Tutorial: Connect a WordPress website in Amazon Lightsail to an Amazon S3 bucket

Last updated: July 30, 2019

This tutorial describes the steps required to connect your WordPress website running on an Amazon Lightsail instance to an Amazon Simple Storage Service (Amazon S3) bucket to store website images and attachments. To do this, you configure a WordPress plugin with a set of Amazon Web Services (AWS) account credentials. The plugin then creates the Amazon S3 bucket for you and configures your website to use the bucket instead of the instance's disk for website images and attachments.

Contents

Step 1: Complete the prerequisites

Before you get started, create a WordPress instance in Lightsail, and make sure it's in a running state. For more information, see Tutorial: Launch and configure a WordPress instance in Amazon Lightsail.

Step 2: Install the WP Offload Media plugin on your WordPress website

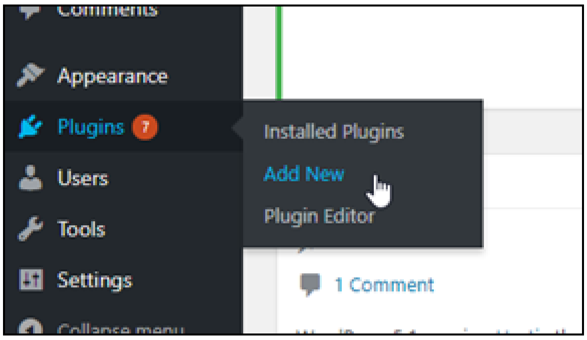
You must use a plugin to configure your website to use an Amazon S3 bucket. Many plugins are available to configure this; one such plugin is WP Offload Media Lite.

Complete the following steps to install the WP Offload Media plugin on your WordPress website:

1. Sign in to your WordPress dashboard as an administrator.

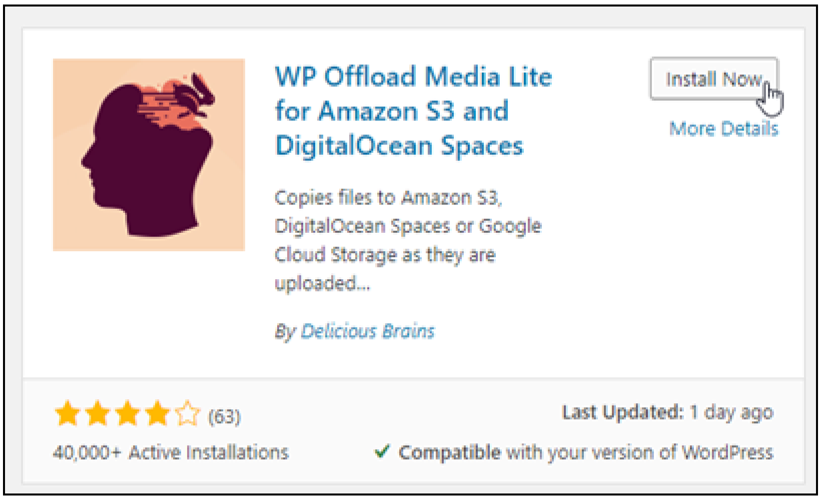
   For more information, see Getting the application user name and password for your 'Powered by Bitnami' instance in Amazon Lightsail.

2. Hover over **Plugins** in the left navigation menu, and choose **Add New**.



3. Search for **WP Offload Media Lite**.

4. In the search results, choose **Install Now** next to the **WP Offload Media** plugin.



WP Offload Media Lite for Amazon S3 and DigitalOcean Spaces

Copies files to Amazon S3, DigitalOcean Spaces or Google Cloud Storage as they are uploaded...

By Delicious Brains

(63)

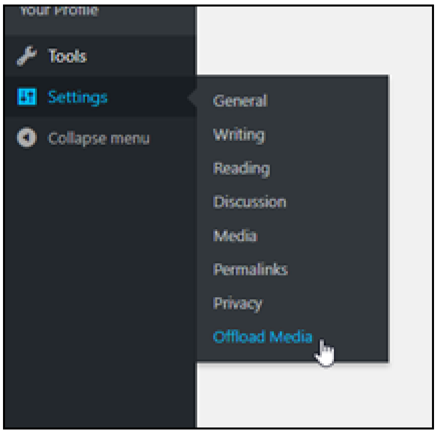
40,000+ Active Installations

Last Updated: 1 day ago

✓ **Compatible** with your version of WordPress

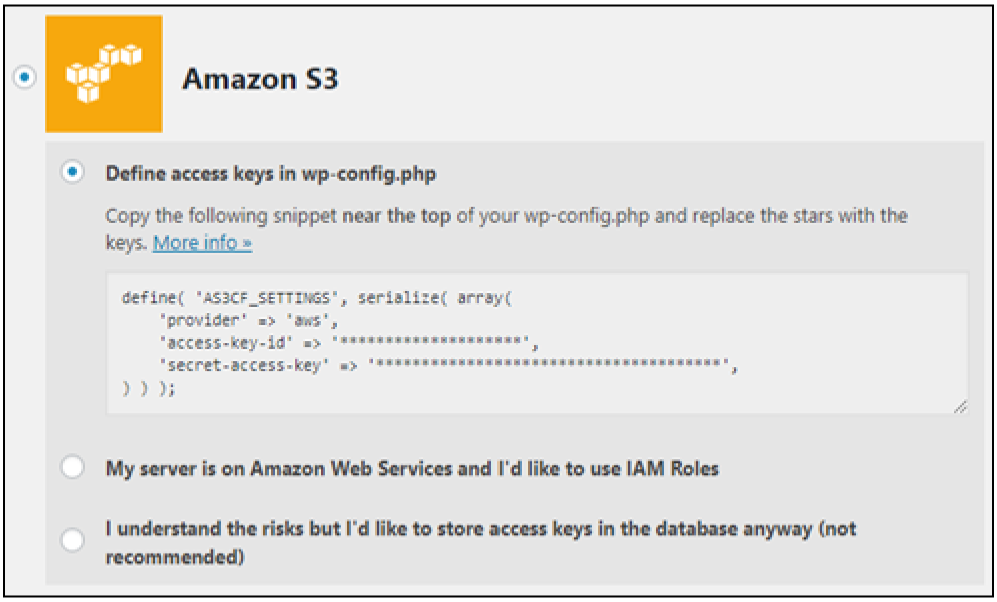
5. Choose **Activate** after the plugin is done installing.

6. In the left navigation menu, choose **Settings**, then choose **Offload Media**.



7. In the **Offload Media** page, choose **Amazon S3** as the storage provider, then choose **Define access keys in wp-config.php**.

   With this option, you must add your AWS account credentials to the `wp-config.php` on the instance. These steps are covered later in this tutorial.



● Amazon S3

● **Define access keys in wp-config.php**

Copy the following snippet near the top of your wp-config.php and replace the stars with the keys. More info »

```
define( 'AS3CF_SETTINGS', serialize( array(
    'provider' => 'aws',
    'access-key-id' => '********************',
    'secret-access-key' => '****************************************',
) ) );
```

○ **My server is on Amazon Web Services and I'd like to use IAM Roles**

○ **I understand the risks but I'd like to store access keys in the database anyway (not recommended)**

Leave the **Offload Media** page open; you will return to it later in this tutorial. Continue to the Step 3: Create an IAM user and policy section of this tutorial.

Step 3: Create an IAM user and policy

The WP Offload Media plugin requires access to your AWS account to create the Amazon S3 bucket, and to upload your website images and attachments.

Complete the following steps to create a new AWS Identity and Access Management (IAM) user and policy for the WP Offload Media plugin:

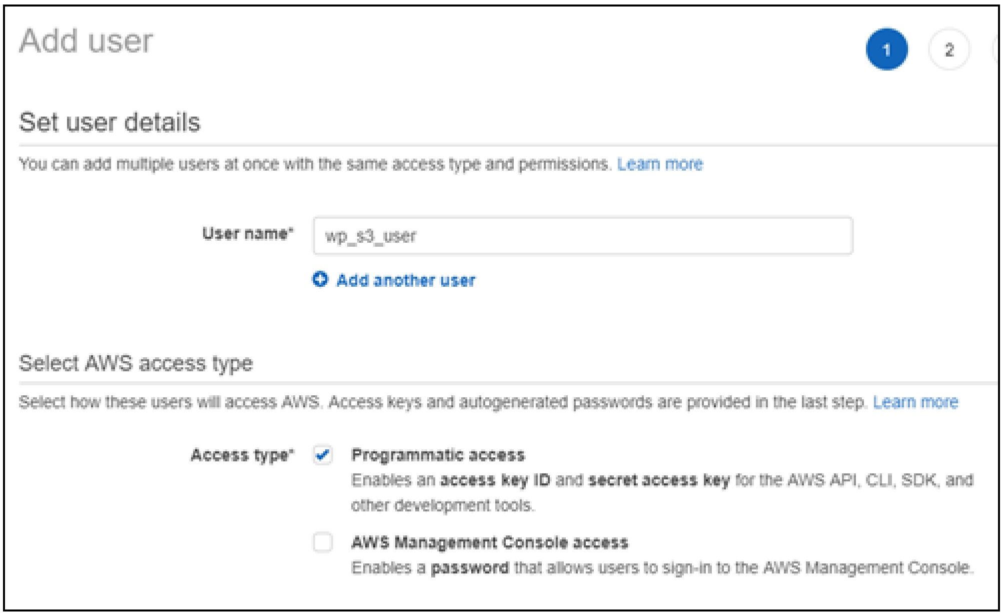
1. Open a new browser tab, and sign in to the IAM console.

2. In the left navigation menu, choose **Users**.

3. Choose **Add user**.

4. In the **User name** text box, enter a name for the new user. Enter something descriptive, such as `wp_s3_user` or `wp_offload_media_plugin_user`, so that you can easily identify it in the future when performing maintenance.

5. Under the **Access type** section, choose **Programmatic access**.



Add user

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* wp_s3_user

Add another user

Select AWS access type

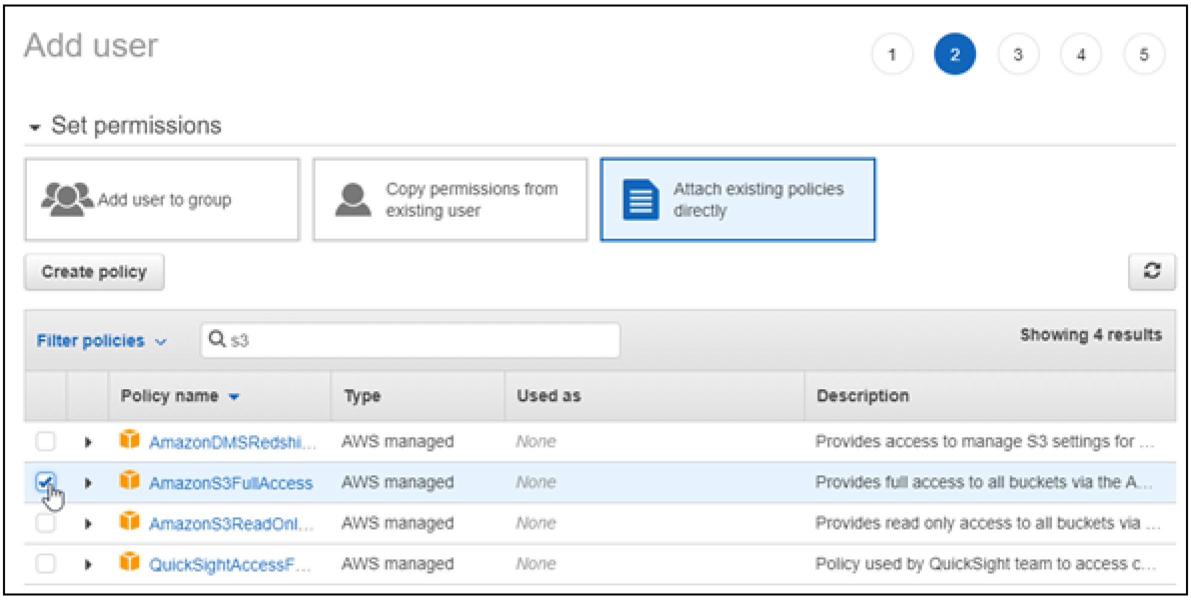
Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type* ☑ Programmatic access
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

□ AWS Management Console access
Enables a **password** that allows users to sign-in to the AWS Management Console.

6. Choose **Next: Permissions**.

7. Choose **Attach existing policies directly**, search for **S3**, and then choose **AmazonS3FullAccess** in the search results.



		Policy name	Type	Used as	Description
□	▸	AmazonDMSRedshi...	AWS managed	None	Provides access to manage S3 settings for ...
☑	▸	AmazonS3FullAccess	AWS managed	None	Provides full access to all buckets via the A...
□	▸	AmazonS3ReadOnl...	AWS managed	None	Provides read only access to all buckets via ...
□	▸	QuickSightAccessF...	AWS managed	None	Policy used by QuickSight team to access c...

8. Choose **Next: Tags**, and then choose **Next: Review**.

9. Review the user details displayed on the page, then choose **Create user**.

10. Make note of the **access key ID** and **secret access key** for the user, or choose **Download .csv** to save a copy of these values to your local drive. You will need these in the next few steps when editing the `wp-config.php` file on the WordPress instance.

Step 4: Edit the WordPress configuration file

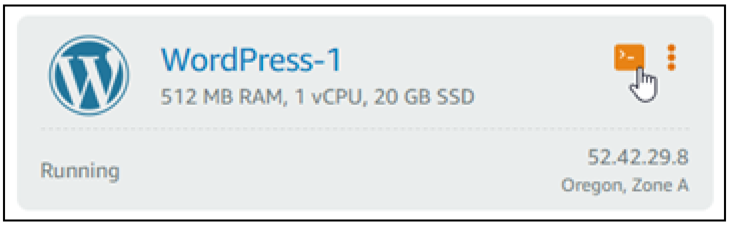
Complete the following steps to connect to your WordPress instance using the browser-based SSH client in the Lightsail console and edit the `wp-config.php` file.

The `wp-config.php` file contains your website’s base configuration details, such as database connection information.

Note
You can also connect to your instance using your own SSH client. For more information, see Download and set up PuTTY to connect using SSH in Amazon Lightsail

1. Sign in to the Lightsail console.

2. Choose the browser-based SSH client icon for the WordPress instance.



WordPress-1
512 MB RAM, 1 vCPU, 20 GB SSD
Running
52.42.29.8
Oregon, Zone A

3. In the SSH client window that appears, enter the following command to create a backup of the `wp-config.php` file in case something goes wrong:

```
sudo cp /opt/bitnami/apps/wordpress/htdocs/wp-config.php
/opt/bitnami/apps/wordpress/htdocs/wp-config.php-backup
```

4. Enter the following command to open the `wp-config.php` file using `nano`, a text editor:

   ```
   nano /opt/bitnami/apps/wordpress/htdocs/wp-config.php
   ```

5. Enter the following text above the `/* That's all, stop editing! Happy blogging. */` text.

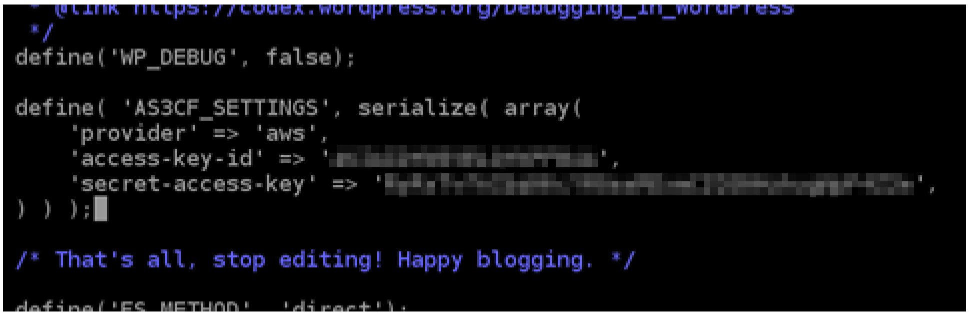
   Be sure to replace *AccessKeyID* with the access key ID and *SecretAccessKey* with the secret access key of the IAM user you created earlier in these steps.

   ```
   define( 'AS3CF_SETTINGS', serialize( array(
       'provider' => 'aws',
       'access-key-id' => 'AccessKeyID',
       'secret-access-key' => 'SecretAccessKey',
   ) ) );
   ```

   Example:

   ```
   define( 'AS3CF_SETTINGS', serialize( array(
       'provider' => 'aws',
       'access-key-id' => 'AKIAIOSFODNN7EXAMPLE',
       'secret-access-key' => 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY',
   ) ) );
   ```

   The result should look like the following example:

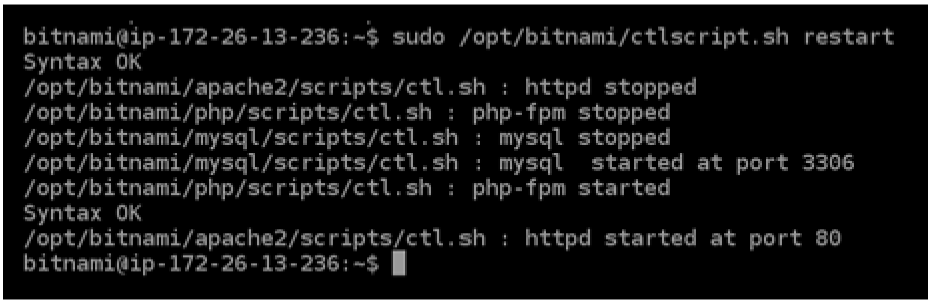
   ```
    * @link https://codex.wordpress.org/Debugging_in_WordPress
    */
   define('WP_DEBUG', false);

   define( 'AS3CF_SETTINGS', serialize( array(
       'provider' => 'aws',
       'access-key-id' => '[redacted]',
       'secret-access-key' => '[redacted]',
   ) ) );

   /* That's all, stop editing! Happy blogging. */

   define('FS_METHOD', 'direct');
   ```

6. Press **Ctrl+X** to exit Nano, then press **Y**, and **Enter** to save your edits to the `wp-config.php` file.

7. Enter the following command to restart the services on the instance:

   ```
   sudo /opt/bitnami/ctlscript.sh restart
   ```

You will see a result similar to the following when the services have restarted:



```
bitnami@ip-172-26-13-236:~$ sudo /opt/bitnami/ctlscript.sh restart
Syntax OK
/opt/bitnami/apache2/scripts/ctl.sh : httpd stopped
/opt/bitnami/php/scripts/ctl.sh : php-fpm stopped
/opt/bitnami/mysql/scripts/ctl.sh : mysql stopped
/opt/bitnami/mysql/scripts/ctl.sh : mysql  started at port 3306
/opt/bitnami/php/scripts/ctl.sh : php-fpm started
Syntax OK
/opt/bitnami/apache2/scripts/ctl.sh : httpd started at port 80
bitnami@ip-172-26-13-236:~$ █
```

Close the SSH window and toggle back to the **Offload Media** page that you left open earlier in this tutorial. You are now ready to create the Amazon S3 bucket using the WP Offload Media plugin.

Step 5: Create the Amazon S3 bucket using the WP Offload Media plugin

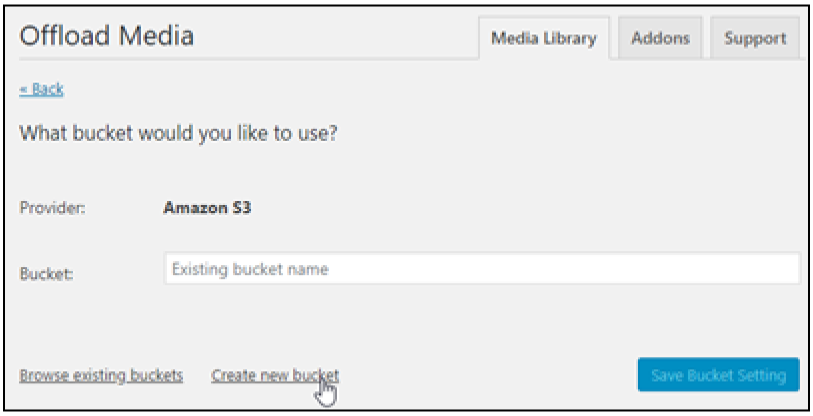
Now that the `wp-config.php` file is configured with the AWS credentials, you can return to the **Offload Media** page to complete the process.

Complete the following steps to create the Amazon S3 bucket using the WP Offload Media plugin.

1. Refresh the **Offload Media** page, or choose **Next**.

   You should now see that the Amazon S3 provider is configured.

2. Choose **Create new bucket**.



Offload Media

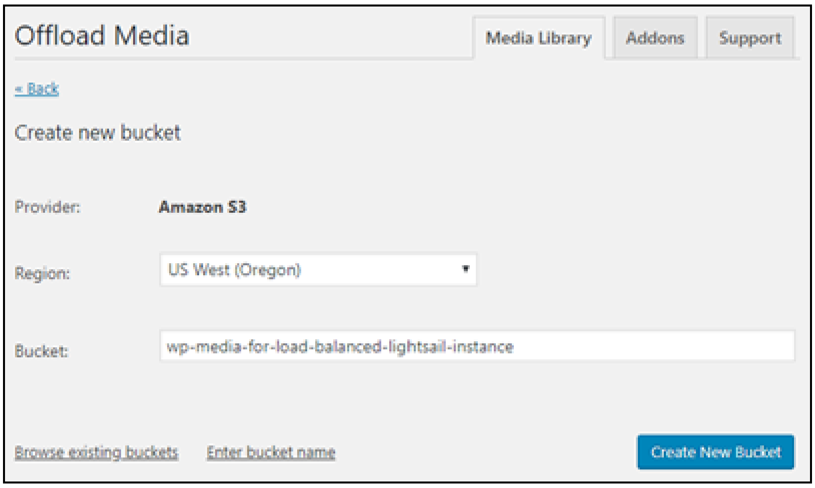
What bucket would you like to use?

Provider: Amazon S3

Bucket: Existing bucket name

3. In the **Region** drop-down menu, choose the desired AWS Region. We recommend that you choose the same region in which your WordPress instance is located.

4. In the **Bucket** text box, enter a name for the new S3 bucket.



Offload Media

Create new bucket

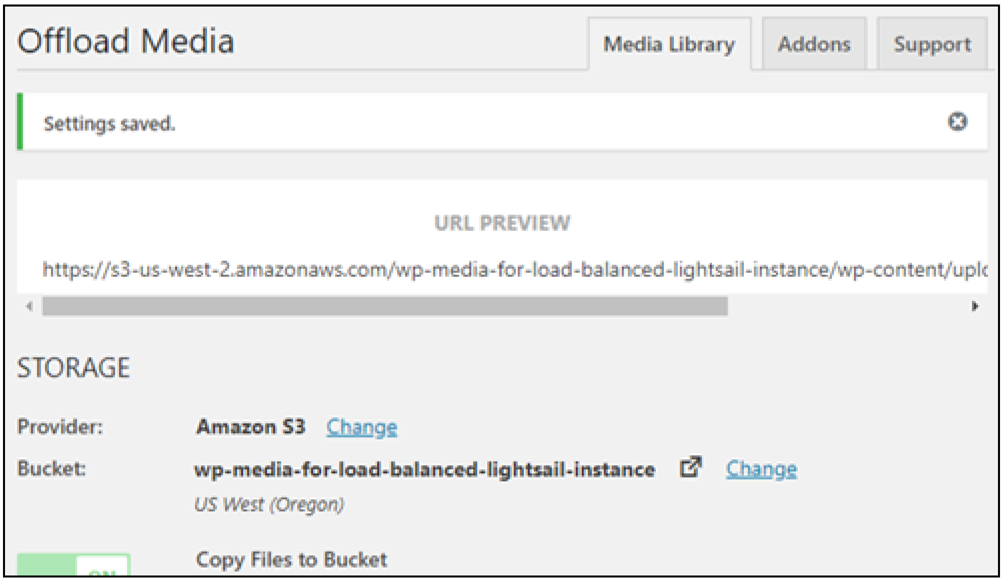
Provider: Amazon S3

Region: US West (Oregon)

Bucket: wp-media-for-load-balanced-lightsail-instance

5. Choose **Create New Bucket**.

   The page refreshes to confirm that a new bucket was created. Review the settings that appear and adjust them accordingly to how you want your WordPress website to behave.



Settings saved.

URL PREVIEW

https://s3-us-west-2.amazonaws.com/wp-media-for-load-balanced-lightsail-instance/wp-content/uplc

STORAGE

Provider: Amazon S3 Change

Bucket: wp-media-for-load-balanced-lightsail-instance Change
US West (Oregon)

Copy Files to Bucket

   From now on, images and attachments added to blog posts are automatically uploaded to the Amazon S3 bucket that you created.

Step 6: Next steps

After you're done connecting your WordPress website to an Amazon S3 bucket, you should create a snapshot of your WordPress instance to back up the changes you made. For more information, see Creating a snapshot of your Linux or Unix instance in Amazon Lightsail.