Install LEMP Stack (Nginx, MariaDB, PHP7.2) on RHEL 8/CentOS 8

This tutorial is going to show you how to install LEMP stack on RHEL 8 and CentOS 8.

What's LEMP Stack?

A software stack is a set of software tools bundled together. LEMP stands for Linux, Nginx (pronounced engine X), MariaDB/MySQL and PHP, all of which are open source. It is the most common software stack that powers dynamic websites and web applications. Linux is the operating system; Nginx is the web server; MariaDB/MySQL is the database server and PHP is the server-side scripting language responsible for generating dynamic web pages.

Prerequisites

You can download and install RHEL 8 by following the tutorial below.

- How to Download and Install RHEL 8 for Free

If you are looking for a VPS (Virtual Private Server), then you can register an account at Vultr via my referral link to get $50 free credit for use over 30 days.

This tutorial uses root account to manage administration tasks. To switch to root, run the following command and enter root password.

```
su -
```

Step 1: Install Nginx Web Server

on RHEL 8/CentOS 8

Nginx is a high performance web server and very popular these days. It also can be used as a reverse proxy and caching server. Enter this command to install Nginx Web server.

```
yum install nginx -y
```

After it’s installed, we can start Nginx with this command:

```
systemctl start nginx
```

Enable Nginx to auto start at system boot time by running the following command.

```
systemctl enable nginx
```

Now check its status.

```
systemctl status nginx
```

Output:

```
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-12-05 02:04:00 EST; 7s ago
 Main PID: 5032 (nginx)
    Tasks: 2 (limit: 11512)
   Memory: 8.4M
   CGroup: /system.slice/nginx.service
           ├─5032 nginx: master process /usr/sbin/nginx
           └─5034 nginx: worker process
```

“**Enabled**” indicates that auto start at boot time is enabled and we can see that Nginx is running. Notice that the above command will not immediately quit after running. You need to press “**q**” to make it quit.

Check Nginx version.

```
nginx -v
```

Output:

```
nginx version: nginx/1.14.1
```

If you are installing LEMP on your local RHEL 8/CentOS 8 computer, then type `127.0.0.1` or `localhost` in the browser address bar. You should see the “Welcome to Nginx” Web page, which means Nginx Web server is running properly.

By default, RHEL 8/CentOS 8 forbids public access to port 80. To allow other computers to access the web page, we need to open port 80 in firewalld, the dynamic firewall manager on RHEL/CentOS. Run the following command to open port 80.

```
firewall-cmd --permanent --zone=public --add-service=http
```

If you want to enable HTTPS on Nginx later, then you also need to open port 443.

```
firewall-cmd --permanent --zone=public --add-service=https
```

The `--permanent` option will make this firewall rule persistent across system reboots. Next, reload the firewall daemon for the change to take effect.

```
systemctl reload firewalld
```

Now the Nginx web page is accessible publicly.

Finally, we need to make user `nginx` as the owner of web directory. By default it's owned by the root user.

```
chown nginx:nginx /usr/share/nginx/html -R
```

Step 2: Install MariaDB Database Server on RHEL 8/CentOS 8

MariaDB is a drop-in replacement for MySQL. It is developed by former members of MySQL team who are concerned that Oracle might turn MySQL into a closed-source product. Enter the following command to install MariaDB on RHEL 8/CentOS 8.

```
yum install mariadb-server mariadb -y
```

After it's installed, we need to start it.

```
systemctl start mariadb
```

Enable auto start at system boot time.

```
systemctl enable mariadb
```

Check status:

```
systemctl status mariadb
```

output:

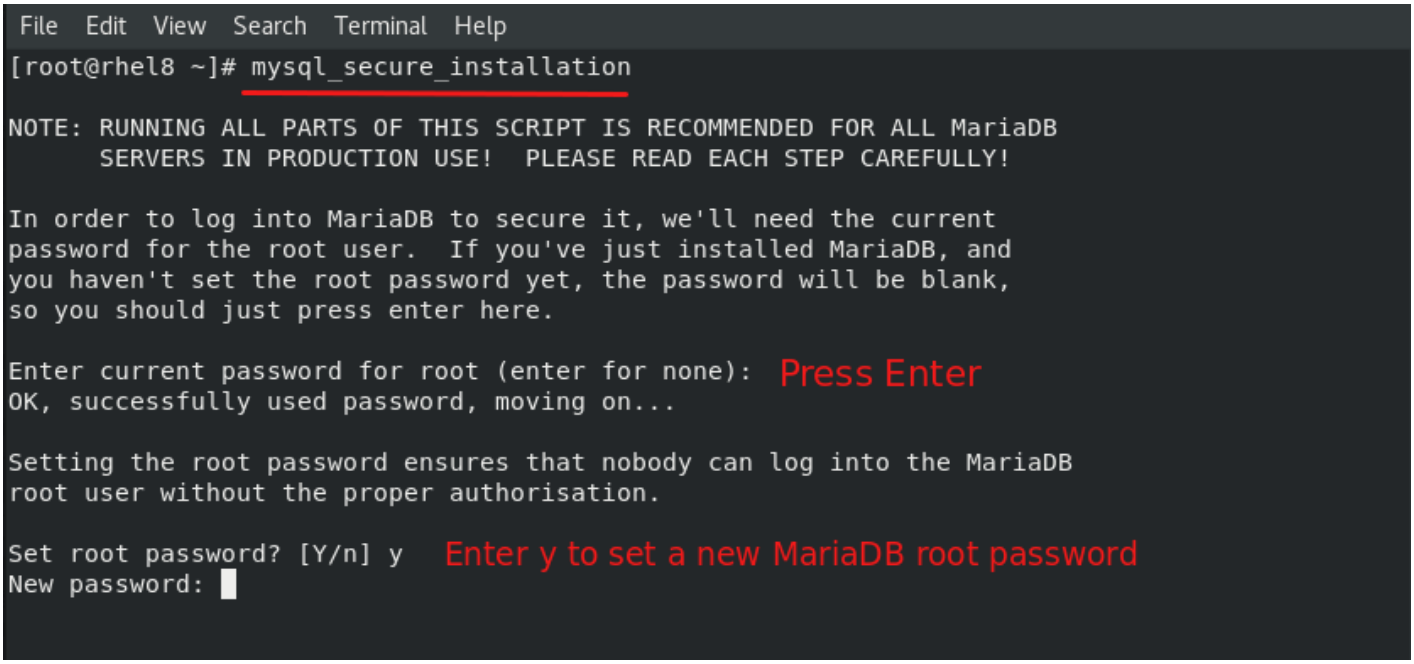
```
● mariadb.service - MariaDB 10.3 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset:
disable>
   Active: active (running) since Wed 2018-12-05 02:40:44 EST; 8s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
 Main PID: 17582 (mysqld)
   Status: "Taking your SQL requests now..."
    Tasks: 30 (limit: 11512)
   Memory: 75.2M
```

```
    CGroup: /system.slice/mariadb.service
            └─17582 /usr/libexec/mysqld --basedir=/usr
```

"**Enabled**" indicates that auto start at boot time is enabled and we can see that MariaDB server is running. Now we need to run the security script.

```
mysql_secure_installation
```

When it asks you to enter MariaDB root password, press Enter key as the root password isn't set yet. Then enter `y` to set the root password for MariaDB server.



Next, you can press Enter to answer all remaining questions, which will remove anonymous user, disable remote root login and remove test database. This step is a basic requirement for MariaDB database security. (Note that the letter `Y` is capitalized, which means it's the default answer.)

```
Remove anonymous users? [Y/n] Press Enter
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] Press Enter
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Press Enter
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n]  Press Enter
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[root@rhel8 ~]#
```

Now you can run the following command and enter MariaDB root password to log into MariaDB shell.

```
mysql -u root -p
```

```
File  Edit  View  Search  Terminal  Help
[root@rhel8 ~]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 18
Server version: 10.3.10-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> exit;
Bye
[root@rhel8 ~]#
```

To exit, run

```
exit;
```

Step 3: Install PHP-FPM on RHEL 8/CentOS 8

Install PHP and related modules using the following command:

```
yum install php php-mysqlnd php-fpm php-opcache php-gd php-xml php-mbstring -y
```

After it’s installed, we need to start it.

```
systemctl start php-fpm
```

Enable auto start at system boot time.

```
systemctl enable php-fpm
```

Check status:

```
systemctl status php-fpm
```

output:

```
● php-fpm.service - The PHP FastCGI Process Manager
   Loaded: loaded (/usr/lib/systemd/system/php-fpm.service; enabled; vendor preset:
disable>
   Active: active (running) since Wed 2018-12-05 03:06:01 EST; 22s ago
 Main PID: 18631 (php-fpm)
   Status: "Processes active: 0, idle: 5, Requests: 0, slow: 0, Traffic: 0req/sec"
    Tasks: 6 (limit: 11512)
   Memory: 29.6M
   CGroup: /system.slice/php-fpm.service
           ├─18631 php-fpm: master process (/etc/php-fpm.conf)
           ├─18632 php-fpm: pool www
```

“**Enabled**” indicates that auto start at boot time is enabled and we can see that PHP-FPM is running. Now edit the PHP-FPM config file:

```
nano /etc/php-fpm.d/www.conf
```

By default, PHP-FPM runs as the `apache` user. Since we are using Nginx web server, we need to change it. Find the following two lines.

```
user = apache
group = apache
```

Change them to

```
user = nginx
group = nginx
```

In this file you can find the following line.

```
listen = /run/php-fpm/www.sock
```

This indicates that PHP-FPM is listening on a Unix socket instead of a TCP/IP socket, which is good. Save and close the file. Reload PHP-FPM for the changes to take effect.

```
systemctl reload php-fpm
```

Step 4: Test PHP

By default, the Nginx package on RHEL 8/CentOS 8 includes configurations for PHP-FPM (`/etc/nginx/conf.d/php-fpm.conf` and `/etc/nginx/default.d/php.conf`). To test PHP-FPM with Nginx Web server, we need to create a `info.php` file in the document root directory.

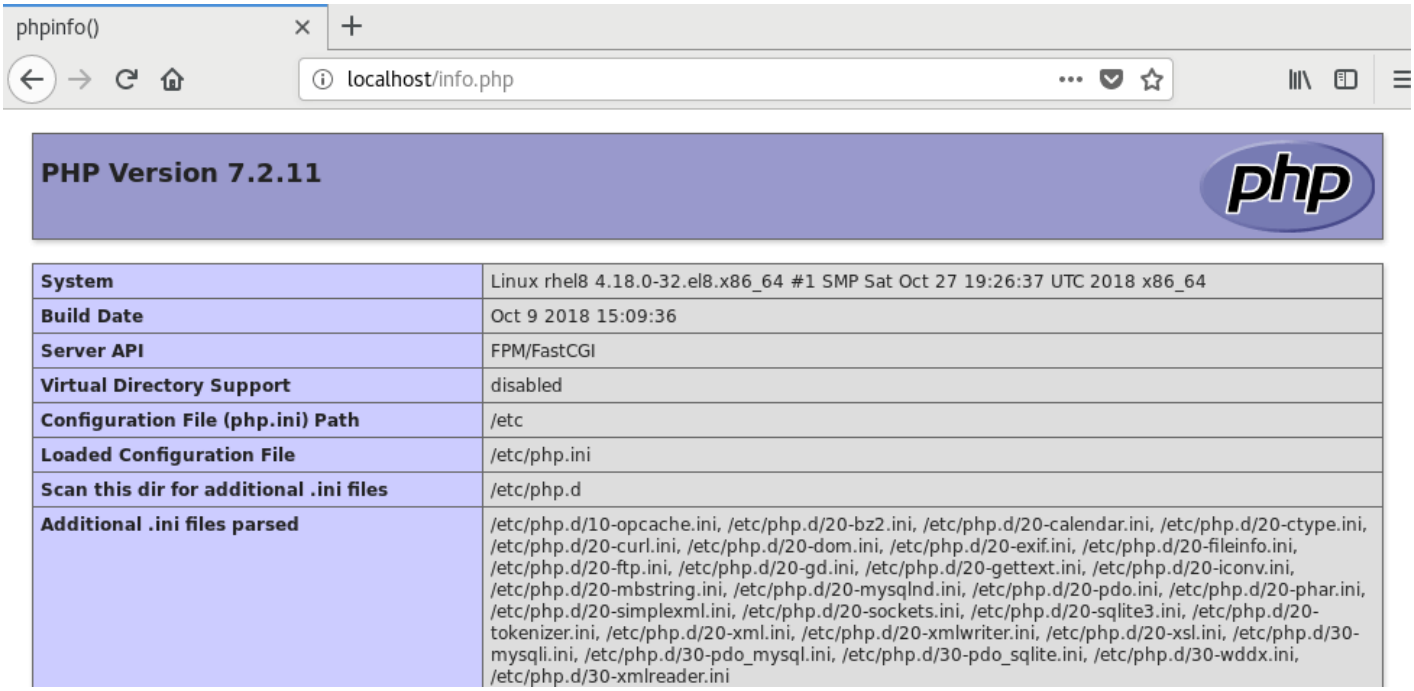
```
nano /usr/share/nginx/html/info.php
```

Paste the following PHP code into the file.

```
<?php phpinfo(); ?>
```

Save and close the file. If you installed LEMP stack on a local RHEL 8/CentOS 8 server, type in `127.0.0.1/info.php` or `localhost/info.php` in the browser address bar. You should see your server's PHP information. This means PHP scripts can run properly with Nginx web server.

If RHEL 8/CentOS is running on a remote server, then enter `server-ip-address/info.php` in browser address bar. Replace `sever-ip-address` with your actual IP address.



If the browser fails to display the PHP info but prompt you to download the **info.php** file, simply restart Nginx and PHP-FPM.

```
sudo systemctl restart nginx php-fpm
```

Then you should be able to see the PHP info in the web browser.

Nginx Automatic Restart

If for any reason your Nginx process is killed, you need to run the following command to restart it.

```
sudo systemctl restart nginx
```

Instead of manually typing this command, we can make Nginx automatically restart by editing the `nginx.service` systemd service unit. To override the default systemd service configuration, we create a separate directory.

```
sudo mkdir -p /etc/systemd/system/nginx.service.d/
```

Then create a file under this directory.

```
sudo nano /etc/systemd/system/nginx.service.d/restart.conf
```

Add the following lines in the file, which will make Nginx automatically restart 5 seconds after a failure is detected.

```
[Service]
Restart=always
RestartSec=5s
```

Save and close the file. Then reload systemd.

```
sudo systemctl daemon-reload
```

To check if this would work, kill Nginx with:

```
sudo pkill nginx
```

Then check Nginx status. You will find Nginx automatically restarted.

```
systemctl status nginx
```

Revision #1
Created 11 July 2021 18:57:27 by Admin
Updated 11 July 2021 18:57:57 by Admin