a region closest to your data center or corporate network to reduce network latency between systems running on AWS and the systems and users on your corporate network.

3. Create a key pair in your preferred region. To do this, in the navigation pane of the Amazon EC2 console, choose **Key Pairs**, **Create Key Pair**, type a name, and then choose **Create**.

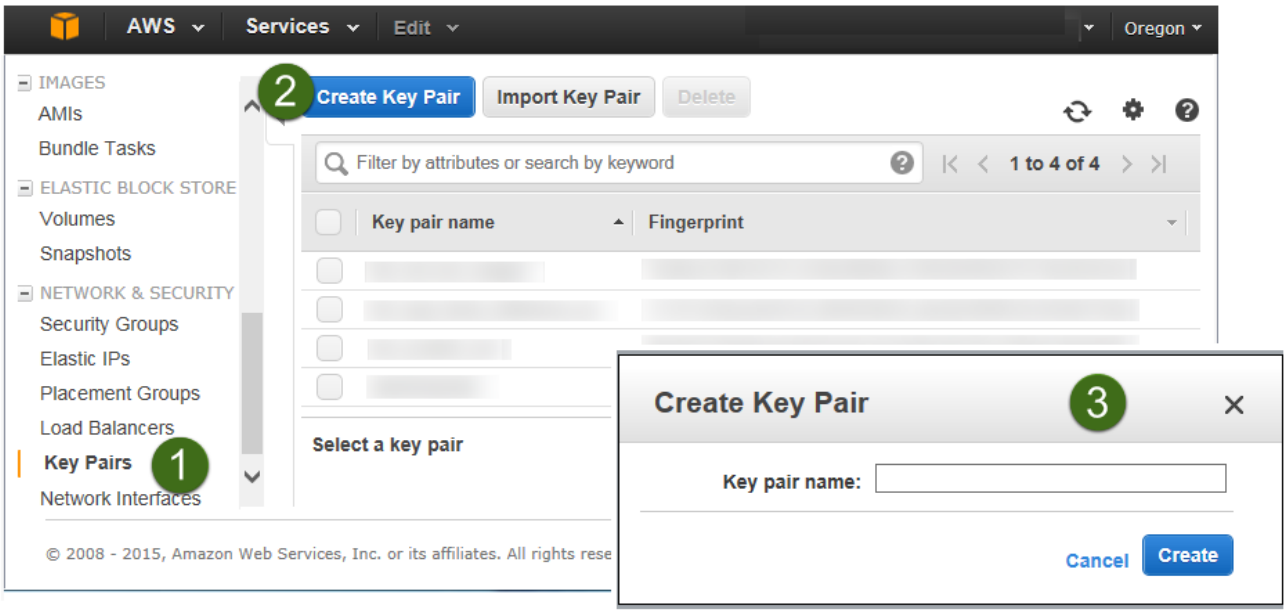

Figure 4: Creating a Key Pair

This key pair will be used to tunnel into your content management instance.

4. If necessary, request a service limit increase for your desired Amazon EC2 instance type. To do this, in the AWS Support Center, choose **Create Case**, **Service Limit Increase**, **EC2 instances**, and then complete the fields in the limit increase form. The current default limit is 20 instances.

   You might need to request an increase if you already have an existing deployment that uses this instance type, and you think you might exceed the default limit with this reference deployment. It might take a few days for the new service limit to become effective. For more information, see Amazon EC2 Service Limits in the AWS documentation.

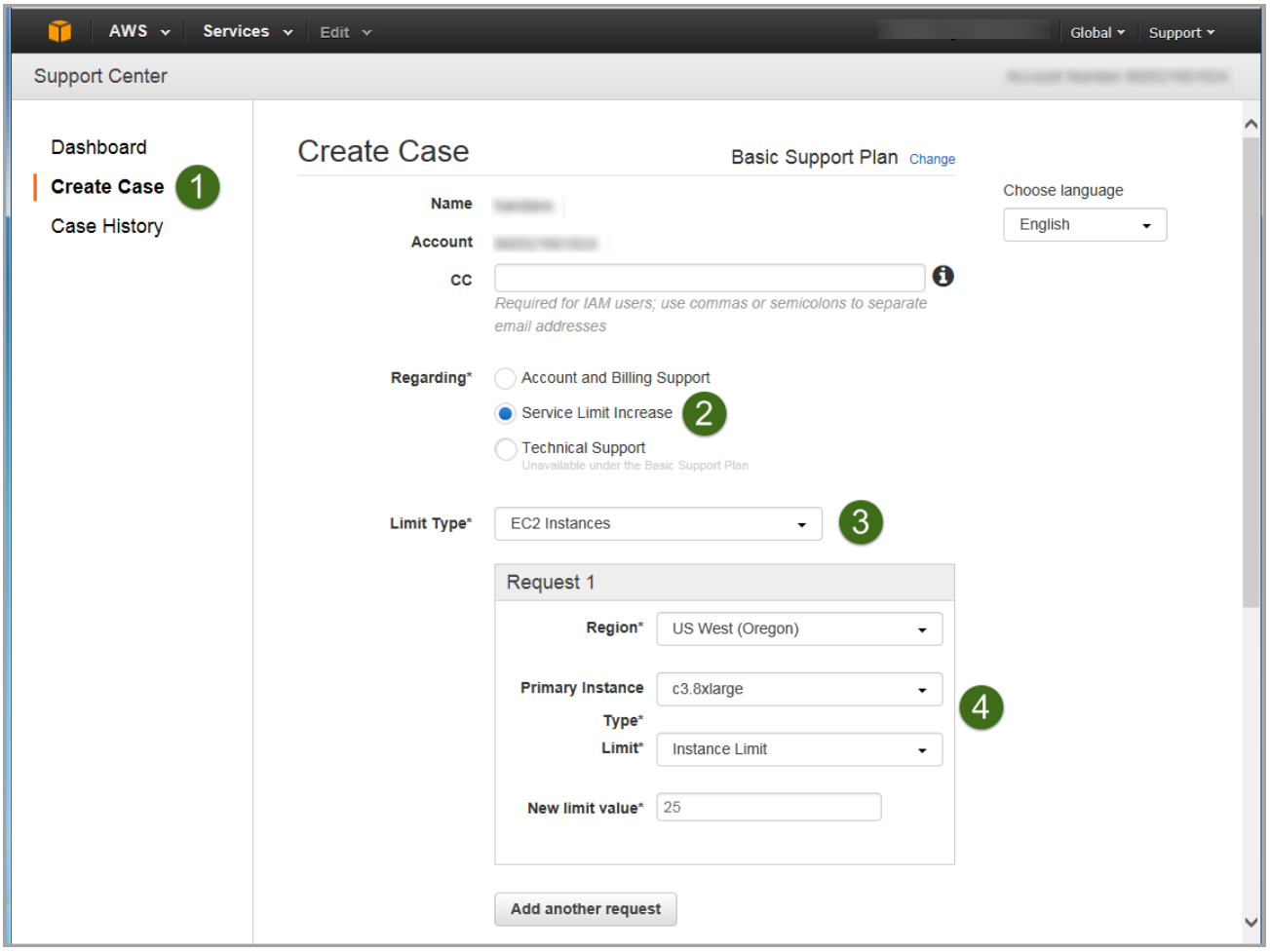

Figure 5: Requesting a Service Limit Increase

Step 2. Launch the Sitecore Stack

This automated AWS CloudFormation template deploys Sitecore 7.2 in multiple Availability Zones into an Amazon VPC. Please make sure that you've created a key pair in your chosen region before launching the stack.

1. Launch the AWS CloudFormation template into your AWS account.

   Launch

   The template is launched in the US West (Oregon) region by default. You can change the region by using the region selector in the navigation bar.

   This stack takes less than 30 minutes to create.

   Note You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using this Quick Start. As of the date of publication, the cost for using the Quick Start with default settings is approximately $2 an hour. Prices are subject to change. See the pricing pages for each AWS service you will be using in this Quick Start for full details.

   You can also download the template to use it as a starting point for your own implementation.

2. On the **Select Template** page, keep the default URL for the AWS CloudFormation template, and then choose **Next**.

3. On the **Specify Details** page, review the parameters for the template. These are described in the following table.

   Provide values for the *AWSKeyName* and *FileSitecoreLicense* parameters. For all other parameters, the template provides default settings that you can customize.

Parameter	Default	Description
AWSKeyName	*Requires input*	Amazon EC2 key that will be used to fetch the Windows password. When you created an AWS account, this is the key pair you created in your preferred region.
AdminPassword	Password123	Password for the local administrator account on each server. This must be at least 8 characters, including letters, numbers, and symbols.
NATInstanceType	t2.small	Amazon EC2 instance type for the NAT instance.

Parameter	Default	Description
SitecoreCMInstanceType	m4.large	Amazon EC2 instance type for the Sitecore content management instances.
SitecoreCDInstanceType	m4.large	Amazon EC2 instance type for the Sitecore content distribution instance.
SitecoreDBInstanceType	m4.xlarge	Amazon EC2 instance type for the Sitecore database instance.
FileSitecoreLicense	*Requires input*	Download location of the Sitecore license file.

Note You can also download the template and edit it to create your own parameters based on your specific deployment scenario.

4. On the **Options** page, you can specify tags (key-value pairs) for resources in your stack and set additional options. When you're done, choose **Next**.
5. On the **Review** page, review and confirm the template settings. Under **Capabilities**, select the check box to acknowledge that the template will create IAM resources.
6. Choose **Create** to deploy the stack.
7. Monitor the status of the stack. When the status displays **CREATE_COMPLETE**, the Sitecore cluster is ready.

Step 3. Create a Sitecore Website for Testing

By default, Sitecore does not provide you with a demo site. You will first have to create one by using the Sitecore application on the content management server. You will need to visit the public endpoint for the hosting instance and log in with Sitecore application credentials. Then you will use the Sitecore software to deploy your first site.

1. Refer to the template output parameter *CM1URL*. This is the URL where you can access your Sitecore content management host. Copy and paste this URL into a browser.
2. Log in to your new Sitecore installation. The default user name is **admin** and the default password is **b**. We recommend that you change these after your first login.
3. Create a basic website to verify and publish.

Step 4. Test your Sitecore Website

Once step 3 is complete, the Sitecore content management software publishes your website to the *Web* database where the Sitecore content delivery instances get their content. The content delivery servers are behind an ELB load balancer, so to validate your newly created Sitecore website, you point your browser to the DNS name that the ELB provides.

1. Refer to the template output parameter *FrontendDNS*. This is the URL where you can access your Sitecore website. Copy and paste this URL into a browser.
2. Verify that the website is the result of what you created and published in step 3.

Security

When you build systems on the AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. In turn, you assume responsibility and management of the guest operating system (including updates and security patches), other associated applications, as well as the configuration of the AWS-provided security group firewall. For more information about security on AWS, visit the AWS Security Center.

Secure Communication

Web applications should leverage SSL/HTTPS whenever possible to reduce the chance of sensitive data being intercepted in transit. This is especially important if your Sitecore page prompts for user input, especially of sensitive information. Putting your Sitecore content delivery servers behind an ELB load balancer will allow you to configure traffic encryption between your load balancer and the clients that initiate HTTPS sessions.

Security for your content management server is also extremely critical. Depending on how that content is created, there are different ways you can protect it. If you choose to expose your content management server endpoint to the public Internet to allow contributors to access it from the public Internet, enforcing HTTPS is of utmost importance. Alternatively, you could launch your content management servers into private subnets (with no direct access to the public Internet) and require that contributors access the application over a VPN connection, an SSH tunnel, or a Remote Desktop Gateway.

Application Security with AWS WAF

From an application security perspective, using a web application firewall plays a key role in protecting your Sitecore deployment. If you use Amazon CloudFront to improve your site loading speed, you can also utilize AWS WAF. AWS WAF is a web application firewall that helps protect your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources. AWS WAF gives you control over which traffic to allow or block to your web application by defining customizable web security rules. You can use AWS WAF to create custom rules that block common attack patterns, such as SQL injection or cross-site scripting, and rules that are designed for your specific application. New rules can be deployed within minutes so that you can respond quickly to changing traffic patterns.

Security Groups

A security group acts as a firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time. The new rules are automatically applied to all instances that are associated with the security group.

The security groups created and assigned to the individual instances as part of this solution are restricted as much as possible while allowing access to the various functions needed by Sitecore. We recommend reviewing security groups to further restrict access as needed once the deployment is up and running.

Additional Resources

AWS services

- AWS CloudFormation
  https://aws.amazon.com/documentation/cloudformation/
- Amazon CloudFront
  https://aws.amazon.com/documentation/cloudfront/
- Amazon EBS
  http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html
- Amazon EC2
  https://aws.amazon.com/documentation/ec2/

- Elastic Load Balancing
  https://aws.amazon.com/documentation/elastic-load-balancing/
- Amazon RDS
  https://aws.amazon.com/documentation/rds/
- Amazon VPC
  https://aws.amazon.com/documentation/vpc/
- AWS WAF
  https://aws.amazon.com/documentation/waf/

Sitecore

- Sitecore website
  http://www.sitecore.net
- Sitecore Scaling Guide
  https://sdn.sitecore.net/upload/sitecore7/70/scaling_guide_sc70_a4.pdf
- Sitecore Installation Guide
  https://sdn.sitecore.net/upload/sitecore7/70/installation_guide_sc70-a4.pdf

Tools

- Sitecore PowerShell Deployment Framework
  https://github.com/adoprog/Sitecore-PowerCore
- Microsoft SQL Server Database Publishing Wizard
  http://weblogs.asp.net/scottgu/recipe-deploying-a-sql-database-to-a-remote-hosting-environment-part-1

Quick Start Reference Deployments

- AWS Quick Start home page
  https://aws.amazon.com/quickstart/
- Quick Start deployment guides
  https://aws.amazon.com/documentation/quickstart/

Send Us Feedback

We welcome your questions and comments. Please post your feedback on the AWS Quick Start Discussion Forum.

You can visit our GitHub repository to download the templates and scripts for this Quick Start, and to share your customizations with others.

Document Revisions

Date	Change	In sections
November 2015	Initial publication	—

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.