

## Guide-Setup MariaDB Database on EC2 Lamp Stack

Web Application Development COP3834

Professor Navarro

This resource is to install MariaDB for a lamp stack on Linux. You need to have the rest of the lamp stack installed to follow these steps.

Step 1: Install MariaDB from CLI on your EC2

```
[ec2-user@ip-172-31-14-199 wpDir]$ sudo yum install -y mariadb-server
```

Step 2: Verify all packages are installed

```
[ec2-user@ip-172-31-14-199 wpDir]$ sudo yum list installed httpd mariadb-server php-mysqld
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
httpd.x86_64                2.4.48-2.amzn2                @amzn2-core
mariadb-server.x86_64       3:10.2.38-1.amzn2.0.1         @amzn2extra-lamp-mariadb10.2-php7.2
php-mysqld.x86_64           7.2.34-1.amzn2                @amzn2extra-php7.2
[ec2-user@ip-172-31-14-199 wpDir]$
```

If your server is installed and running with PHP, and your file permissions are set correctly, you will need to test that Apache responds to PHP requests before continuing with MariaDB.

You will need to create a PHP file in the `/var/www/html` directory that is reachable from the internet.

see: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-lamp-amazon-linux-2.html>

Step 3: Create a PHP file in the Apache document root.

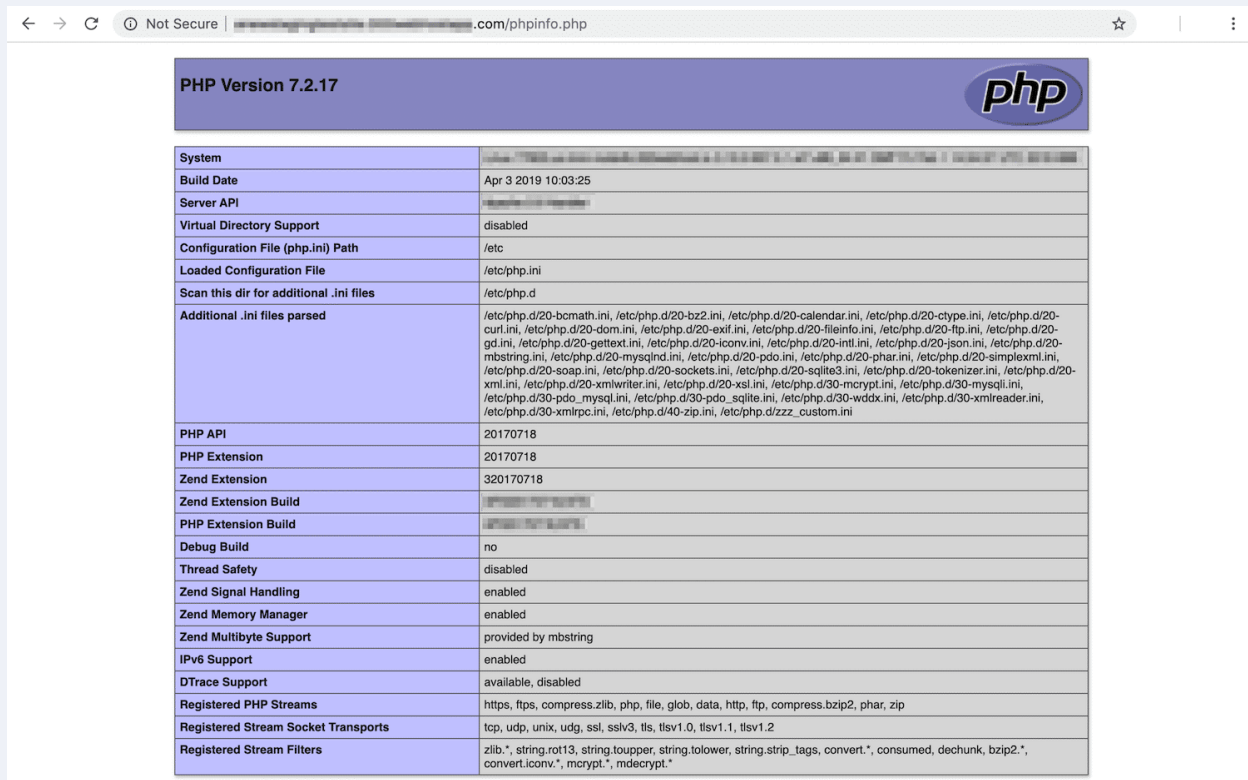
```
[ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

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 the set file permissions.

Step 4: Type the URL of the file that you just created in web browser to verify PHP

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 in your browser similar to:



<b>PHP Version 7.2.17</b>	
<b>System</b>	
<b>Build Date</b>	Apr 3 2019 10:03:25
<b>Server API</b>	
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc
<b>Loaded Configuration File</b>	/etc/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php.d
<b>Additional .ini files parsed</b>	/etc/php.d/20-bcmath.ini, /etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gd.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-intl.ini, /etc/php.d/20-json.ini, /etc/php.d/20-mbstring.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-simplexml.ini, /etc/php.d/20-soap.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/20-xml.ini, /etc/php.d/20-xmlwriter.ini, /etc/php.d/20-xsl.ini, /etc/php.d/30-mcrypt.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini, /etc/php.d/30-wddx.ini, /etc/php.d/30-xmldr.ini, /etc/php.d/30-xmlrpc.ini, /etc/php.d/40-zip.ini, /etc/php.d/zzz_custom.ini
<b>PHP API</b>	20170718
<b>PHP Extension</b>	20170718
<b>Zend Extension</b>	320170718
<b>Zend Extension Build</b>	
<b>PHP Extension Build</b>	
<b>Debug Build</b>	no
<b>Thread Safety</b>	disabled
<b>Zend Signal Handling</b>	enabled
<b>Zend Memory Manager</b>	enabled
<b>Zend Multibyte Support</b>	provided by mbstring
<b>IPv6 Support</b>	enabled
<b>DTrace Support</b>	available, disabled
<b>Registered PHP Streams</b>	https, ftps, compress.zlib, php, file, glob, data, http, ftp, compress.bzip2, phar, zip
<b>Registered Stream Socket Transports</b>	tcp, udp, unix, udg, ssl, sslv3, tls, tlsv1.0, tlsv1.1, tlsv1.2
<b>Registered Stream Filters</b>	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, bzip2.*, convert.iconv.*, mcrypt.*, mdecrypt.*

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.

```
[ec2-user ~]$ sudo yum list installed httpd mariadb-server php-mysqlnd
```

If any of the required packages are not listed in your output, install them with the `sudo yum install` package command. Also verify that the `php7.2` and `lamp-mariadb10.2-php7.2` extras are enabled in the output of the `amazon-linux-extras` command.

Step 5: Delete the `phpinfo.php` file.

Do this step if everything was ok up until here.

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

Although this php file has useful information, it should not be broadcast to the internet for security reasons.

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.

#### Step 6: Secure the database server

The default installation of the MariaDB 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 MariaDB server, it is recommend performing this procedure.

#### Step 7: Start the MariaDB server

```
[ec2-user ~]$ sudo systemctl start mariadb
```

#### Step 8: Run `mysql_secure_installation`.

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

When prompted, Type the current root password. By default, the **root account does not have** a password set. **Press Enter**.

#### Step 9: Set password

**Type Y** to set a password, and type a secure password twice. For more information about creating a secure password, see <https://identitysafe.norton.com/password-generator/>. Make sure to store this password in a safe place.

Setting a root password for MariaDB 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.

- **Type Y** to remove the anonymous user accounts.
- **Type Y** to disable the remote root login.
- **Type Y** to remove the test database.

- **Type Y** to reload the privilege tables and save your changes.

(Optional) If you do not plan to use the MariaDB server right away, stop it. You can restart it when you need it again.

Step 10: Stop the database

```
[ec2-user ~]$ sudo systemctl stop mariadb
```

Step 11: To start db at every boot (optional)

```
[ec2-user ~]$ sudo systemctl enable mariadb
```

Complete

Note: Instructions were taken and modified from Amazon aws documentation (2021)