



Tutorial: Installing a LAMP Web Server on Amazon Linux

September 29, 2016

The following procedures help you install the Apache web server with PHP and MySQL support on your Amazon Linux instance (sometimes called a LAMP web server or LAMP stack). You can use this server to host a static website or deploy a dynamic PHP application that reads and writes information to a database.

Prerequisites

This tutorial assumes that you have already launched an instance with a public DNS name that is reachable from the Internet. For more information, see [Step 1: Launch an Instance](#). You must also have configured your security group to allow `SSH` (port 22), `HTTP` (port 80), and `HTTPS` (port 443) connections. For more information about these prerequisites, see [Setting Up with Amazon EC2](#).

Important

If you are trying to set up a LAMP web server on an Ubuntu instance, this tutorial will not work for you. These procedures are intended for use with Amazon Linux. For more information about other distributions, see their specific documentation. For information about LAMP web servers on Ubuntu, see the Ubuntu community documentation [ApacheMySQLPHP](#) topic.

To install and start the LAMP web server on Amazon Linux

1. [Connect to your instance](#).
2. To ensure that all of your software packages are up to date, perform a quick software update on your instance. This process may take a few minutes, but it is important to make sure you have the latest security updates and bug fixes.

Note

The `-y` option installs the updates without asking for confirmation. If you would like to examine the updates before installing, you can omit this option.

```
[ec2-user ~]$ sudo yum update -y
```

3. Now that your instance is current, you can install the Apache web server, MySQL, and PHP software packages. Use the **yum install** command to install multiple software packages and all related dependencies at the same time.

```
[ec2-user ~]$ sudo yum install -y httpd24 php56 mysql55-server  
php56-mysqlnd
```

4. Start the Apache web server.

```
[ec2-user ~]$ sudo service httpd start  
Starting httpd: [ OK ]
```

5. Use the **chkconfig** command to configure the Apache web server to start at each system boot.

```
[ec2-user ~]$ sudo chkconfig httpd on
```

Tip

The **chkconfig** command does not provide any confirmation message when you successfully enable a service. You can verify that **httpd** is on by running the following command.

```
[ec2-user ~]$ chkconfig --list httpd  
httpd          0:off  1:off  2:on   3:on   4:on   5:on  
6:off
```

Here, **httpd** is on in runlevels 2, 3, 4, and 5 (which is what you want to see).

6. Test your web server. In a web browser, enter the public DNS address (or the public IP address) of your instance; you should see the Apache test page. You can get the public DNS for your instance using the Amazon EC2 console (check the Public DNS column; if this column is hidden, choose Show/Hide and select Public DNS).

Tip

If you are unable to see the Apache test page, check that the security group you are using contains a rule to allow HTTP (port 80) traffic. For information about adding an HTTP rule to your security group, see [Adding Rules to a Security Group](#).

Important

If you are not using Amazon Linux, you may also need to configure the firewall on your instance to allow these connections. For more information about how to configure the firewall, see the documentation for your specific distribution.

Amazon Linux AMI Test Page

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly, but has not yet been configured.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting `www.example.com`, you should send e-mail to "webmaster@example.com".

The [Amazon Linux AMI](#) is a supported and maintained Linux image provided by [Amazon Web Services](#) for use on [Amazon Elastic Compute Cloud \(Amazon EC2\)](#). It is designed to provide a stable, secure, and high performance execution environment for applications running on [Amazon EC2](#). It also includes packages that enable easy integration with [AWS](#), including launch configuration tools and many popular AWS libraries and tools. [Amazon Web Services](#) provides ongoing security and maintenance updates to all instances running the [Amazon Linux AMI](#). The [Amazon Linux AMI](#) is provided at no additional charge to [Amazon EC2 users](#).

If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the images below on Apache and Amazon Linux AMI powered HTTP servers. Thanks for using Apache and the Amazon Linux AMI!



Note

This test page appears only when there is no content in `/var/www/html`. When you add content to the document root, your content appears at the public DNS address of your instance instead of this test page.

Apache **httpd** serves files that are kept in a directory called the Apache document root. The Amazon Linux Apache document root is `/var/www/html`, which is owned by `root` by default.

```
[ec2-user ~]$ ls -l /var/www
total 16
drwxr-xr-x 2 root root 4096 Jul 12 01:00 cgi-bin
drwxr-xr-x 3 root root 4096 Aug  7 00:02 error
drwxr-xr-x 2 root root 4096 Jan  6 2012 html
drwxr-xr-x 3 root root 4096 Aug  7 00:02 icons
```

To allow `ec2-user` to manipulate files in this directory, you need to modify the ownership and permissions of the directory. There are many ways to accomplish this task; in this tutorial, you add a `www` group to your instance, and you give that group ownership of the `/var/www` directory and add write permissions for the group. Any members of that group will then be able to add, delete, and modify files for the web server.

To set file permissions

1. Add the `www` group to your instance.

```
[ec2-user ~]$ sudo groupadd www
```

2. Add your user (in this case, `ec2-user`) to the `www` group.

```
[ec2-user ~]$ sudo usermod -a -G www ec2-user
```

Important

You need to log out and log back in to pick up the new group. You can use the **exit** command, or close the terminal window.

3. Log out and then log back in again, and verify your membership in the `www` group.
 - a. Log out.

```
[ec2-user ~]$ exit
```

- b. Reconnect to your instance, and then run the following command to verify your membership in the `www` group.

```
[ec2-user ~]$ groups  
ec2-user wheel www
```

4. Change the group ownership of `/var/www` and its contents to the `www` group.

```
[ec2-user ~]$ sudo chown -R root:www /var/www
```

5. Change the directory permissions of `/var/www` and its subdirectories to add group write permissions and to set the group ID on future subdirectories.

```
[ec2-user ~]$ sudo chmod 2775 /var/www  
[ec2-user ~]$ find /var/www -type d -exec sudo chmod 2775 {} \;
```

6. Recursively change the file permissions of `/var/www` and its subdirectories to add group write permissions.

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
```

Now `ec2-user` (and any future members of the `www` group) can add, delete, and edit files in the Apache document root. Now you are ready to add content, such as a static website or a PHP application.

(Optional) Secure your web server

A web server running the HTTP protocol provides no transport security for the data that it sends or receives. When you connect to an HTTP server using a web browser, the URLs that you enter, the content of web pages that you receive, and the contents (including passwords) of any HTML forms that you submit are all visible to eavesdroppers anywhere along the network pathway. The best practice for securing your web server is to install support for HTTPS (HTTP Secure), which protects your data with SSL/TLS encryption.

For information about enabling HTTPS on your server, see [Tutorial: Configure Apache Web Server on Amazon Linux to use SSL/TLS](#).

To test your LAMP web server

If your server is installed and running, and your file permissions are set correctly, your `ec2-user` account should be able to create a simple PHP file in the `/var/www/html` directory that will be available from the Internet.

1. Create a simple PHP file in the Apache document root.

```
[ec2-user ~]$ echo "<?php phpinfo(); ?>" >  
/var/www/html/phpinfo.php
```


Tip

If you get a "Permission denied" error when trying to run this command, try logging out and logging back in again to pick up the proper group permissions that you configured in [To set file permissions](#).

2. In a web browser, enter the URL of the file you just created. This URL is the public DNS address of your instance followed by a forward slash and the file name. For example:

```
http://my.public.dns.amazonaws.com/phpinfo.php
```

You should see the PHP information page:

PHP Version 5.6.6	
	
System	Linux ip-172-31-7-35 3.14.35-28.38.amzn1.x86_64 #1 SMP Wed Mar 11 22:50:37 UTC 2015 x86_64
Build Date	Mar 5 2015 23:26:53
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php-5.6.d
Additional .ini files parsed	/etc/php-5.6.d/20-bz2.ini, /etc/php-5.6.d/20-calendar.ini, /etc/php-5.6.d/20-ctype.ini, /etc/php-5.6.d/20-curl.ini, /etc/php-5.6.d/20-dom.ini, /etc/php-5.6.d/20-exif.ini, /etc/php-5.6.d/20-fileinfo.ini, /etc/php-5.6.d/20-ftp.ini, /etc/php-5.6.d/20-gettext.ini, /etc/php-5.6.d/20-iconv.ini, /etc/php-5.6.d/20-mysqlnd.ini, /etc/php-5.6.d/20-pdo.ini, /etc/php-5.6.d/20-phar.ini, /etc/php-5.6.d/20-posix.ini, /etc/php-5.6.d/20-shmop.ini, /etc/php-5.6.d/20-simplexml.ini, /etc/php-5.6.d/20-sockets.ini, /etc/php-5.6.d/20-sqlite3.ini, /etc/php-5.6.d/20-sysvmsg.ini, /etc/php-5.6.d/20-sysvsem.ini, /etc/php-5.6.d/20-sysvshm.ini, /etc/php-5.6.d/20-tokenizer.ini, /etc/php-5.6.d/20-xml.ini, /etc/php-5.6.d/20-xmlwriter.ini, /etc/php-5.6.d/20-xsl.ini, /etc/php-5.6.d/20-zip.ini, /etc/php-5.6.d/30-mysql.ini, /etc/php-5.6.d/30-mysqli.ini, /etc/php-5.6.d/30-pdo_mysql.ini, /etc/php-5.6.d/30-pdo_sqlite.ini, /etc/php-5.6.d/30-xmlreader.ini, /etc/php-5.6.d/40-json.ini, /etc/php-5.6.d/php.ini
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API20131226,NTS
PHP Extension Build	API20131226,NTS

Note

If you do not see this page, verify that the `/var/www/html/phpinfo.php` file was created properly in the previous step. You can also verify that all of the required packages were installed with the following command (the package versions in the second column do not need to match this example output):

```
[ec2-user ~]$ sudo yum list installed httpd24 php56 mysql55-server
php56-mysqld
Loaded plugins: priorities, update-motd, upgrade-helper
959 packages excluded due to repository priority protections
Installed Packages
httpd24.x86_64                2.4.16-1.62.amzn1
                               @amzn-main
mysql55-server.x86_64        5.5.45-1.9.amzn1
                               @amzn-main
php56.x86_64                 5.6.13-1.118.amzn1
                               @amzn-main
php56-mysqld.x86_64          5.6.13-1.118.amzn1
                               @amzn-main
```

If any of the required packages are not listed in your output, install them with the **`sudo yum install package`** command.

To secure the MySQL server

The default installation of the MySQL server has several features that are great for testing and development, but they should be disabled or removed for production servers. The **`mysql_secure_installation`** command walks you through the process of setting a root password and removing the insecure features from your installation. Even if you are not planning on using the MySQL server, performing this procedure is a good idea.

1. Start the MySQL server.

```
[ec2-user ~]$ sudo service mysqld start
Initializing MySQL database: Installing MySQL system tables...
```


Tutorial: Installing a LAMP Web Server on Amazon Linux

OK

Filling help tables...

OK

To start mysqld at boot time you have to copy
support-files/mysql.server to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !

...

Starting mysqld: [OK]

2. Run `mysql_secure_installation`.

```
[ec2-user ~]$ sudo mysql_secure_installation
```

- a. When prompted, enter a password for the `root` account.
 - i. Enter the current `root` password. By default, the `root` account does not have a password set, so press **Enter**.
 - ii. Type **Y** to set a password, and enter a secure password twice. For more information about creating a secure password, see <http://www.pctools.com/guides/password/>. Make sure to store this password in a safe place.

Note

Setting a root password for MySQL is only the most basic measure for securing your database. When you build or install a database-driven application, you typically create a database service user for that application and avoid using the root account for anything but database administration.

- b. Type **Y** to remove the anonymous user accounts.
- c. Type **Y** to disable remote `root` login.
- d. Type **Y** to remove the test database.
- e. Type **Y** to reload the privilege tables and save your changes.

3. (Optional) Stop the MySQL server if you do not plan to use it right away. You can restart the server when you need it again.

```
[ec2-user ~]$ sudo service mysqld stop
```

```
Stopping mysqld: [ OK ]
```

4. (Optional) If you want the MySQL server to start at every boot, enter the following command.

```
[ec2-user ~]$ sudo chkconfig mysqld on
```

You should now have a fully functional LAMP web server. If you add content to the Apache document root at `/var/www/html`, you should be able to view that content at the public DNS address for your instance.

(Optional) Install phpMyAdmin

[phpMyAdmin](#) is a web-based database management tool that you can use to view and edit the MySQL databases on your EC2 instance. Follow the steps below to install and configure phpMyAdmin on your Amazon Linux instance.

Important

We do not recommend using phpMyAdmin to access a LAMP server unless you have enabled SSL/TLS in Apache; otherwise, your database administrator password and other data will be transmitted insecurely across the Internet. For information about configuring a secure web server on an EC2 instance, see [Tutorial: Configure Apache Web Server on Amazon Linux to use SSL/TLS](#).

1. Enable the Extra Packages for Enterprise Linux (EPEL) repository from the Fedora project on your instance.

```
[ec2-user ~]$ sudo yum-config-manager --enable epel
```

2. Install the phpMyAdmin package.

```
[ec2-user ~]$ sudo yum install -y phpMyAdmin
```

Note

Answer `y` to import the GPG key for the EPEL repository when prompted.

3. Configure your `phpMyAdmin` installation to allow access from your local machine. By default, `phpMyAdmin` only allows access from the server that it is running on, which is not very useful because Amazon Linux does not include a web browser.
 - a. Find your local IP address by visiting a service such as whatismyip.com.
 - b. Edit the `/etc/httpd/conf.d/phpMyAdmin.conf` file and replace the server IP address (127.0.0.1) with your local IP address with the following command, replacing `your_ip_address` with the local IP address that you identified in the previous step.

```
[ec2-user ~]$ sudo sed -i -e  
's/127.0.0.1/your_ip_address/g'  
/etc/httpd/conf.d/phpMyAdmin.conf
```

4. Restart the Apache web server to pick up the new configuration.

```
[ec2-user ~]$ sudo service httpd restart  
Stopping httpd: [ OK ]  
Starting httpd: [ OK ]
```

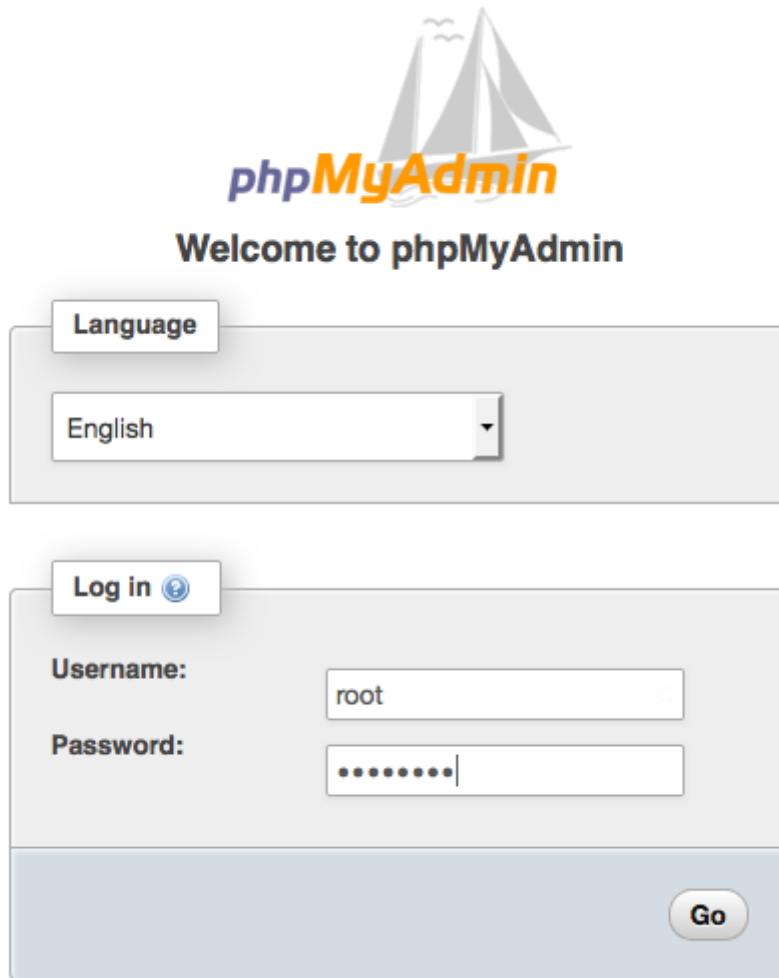
5. Restart the MySQL server to pick up the new configuration.

```
[ec2-user ~]$ sudo service mysqld restart  
Stopping mysqld: [ OK ]  
Starting mysqld: [ OK ]
```

6. In a web browser, enter the URL of your `phpMyAdmin` installation. This URL is the public DNS address of your instance followed by a forward slash and `phpmyadmin`. For example:

```
http://my.public.dns.amazonaws.com/phpmyadmin
```

You should see the `phpMyAdmin` login page:



The image shows the phpMyAdmin welcome screen. At the top is the phpMyAdmin logo, which features a stylized sailboat and the text 'phpMyAdmin'. Below the logo is the text 'Welcome to phpMyAdmin'. There are two main sections: a 'Language' section with a dropdown menu set to 'English', and a 'Log in' section. The 'Log in' section has fields for 'Username' (containing 'root') and 'Password' (containing masked characters). A 'Go' button is located at the bottom right of the login section.

Note

If you get a 403 Forbidden error, verify that you have set the correct IP address in the `/etc/httpd/conf.d/phpMyAdmin.conf` file. You can see what IP address the Apache server is actually getting your requests from by viewing the Apache access log with the following command:

```
[ec2-user ~]$ sudo tail -n 1 /var/log/httpd/access_log | awk  
'{ print $1 }'  
205.251.233.48
```

Repeat [Step 3.b](#), replacing the incorrect address that you previously entered with the address returned here; for example:

Tutorial: Installing a LAMP Web Server on
Amazon Linux

```
[ec2-user ~]$ sudo sed -i -e  
's/previous_ip_address/205.251.233.48/g'  
/etc/httpd/conf.d/phpMyAdmin.conf
```

After you've replaced the IP address, restart the `httpd` service with [Step 4](#).

7. Log into your `phpMyAdmin` installation with the `root` user name and the MySQL root password you created earlier. For more information about using `phpMyAdmin`, see the [phpMyAdmin User Guide](#).