# **Installing a LAMP Web Server on Amazon Linux**

The following procedures help you install the Apache web server with PHP and MySQL support on your Amazon Linux instance

**Prerequisites:**

Configure security group to allow SSH (port 22), HTTP (port 80), and HTTPS (port 443) connections.

**To install and start the LAMP web server on Amazon Linux**

1. [Connect to your instance](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html#ec2-connect-to-instance-linux).
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.

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

Now install the Apache web server, MySQL, and PHP software packages.

1. 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 php70 mysql56-server php70-mysqlnd php70-soap php70-mcrypt php70-mbstring php70-gd php70-xml**

1. Start the Apache web server.

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

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

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

1. 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).

1. 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**).

**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.

To allow ec2-user to manipulate files in /var/www/html 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**

1. 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.

1. Log out and then log back in again, and verify your membership in the www group.
   1. Log out.

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

* 1. 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

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

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

1. 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 {} \;**

1. 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**

For information about enabling HTTPS on your server, see [Tutorial: Configure Apache Web Server on Amazon Linux to use SSL/TLS](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/SSL-on-an-instance.html).

**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:   
...  
  
PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !  
...  
  
Starting mysqld: [ OK ]

1. Run **mysql\_secure\_installation**.

[ec2-user ~]$ **sudo mysql\_secure\_installation**

* 1. When prompted, enter a password for the root account.
     1. Enter the current root password. By default, the root account does not have a password set, so press **Enter**.
     2. Type **Y** to set a password, and enter a secure password twice.

**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.

* 1. Type **Y** to remove the anonymous user accounts.
  2. Type **Y** to disable remote root login.
  3. Type **Y** to remove the test database.
  4. Type **Y** to reload the privilege tables and save your changes.

1. (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.

**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.

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***

1. Install the phpMyAdmin package.

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

1. 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.

[ec2-user ~]$ **sudo sed -i -e 's/127.0.0.1/*your\_ip\_address*/g' /etc/httpd/conf.d/phpMyAdmin.conf**

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

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

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

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

1. 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](http://my.public.dns.amazonaws.com/phpmyadmin)

You should see the phpMyAdmin login page