


How to set up WordPress on Amazon Lightsail

📅 Deadline	
☑ Done?	<input type="checkbox"/>
🔗 Project	 Technical Documentation
🔍 Project Start Date	April 4, 2020

Introduction

I installed WordPress on Amazon Lightsail multiple times so you don't have to. I created this guide after hours of reading and troubleshooting to figure out a more streamlined installation. I'm not a programmer and you don't have to be one either to follow this guide. But you do need to have the pre-requisites.

This is a complete guide to installing WordPress on Amazon Lightsail, with the following criteria:

- Linux/Unix-based instance
- Domain registered at Namecheap

You can follow these instructions if you have your domains registered at another registrar but keep in mind that domain instructions will be targeted towards Namecheap.

Pre-requisites

- An AWS account
- A domain registered at Namecheap
- Some technical skills like the ability to troubleshoot if you need to
- Some skills in using a command line interface (or not afraid to learn)
- ~1 hour barring any issues

Warnings

- This is not a one-click solution. If you're looking for something quick and easy where you don't have to deal with all the back-end stuff, try a managed WordPress hosting service.
- Understand the limitations:
 - Since you're linking your Namecheap domain to Amazon, there's no DNNSEC support.
 - DNNSEC are a set of security rules that protects against counterfeit DNS data by verifying digital signatures. You can get into the weeds at this [Namecheap article here](#).
 - You'll only get one month free of Lightsail opposed to AWS EC2's one year free.
 - There might be other technical limitations I'm not aware of yet.
- Understand the extra maintenance work you might have to do:

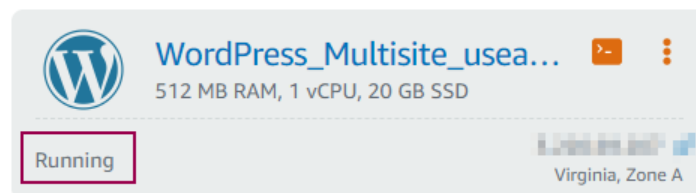
- Wordpress updates, Bitnami stuff, maybe security patches
- Troubleshooting if something goes wrong
- Lightsail doesn't come with email service, you'll have to set it up with AWS SES or buy it somewhere else.
- You'll see the term **instance** used. An instance is your virtual private server (VPS) that lives in the AWS cloud. This is where your operating system will be installed. Then the Bitnami WordPress application is installed in the operating system. You'll learn more about this as you go through the instructions.
- If after all these warnings, you still want to proceed...continue below.

[LINK TO CONTINUE OR EXPAND INSTRUCTIONS]

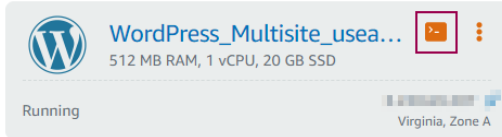
Table of Contents

Create a Linux/Unix-based Instance on Lightsail

1. On the Lightsail home page, click on **Create Instance**
2. Select a region for your instance location. The closer your instance is to your main audience, the less delay they'll experience when accessing your site. If you don't know which one to pick, choose **Virginia us-east-1a**.
3. Select the following platform: **Linux/Unix**
4. Select the following blueprint:
 - Apps + OS
 - WordPress or ~~WordPress Multisite~~
 - Choose WordPress. Only choose Multisite if you understand the differences.
5. Skip the **Optional** stuff for now
6. Choose your instance plan
7. Identify your instance with a unique name. I named mine with this format: **WP-Multisite-useast1a**
8. Click on **Create Instance** and wait a few minutes for it to be ready. When ready, your instance module will display **Running**.



1. Once your instance is running, do a quick check to make sure it's working:
 - Click on the orange terminal icon, a window should pop up and it should look like this.



```
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1102-aws x86_64)
*** System restart required ***

  bitnami

*** Welcome to the Bitnami WordPress Multisite 5.3.2-3 ***
*** Documentation: https://docs.bitnami.com/aws/apps/wordpress-multisite/ ***
*** Bitnami Forums: https://community.bitnami.com/ ***

#####
### For frequently used commands, please run: ###
### sudo /opt/bitnami/bnhelper-tool ###
#####
```

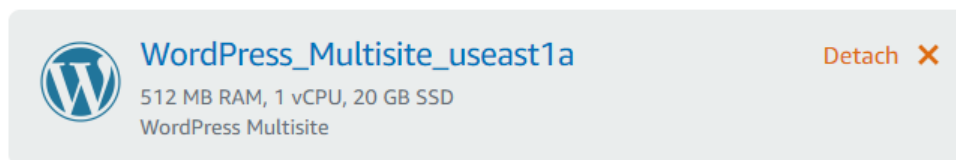
- This means you've connected successfully to your instance.
- Close the terminal, we're going to create a static IP now.

Create a Static IP

1. On the Lightsail Home page, click **Networking**
2. Click **Create static IP**
3. Select the same region that your instance is in. Remember, the region is the location you picked when creating your instance.
4. Attach the IP to your instance. If you don't attach it, nothing we're going to do will work and you'll be charged for the IP. It's free when connected to an instance.

Attach to an instance

Attaching a static IP replaces that instance's dynamic IP address.



5. Enter a name for your static IP. I named mine **StaticIP-WP-Multisite-useast1a**.
6. Click **Create** to create your IP.
7. Next, we're going to point a custom domain to your WordPress site.

Point a Custom Domain to Your WordPress Site

- In **Networking**, select **Create DNS zone**

Create DNS zone

- Enter your domain name
- Add tags if you need to, if not then skip the tags section
- Click on **Create DNS Zone**
- You'll be directed to a DNS records page with a **list of name servers**. Take note of the name servers, you'll need them later.
- On the same page, choose **Add Record**

+ Add record

- Add these two records:
 - **A record** - This associates your domain or subdomain with an IP address
 - **Subdomain:** @
 - This is the apex of your domain i.e. cactusbright.com opposed to www.cactusbright.com
 - **Resolves to:** <choose your static ip from the drop down>

A record — Associate your domain or a subdomain with an IP address.

Subdomain: @ .cactusbright.com

Resolves to: p.0.0.0

STATIC IP ADDRESSES

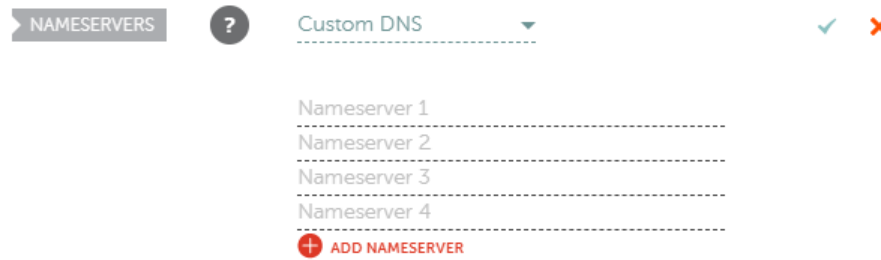
Staticip-WP-Multisite-useas...

! Must be a valid IP address

- **CNAME record** - This maps an alias or subdomain
 - **Subdomain:** www
 - **Maps to:** <yourdomain.com>
- **TXT record (Optional)** - If you don't need this right now, you can skip it
- Add any other records you may need. If not, let's move on.

Add Name servers to your domain in Namecheap

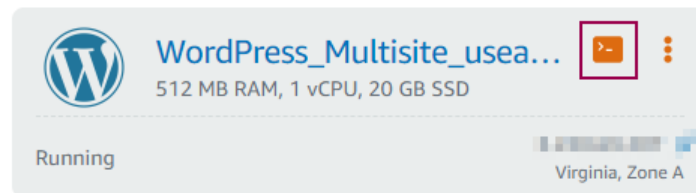
- Sign into Namecheap and navigate to your dashboard.
- Click **Manage** on your selected domain
- In the **Nameservers** category, choose **Custom DNS**



- Enter the Lightsail nameservers that you got from step # and make sure to **click the green check mark** to save your changes.
- The name server change will propagate through the internet's DNS now. It can take from a couple minutes to several hours. In my experience, it only takes a few minutes. Once completed, you should be able to navigate to your domain.

Generate an SSL Certificate (https)

- Go back to Lightsail Home
- Click on the orange terminal icon to connect to your instance. If you forgot where that is, see this screenshot.



- When the terminal has launched, type in the following command and follow the prompts: `sudo /opt/bitnami/bncert-tool`
- When it prompts you for your domain list, type in the address of the apex of your domain and the www subdomain. See this screenshot for an example.

```
-----
Domains

Please provide a valid space-separated list of domains for which you wish to
configure your web server.

Domain list []: cactusbright.com www.cactusbright.com
```

- Once that's complete, keep your terminal open so we can force your site to use https.

Force your site to use https (only need to do this for multi-sites)

- While in the terminal, type in the following command: `sudo nano /opt/bitnami/apache2/conf/bitnami/bitnami.conf`
 - This command opens the bitnami configuration file in a built-in text editor called nano

- Scroll down using your keyboard arrow key until you find **<VirtualHost_default_:80>**
- Under DocumentRoot.....add the below code replacing YOURDOMAIN to your domain:

```
RewriteEngine On
RewriteCond %{HTTPS} !=on
RewriteCond %{HTTP_HOST} !^(localhost|127.0.0.1)
RewriteRule ^/(.*)$ https://YOURDOMAIN.com/$1 [R=permanent,L]
```

- Make sure the code is inside the **</VirtualHost>** tag. The completed code block should look like this:

```
<VirtualHost _default_:80>
DocumentRoot "/opt/bitnami/apache2/htdocs"
RewriteEngine On
RewriteCond %{HTTPS} !=on
RewriteCond %{HTTP_HOST} !^(localhost|127.0.0.1)
RewriteRule ^/(.*)$ https://YOURDOMAIN.com/$1 [R=permanent,L]
...
</VirtualHost>
```

- Now we're going to copy and paste the code you just inserted in another area of the same document.
- You can either type out the code again or copy and paste by doing the following:
 - Position the cursor at the beginning of the character that you want to copy.
 - Press and release **Alt + Shift + A** to set the mark.
 - Then use arrow keys to highlight the text to copy.
 - Once highlighted, press and release **Alt + Shift + 6** to copy.
 - Scroll down using your keyboard arrow until you find **<VirtualHost_default_:443>**
 - Under DocumentRoot.....press **Ctrl + U** to paste.
- Add the extra rule highlighted below to your code. The rest of the code should stay the same.

```
<VirtualHost_default_:443>
DocumentRoot.....
RewriteEngine On
RewriteCond %{HTTP_HOST} !^cactusbright.com$
RewriteCond %{HTTPS} !=on
RewriteCond %{HTTP_HOST} !^(localhost|127.0.0.1)
RewriteRule ^/(.*)$ https://cactusbright.com/$1 [R=permanent,L]
...
</VirtualHost>
```

- Then press **Ctrl + X** to exit, **Y** to save, **Enter** to overwrite the file.
- Now enter the following command to restart the apache server: `sudo /opt/bitnami/ctlscript.sh restart apache`
- Once the restart is complete, test to see if it works by going to your website. It should redirect you to https.

We're almost done

Are you tired of following these instructions yet? Cause I'm tired of writing them #regrets. But! We're almost done. Let's access your WordPress admin panel now.

- While in your terminal, run the following command to get your credentials: `cat ./bitnami_credentials`

- Note the username and password
- Log into your admin console by navigating to yourdomain.com/wp-login
- Success! Or should be. Now you can edit the super admin account, create new accounts, and start creating your website.

But wait! What about that annoying Bitnami banner

- This one?



- Okay, let's get rid of that.
- In your terminal, run this command to disable the banner: `sudo /opt/bitnami/apps/wordpress/bnconfig --disable_banner`
 1
 - if that doesn't work than do this:
`sudo touch /opt/bitnami/apps/bitnami/banner/disable-banner`
- Then run this command to delete the bitnami page: `sudo rm -rf /opt/bitnami/apps/bitnami/banner/htdocs`
- Finally, restart the apache web server by running this command: `sudo /opt/bitnami/ctlscript.sh restart apache`

Congratulations, you're officially done!

- During this guide, you created an instance on Lightsail to host your WordPress site and pointed it to a custom domain. Than you generated SSL certificates and forced https so that your site can be secure. Lastly, you got rid of the Bitnami banner and now you're ready to create a website.
- If you notice any errors in this documentation, please kindly let me know.

Sources:

Amazon Lightsail and Bitnami has great documentation. These are all the resources I used to put together this guide.

- https://lightsail.aws.amazon.com/ls/docs/en_us/articles/getting-started-with-amazon-lightsail
- https://lightsail.aws.amazon.com/ls/docs/en_us/articles/lightsail-create-static-ip
- https://lightsail.aws.amazon.com/ls/docs/en_us/articles/lightsail-how-to-create-dns-entry
- <https://docs.bitnami.com/aws/apps/wordpress/administration/generate-configure-certificate-letsencrypt/>
- <https://docs.bitnami.com/bch/apps/wordpress-multisite/administration/force-https-apache/>
- <https://www.youtube.com/watch?v=amiutv8BEw>
- <https://askubuntu.com/questions/833102/copy-only-copy-not-cutting-in-nano>
- <https://docs.bitnami.com/aws/faq/get-started/find-credentials/>

- <https://docs.bitnami.com/aws/faq/get-started/access-phpmyadmin/>
- <https://docs.bitnami.com/aws/apps/wordpress-multisite/>
- <https://stackoverflow.com/questions/34744552/how-can-i-remove-bitnami-banner-on-wordpress-site>
- <https://community.bitnami.com/t/cannot-remove-the-bitnami-info-banner-for-a-wordpress-multisite/29988/8>
- <https://docs.bitnami.com/google/how-to/bitnami-remove-banner/>