Getting Started with AWS

Hosting WordPress on AWS

Getting Started with AWS: Hosting WordPress on AWS

Copyright © 2016 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Table of Contents

Hosting WordPress on Amazon Web Services

Amazon Web Services (AWS) provides on-demand computing resources and services in the cloud, with pay-as-you-go pricing.

You can use AWS Elastic Beanstalk to deploy the WordPress application on AWS in a matter of minutes. Elastic Beanstalk handles the details of your hosting environment, including provisioning AWS resources such as application servers, and configuring load balancing, scaling, and monitoring.

Hosting Architecture

To host WordPress on AWS, you'll use the following AWS resources:

- Application servers from Amazon Elastic Compute Cloud (Amazon EC2), known as *instances*
- Storage space from Amazon Simple Storage Service (Amazon S3), known as a *bucket*
- A managed relational database from Amazon Relational Database Service (Amazon RDS), known as a *DB instance*
- A *load balancer* from the Elastic Load Balancing service, to distribute traffic to your application servers
- Scaling services from the Auto Scaling service, to ensure that you have a minimum number of available application servers, and can add or remove application servers as demand on your WordPress website or blog changes.

You'll also install a WordPress plugin that enables WordPress to use Amazon S3 as a content delivery network (CDN).

The following diagram shows the architecture for this tutorial:
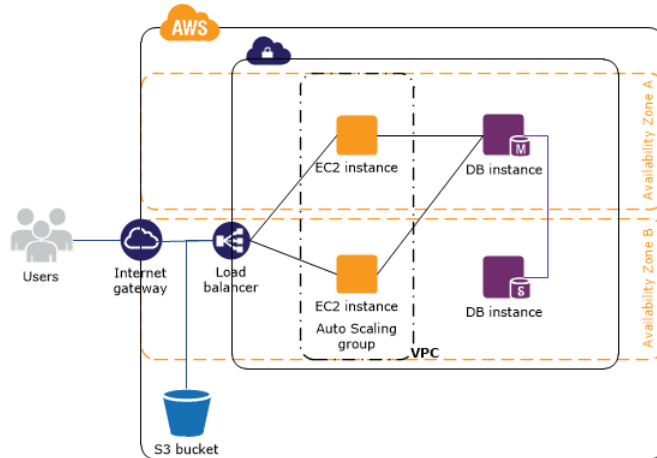
Tutorial

This tutorial walks you through the process of deploying the WordPress application on AWS. We'll use the AWS Management Console to access AWS.

Steps

Pricing

You can use the AWS Simple Monthly Calculator to estimate what it would cost to host your WordPress website on AWS.

Note that if you created your AWS account within the last 12 months, you are eligible for the AWS Free Tier.

For more information about the estimated costs for this tutorial, see Build a WordPress Website: Services Used and Costs. For more information about AWS pricing, see Pricing.

Step 1: Prepare Storage for Your Static Assets

You can store the static assets for your WordPress website, such as a media library and theme files, in Amazon S3. You must create an S3 bucket and create an AWS Identity and Access Management (IAM) user with permission to store assets in your S3 bucket.

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS and you can start using them immediately. You are charged only for the services that you use.

If you created your AWS account less than 12 months ago, you can get started with AWS for free. For more information, see AWS Free Tier.

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

To create an AWS account

1. Open http://aws.amazon.com/, and then choose **Create an AWS Account**.
2. Follow the online instructions.

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

Create an Amazon S3 Bucket

Use the AWS Management Console to create a bucket for the static assets for your WordPress website.
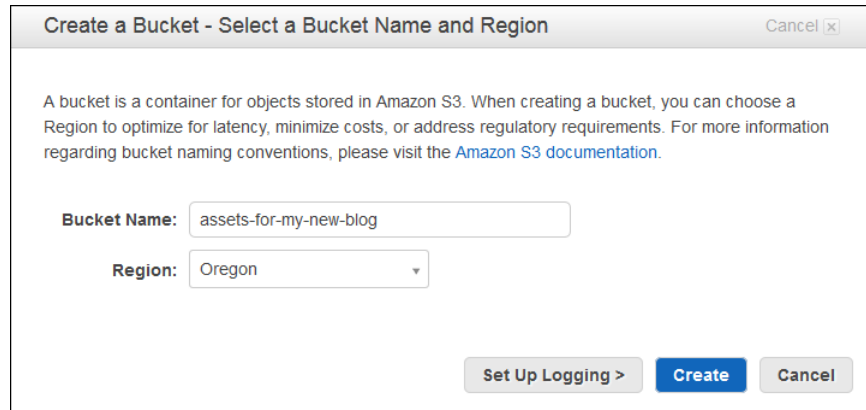
To create an S3 bucket

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose **Create Bucket**.
3. For **Bucket Name**, type a name for the bucket.

4. For **Region**, choose a region that is close to you, which reduces latency and minimizes costs.

> **Warning**
> The WordPress plugin that you will use to enable WordPress to use Amazon S3 as a CDN does not support regions introduced after January 2014. You can't complete this tutorial if you select an unsupported region. For more information, see W3 Total Cache is not compatible with newest S3 regions.

5. Choose **Create**.

Create an IAM User

Create an IAM user that can list all your S3 buckets and has full access to the S3 bucket that you just created. You'll provide the credentials for this IAM user to a WordPress plugin that enables you to store static assets for your WordPress website in Amazon S3.

1. Open the Identity and Access Management (IAM) console at https://console.aws.amazon.com/iam/.
2. On the navigation pane, choose **Users**.
3. Choose **Create New Users**.
4. On the **Create User** page, type a user name (for example, wp-s3-user), select **Generate an access key for each user**, and then choose **Create**.

> **Important**
> Save the credentials for this user in a safe location.

5. Choose the name (not the check box) of the IAM user.
6. On the **Permissions** tab, under **Inline Policies**, choose **click here** to create an inline policy.
7. Select **Custom Policy** and then choose **Select** to open the policy editor.
8. For **Policy Name**, type a name for the policy (for example, AllowWordPressBucketAccess). For **Policy Document**, copy and paste the following policy document, and then choose **Apply Policy**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListAllMyBuckets"],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": ["s3:*"],
      "Resource": [
        "arn:aws:s3:::assets-for-my-new-blog",
        "arn:aws:s3:::assets-for-my-new-blog/*"
      ]
    }
  ]
}
```

Step 2: Create a Database

Amazon RDS provides managed relational databases. The basic building block is a *DB instance*, which is an isolated database environment in the AWS Cloud. A DB instance can contain multiple databases.
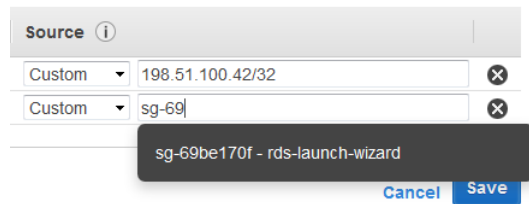
In this step, you launch a *Multi-AZ DB instance*. In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups. In the event of a planned or unplanned outage of your DB instance, Amazon RDS automatically switches to the standby replica.

To launch a DB instance

1. Open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. On the navigation bar, select the same region that you used to create your S3 bucket.
3. On the navigation pane, choose **Instances**.
4. Choose **Launch DB Instance**.
5. On the **Select Engine** page, on the **MySQL** tab, choose **Select**.
6. On the **Do you plan to use this database for production purposes** page, under **Production**, select **MySQL**, and then choose **Next Step**.
7. On the **Specify DB Details** page, do the following:

    a. (Optional) For **DB Instance Class**, select an instance size that is large enough to meet the demand of your production workload. We recommend that you start with **db.r3.large** and perform load testing to determine whether it meets your needs.

    b. For **Multi-AZ Deployment**, we recommend that you select **Yes**.

    c. For **DB Instance Identifier**, type a unique name for the instance (for example, `my-db-instance-for-wordpress`).

    d. For **Master Username**, type a login for the master user (for example, `master_user`).

    e. Type a password for the master user in **Master Password** and **Confirm Password**. Record your password in a safe place.

    f. Choose **Next Step**.

8. On the **Configure Advanced Settings** page, do the following:

    a. Keep the default virtual private cloud (VPC) and default subnet group selected.

      b.   For **Publicly Accessible**, select **No**.

      c.   For **Database Name**, type a name for the initial database (for example, `my_wordpress_database`).

      d.   For **Enhanced Monitoring**, select **No**.

      e.   Choose **Launch DB Instance**.

9.   After a few minutes, the launch completes. When you see the message that your instance is being created, choose **View Your DB Instances**.

     Initially, the status of your DB instance is `creating`. After the status changes to `available`, your DB instance ready for use.

10. Copy **Endpoint**, omitting the port information at the end (for example, omit `:3306`), and save it for future use in this tutorial.

11. On the **Details** tab, under **Security and Network**, open the link to view the security group in the Amazon EC2 console. On the **Inbound** tab, choose **Edit**. By default, there is a rule based on the DB engine that you chose. Add another rule with the same **Type**. For **Source**, specify the ID of the security group itself; start typing the ID and then select the security group from the list. This allows resources with this security group to receive traffic from the database on the database port. Choose **Save**.

Step 3: Download WordPress

To prepare to deploy WordPress using AWS Elastic Beanstalk, you must copy the WordPress files to your computer and provide some configuration information. AWS Elastic Beanstalk requires a source bundle, in the format of a ZIP or WAR file.

To download WordPress and create a source bundle

1. Open http://wordpress.org/download/.
2. Download the latest release.
3. Extract the files from the download to a folder on your local computer.
4. Create a copy of the `wp-config-sample.php` file and name it `wp-config.php`. This creates a new configuration file and keeps the original sample file intact as a backup.
5. Open the `wp-config.php` file in a text editor.
6. To configure WordPress to use your Amazon RDS database, replace the following database settings:

```
define('DB_NAME', 'database_name_here');
define('DB_USER', 'username_here');
define('DB_PASSWORD', 'password_here');
define('DB_HOST', 'localhost');
```

Copy and paste these database settings, which use environment variables that you will set to provide connection information for your Amazon RDS database to WordPress:

```
define('DB_NAME',     $_SERVER["RDS_DB_NAME"]);
define('DB_USER',     $_SERVER["RDS_USERNAME"]);
define('DB_PASSWORD', $_SERVER["RDS_PASSWORD"]);
define('DB_HOST',     $_SERVER["RDS_HOSTNAME"]);
```

7. To improve the security of your WordPress website, replace the following authentication settings:

```
define('AUTH_KEY',         'put your unique phrase here');
define('SECURE_AUTH_KEY',  'put your unique phrase here');
define('LOGGED_IN_KEY',    'put your unique phrase here');
define('NONCE_KEY',        'put your unique phrase here');
define('AUTH_SALT',        'put your unique phrase here');
define('SECURE_AUTH_SALT', 'put your unique phrase here');
```

```
define('LOGGED_IN_SALT',    'put your unique phrase here');
define('NONCE_SALT',        'put your unique phrase here');
```

Copy and paste these authentication settings, which use environment variables that you will set to provide information to WordPress:

```
define('AUTH_KEY',         $_SERVER["AUTH_KEY"]);
define('SECURE_AUTH_KEY',  $_SERVER["SECURE_AUTH_KEY"]);
define('LOGGED_IN_KEY',    $_SERVER["LOGGED_IN_KEY"]);
define('NONCE_KEY',        $_SERVER["NONCE_KEY"]);
define('AUTH_SALT',        $_SERVER["AUTH_SALT"]);
define('SECURE_AUTH_SALT', $_SERVER["SECURE_AUTH_SALT"]);
define('LOGGED_IN_SALT',   $_SERVER["LOGGED_IN_SALT"]);
define('NONCE_SALT',       $_SERVER["NONCE_SALT"]);
```

8. Save the `wp-config.php` file and close the text editor.
9. Create a ZIP file from the files and folders in the WordPress folder (not the parent directory), using one of the following methods, depending on your operating system:

    *   Windows — In Windows Explorer, select the files and folders, right-click, and then choose **Send to**, **Compressed (zipped) Folder**. Name the file `wordpress-x.y.z.zip`, where x.y.z is the version of WordPress.

    *   Mac OS X and Linux — Use the following command, where x.y.z is the version of WordPress:

    ```
    zip -r ../wordpress-x.y.z.zip .
    ```

Step 4: Deploy WordPress

AWS Elastic Beanstalk makes it easy to deploy, manage, and scale your WordPress site on AWS.

> **Note**
> In this tutorial, we install WordPress over HTTP on a public endpoint. To increase security, you could install WordPress on a local server first, and deploy using Elastic Beanstalk only after setting the administrator password.

Prerequisites

- Create a key pair as described in Creating Your Key Pair in the *Amazon EC2 User Guide for Linux Instances*. You must specify a key pair when you configure you application servers or you can't connect to them.
- Create a DB instance using Amazon RDS as described in Step 2: Create a Database (p. 6). You will configure environment properties that Elastic Beanstalk will use to pass the database connection information to WordPress.
- Create a ZIP file for WordPress as described in Step 3: Download WordPress (p. 8). You must specify this file when you launch the application server.

Launch the Application Server Using Elastic Beanstalk

Provide information about your application version to Elastic Beanstalk. Elastic Beanstalk provisions the AWS resources that you need to run WordPress, such as an EC2 instance.

To launch WordPress using Elastic Beanstalk

1. Open the Elastic Beanstalk console at https://console.aws.amazon.com/elasticbeanstalk/.
2. On the navigation bar, select the same region that you used to create your S3 bucket.
3. Choose **Create New Application**.
4. Type a name and description for your application, and then choose **Next**.

5. On the **New Environment** page, choose **Create web server**.

6. For **Predefined configuration**, select PHP. For **Environment type**, keep Load balancing, auto scaling. Using load balancing and auto scaling enables incoming web traffic for your website to be routed across a fleet of virtual servers that can scale automatically as demand on your website changes. Choose **Next**.

7. Upload your ZIP file as the source of the initial application version, and then choose **Next**.

8. Type a name and URL for your environment. Choose **Check availability** to verify that this name is available, and then choose **Next**.

9. On the **Additional Resources** page, choose **Next**.

> **Warning**
> For a production WordPress website, we recommend that you do not create the DB instance using Elastic Beanstalk. If Elastic Beanstalk creates the DB instance, it will delete the DB instance when you terminate the environment and you will lose your data.

10. On the **Configuration details** page, do the following:

    a. For **Instance type**, select **t2.large**.

    b. For **EC2 key pair**, select your key pair.

    c. For **Application health check URL**, type `/readme.html` to configure the load balancer to send health check requests to that file in the application root directory using HTTP on port 80.

    d. Choose **Next**.

11. (Optional) On the **Environment Tags** page, you can tag your environment, or continue to the next step. To add a tag, specify its key and value. When you are finished, choose **Next**.

12. On the **Permissions** page, choose **Next**.

13. On the **Review** page, choose **Launch**.

14. As Elastic Beanstalk creates your AWS resources, it adds information about them under **Recent Events**.

    After the launch is complete, continue to the next step.

15. On the navigation pane, choose **Configuration**.

16. For **Scaling**, choose the gear icon.

17. Under **Auto Scaling**, set **Minimum instance count** and **Maximum instance count** to 1, and then choose **Apply**. Note that this temporarily restricts the Auto Scaling group to one instance. We recommend that you do this whenever you configure or update WordPress.

    After the environment update is complete, continue to the next procedure.

Update Your Configuration

You must associate the security group for your DB instance with the EC2 instances that Elastic Beanstalk launches, provide information about your DB instance, and provide authentication settings for WordPress.

Prerequisite

Open https://api.wordpress.org/secret-key/1.1/salt/. This site generates settings that you can use for the environment variables that you specified for the authentication settings in your `wp-config.php` file. Keep these settings handy while you complete this procedure using the Elastic Beanstalk console.

To update your configuration

1. On the navigation pane, choose **Configuration**.

2. For **Instances**, choose the gear icon.

3. For **EC2 security groups**, go to the end of the current text, type a comma, and then type the name of the security group for your DB instance (for example, rds-launch-wizard). Choose **Apply**. Read the warning, and then choose **Save**.

    After the environment update is complete, continue to the next step.

4. On the navigation pane, choose **Configuration**.

5. For **Software Configuration**, choose the gear icon.

6. For **Environment Properties**, define the following properties to allow Elastic Beanstalk to access your DB instance:

    - RDS_DB_NAME — The name of the database
    - RDS_USERNAME — The name of the master user
    - RDS_PASSWORD — The password for the master user
    - RDS_HOSTNAME — The endpoint of the DB instance (from the Amazon RDS console)
    - RDS_PORT — The port of the DB instance (for example, 3306)

    Define the following authentication settings:

    - AUTH_KEY — The value generated for AUTH_KEY.
    - SECURE_AUTH_KEY — The value generated for SECURE_AUTH_KEY.
    - LOGGED_IN_KEY — The value generated for LOGGED_IN_KEY.

- NONCE_KEY — The value generated for NONCE_KEY.
- AUTH_SALT — The value generated for AUTH_SALT.
- SECURE_AUTH_SALT — The value generated for SECURE_AUTH_SALT.
- LOGGED_IN_SALT — The value generated for LOGGED_IN_SALT.
- NONCE_SALT — The value generated for NONCE_SALT.

7. Choose **Apply**. After the environment update is complete, continue to the next procedure.

Install WordPress

After your application server is launched, you can install WordPress.

To install WordPress

1. Choose the URL of the environment (for example, `mywordpressblog-env.us-west-2.elasticbeanstalk.com`). If everything is working, the browser displays the WordPress installation page.
2. Select a language and choose **Continue**.
3. Provide a site title, user name, and email address. Be sure to save the password in a safe place, and then choose **Install WordPress**.

4. After the install finishes, you'll see a success message. Choose **Log In**.
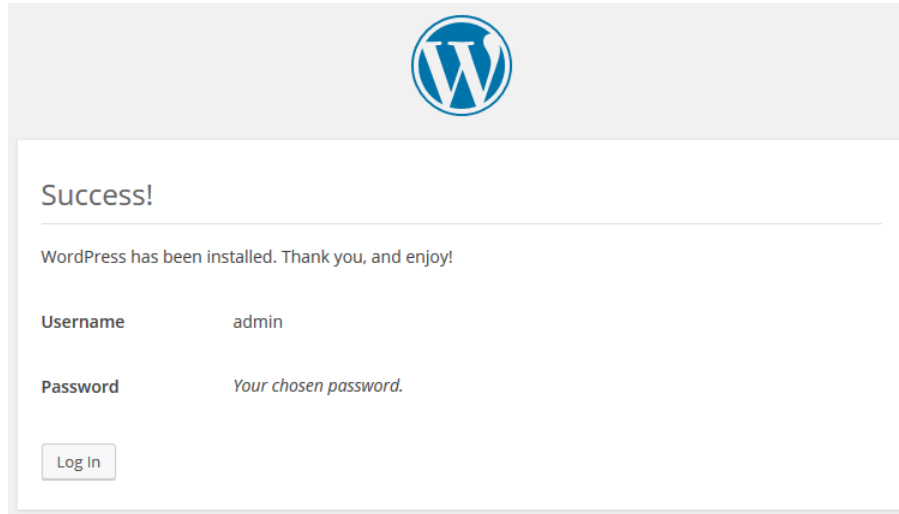
5. Log in using your user name and password.
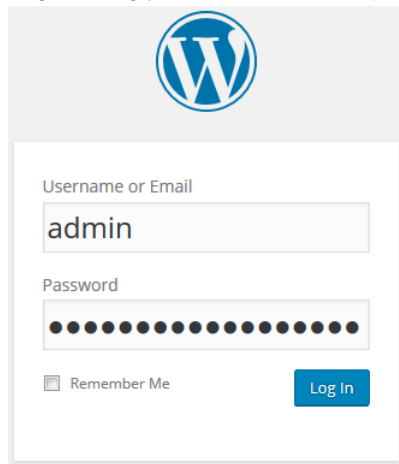
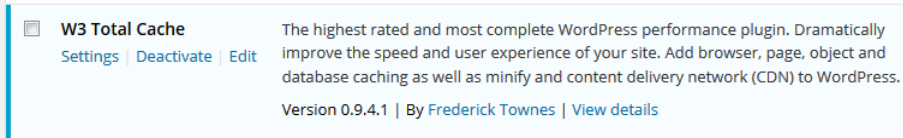# Enable WordPress to Store Assets in Amazon S3

Install the W3 Total Cache plugin to enable WordPress to store assets in Amazon S3.

To install the W3 Total Cache plugin

1. From the WordPress dashboard, on the navigation pane, choose **Plugins**.
2. On the **Plugins** page, choose **Add New**.
3. Search for the plugin named **W3 Total Cache**. Choose **Install Now**.
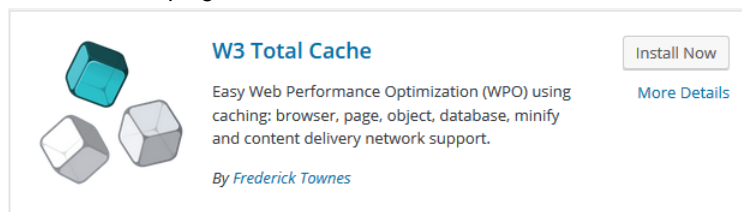
4. After the plugin is installed, choose **Activate Plugin**.

5. On the **Plugins** page, for the **W3 Total Cache** plugin, choose **Settings**.
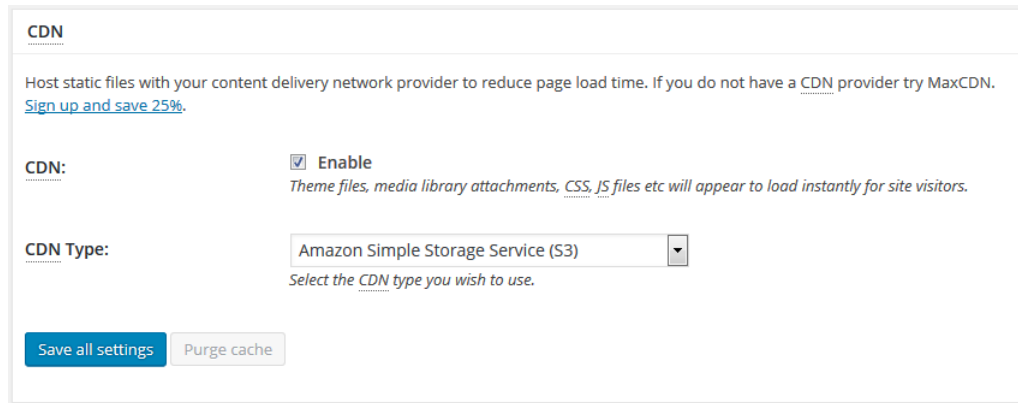
☐ **W3 Total Cache**   The highest rated and most complete WordPress performance plugin. Dramatically
Settings | Deactivate | Edit   improve the speed and user experience of your site. Add browser, page, object and
database caching as well as minify and content delivery network (CDN) to WordPress.

Version 0.9.4.1 | By Frederick Townes | View details

6. Scroll to the **CDN** pane. For **CDN**, select **Enable**. For **CDN type**, select **Amazon Simple Storage
Service (S3)**. Choose **Save all settings**.

CDN

Host static files with your content delivery network provider to reduce page load time. If you do not have a CDN provider try MaxCDN.
Sign up and save 25%.

CDN: ☑ Enable
Theme files, media library attachments, CSS, JS files etc will appear to load instantly for site visitors.

CDN Type: Amazon Simple Storage Service (S3) ▾
Select the CDN type you wish to use.

[Save all settings] [Purge cache]

7. From the navigation pane, choose **CDN**.

8. Scroll to the **Configuration** pane and do the following:

   a. For **Access key ID**, copy and paste the access key ID for the IAM user that you created.

   b. For **Secret key**, copy and paste the secret key for the IAM user that you created.

   c. For **Bucket**, type the name of the S3 bucket that you created.

   d. Choose **Save all settings**.

9. (Optional) You can change other plugin settings as needed. For more information, see the W3 Total Cache Installation page.

Store WordPress Assets in Amazon S3

Use the W3 Total Cache plugin to store your WordPress assets in Amazon S3.

To store WordPress assets in your S3 bucket

1. Locate the following message on the **Plugins** or **Dashboard** page.

2. Choose **wp-includes**. On the **Includes files export** page, choose **Start**. If your permissions have been set up correctly, the files are uploaded to Amazon S3. After the upload is complete, close the window.

3. (Optional) Open the Amazon S3 console and verify that these files have been stored in your S3 bucket.

4. Choose **theme files**. On the **Theme files export** page, choose **Start**.

5. Choose **custom files**. On the **Custom files export** page, choose **Start**.

6. (Optional) After uploading all files, choose **Hide this message**.

7. (Optional) On the **Dashboard** page, in the **Welcome to WordPress** pane, choose **View your site**. Use your browser to view the source for the page. Notice that the URLs for the stylesheets and other assets point to your S3 bucket.

8. If you change the theme using the **Appearance** page, W3 Total Cache prompts you to upload the updated theme files.

Use a Custom Domain Name

If you own a domain name, you can associate it with the domain name provided by Elastic Beanstalk so that your users can access your WordPress site using your domain name. You can use your DNS service, such as your domain registrar, to create a CNAME record to route queries to your WordPress website. For more information, see the documentation for your DNS service. Alternatively, you can use Amazon Route 53 as your DNS service, and configure it to route queries to your WordPress website. If you don't have a domain name for your WordPress site, you can purchase one using Amazon Route 53.

For more information, see Your Elastic Beanstalk Environment's Domain Name in the *AWS Elastic Beanstalk Developer Guide*.

Step 5: Update the Application Version

In the process of configuring WordPress and installing the WordPress plugin, you have changed the application server. If Auto Scaling were to launch a new application server in response to a scale out event or terminate the current application server and replace it, these changes will not be applied to the new application server. To ensure that a newly launched application server matches the current application server, create a new application version using Elastic Beanstalk.

Note that you must follow this procedure whenever you install WordPress updates, change the configuration of your WordPress site, or install new plug-ins.

Prerequisites

Ensure that you have only 1 instance running in your Auto Scaling group before you make any changes to your WordPress installation.

**To restrict your Auto Scaling group to 1 instance**

1. Open the Elastic Beanstalk console at https://console.aws.amazon.com/elasticbeanstalk/.
2. Select the environment.
3. On the navigation pane, choose **Configuration**.
4. For **Scaling**, choose the gear icon.
5. For **Minimum instance count** and **Maximum instance count**, type 1, and then choose **Apply**.
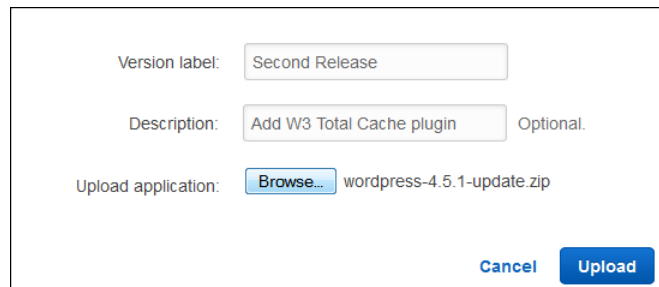
Add a New Application Version Using Elastic Beanstalk

Each application version has a unique ZIP or WAR file, as well as information about the version.

To add a new application version

1. After you finish updating your WordPress installation, use SCP or WinSCP to download the web folder from the application server (/var/app/current) to your local computer. You'll need the public DNS name of the application server and the path to the .pem file for your key pair.

2. Create a new .zip file from the files that you downloaded.

3. Open the Elastic Beanstalk console at https://console.aws.amazon.com/elasticbeanstalk/.

4. Choose **Actions**, **View Application Versions**.
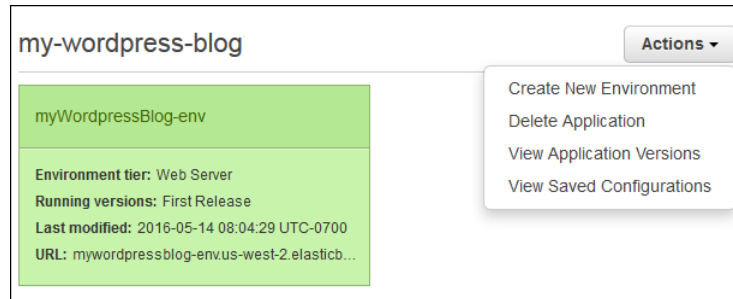
5. Choose **Upload**.

6. On the upload page, do the following:

   a. Type a version label.

   b. (Optional) Type a description.

   c. Choose **Browse** and select the new .zip file as the source of the new application version.

   d. Choose **Upload**.

7. If you reduced the size of your Auto Scaling group, restore it.

   a. On the navigation pane, choose **Configuration**.

   b. For **Scaling**, choose the gear icon.

   c. Restore the previous values of **Minimum instance count** and **Maximum instance count**, and then choose **Apply**.