Snippets

Al Mehedi / WordPress with LAMP Stack on CentOS 8 Stop watching Clone Edit Delete

Source Revisions

Created by Al Mehedi last modified just now

wordpress_with_lamp_stack_on_centos_8.markdown Raw

About

WordPress is the most popular and widely used content management system or blogging platform in the world. WordPress can be used as a Blog, eCommerce platform, Portfolio Website. At the backend, WordPress uses a MySQL database for storage and works on PHP processing. WordPress provides a lot of features and these features can also be extended by using a wide variety of available plugins for it

Installing Apache

Apache is the most popular and most used HTTP web server in the world. Apache is the opensource, cross-platform, powerful, stable, reliable and free web server providing features which can be extended by the wide variety of modules. It is also used as a reverse proxy server in different scenarios.

```
# dnf install httpd
```

After completion of the above command, Apache is installed on your system. Run the following command to check apache service status.

```
# systemctl enable httpd
# systemctl start httpd
# systemctl status httpd
```

Verify Apache Installation

Open your favorite browser and open below URL to verify Apache installation.

```
http://YOUR_IP_ADDRESS_OR_DOMAIN_NAME/
```

Apache important default files

Following are the Apache important files and directories:

- Apache main configuration file is located at `/etc/httpd/conf/httpd.conf`
- Apache main configuration files directory: `/etc/httpd/`
- Ports where apache can listen are defined in `/etc/httpd/ports.conf`
- Virtual Host files stored at `/etc/httpd/conf.d` directory.
- Error log file located at `/var/log/httpd/error.log`
- Access log file located at `/var/log/httpd/access.log`
- Document root for web files `/var/www/html/`

Installing MySQL

MySQL is an open-source database management system, commonly installed as part of the popular LEMP (Linux, Nginx, MySQL/MariaDB, PHP/Python/Perl) stack. It implements the relational model and Structured Query Language (SQL) to manage and query data.

```
# dnf install mysql-server
```

After installing the `mysql-server` run following commands:

```
# systemctl start mysqld
# systemctl status mysqld
# systemctl enable mysqld
```

Securing MySQL

MySQL includes a security script that allows you to change some default configuration options in order to improve MySQL's security.

To use the security script, run the following command:

```
# mysql_secure_installation
```

Testing MySQL

You can verify your installation and get information about it by connecting with the `mysqladmin` tool, a client that let's you run administrative commands. Use the following command to connect to MySQL as root (`-u root`), prompt for a password (`-p`), and return the installation's version:

```
# mysqladmin -u root -p version
```

If you'd like to connect to MySQL and begin adding data to it, run the following:

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

Create MySQL Database and Grant Permissions

Create the MySQL database for your WordPress typing below command:

```
mysql> CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
```

Now create a MySQL user and grant permissions for it using below command:

```
mysql> CREATE USER 'new_user'@'localhost' IDENTIFIED BY 'password';
mysql> GRANT ALL PRIVILEGES ON wordpress.* TO 'new_user'@'localhost';
```

Now flush the database so the current MySQL version will know about these changes:

```
mysql> FLUSH PRIVILEGES;
```

Exit MySQL using below command:

```
mysql> EXIT;
```

Tuning MySQL

The values below should be adjusted depending on the available physical memory (RAM) of the MySql server. The following configuration is based on an instance of 4GB RAM.

Open and edit MySQL configuration file by running: `# nano /etc/my.cnf.d/mysql-server.cnf`.

```
innodb_buffer_pool_instances = 4
innodb_buffer_pool_size = 2G
innodb_file_per_table = ON
innodb_log_file_size = 256M
innodb_stats_on_metadata = OFF
```

Finally restart the php-fpm by running: `# systemctl restart mysqld`.

Installing PHP 7.x

PHP (Hypertext Preprocessor) is the most popular server-side scripting language and used in developing static and dynamic web sites.

Enabling Remi Repository

In the default CentOS 8 and RHEL 8 package repositories, **PHP 7.2 / 7.3** is available and to install latest version of **PHP 7.x**, we must configure REMI repositories, so to enable the repository, run the following commands:

```
# dnf install https://rpms.remirepo.net/enterprise/remi-release-8.rpm
```

Once the repository is configured and enabled, run the following dnf command to view available PHP versions:

```
# dnf module list php
```

Enabling Latest PHP Module Stream

```
# dnf module enable php:remi-7.x
```

Install and Verify PHP

```
# dnf install php php-cli php-common
# php -v
```

While installing PHP, php-fpm is also installed as its dependency, it will be treated as FastCGI Server, so use below command to start and enable FPM service:

```
# systemctl start php-fpm
# systemctl enable php-fpm
# systemctl status php-fpm
```

Adjusting PHP-FPM for Performance + Low Memory

PHP-FPM has as a default configuration that uses more memory than necessary. It has spare php-fpm processes ready to go, taking up memory in case there is PHP code to process. While not a problem if you have tons of RAM, it can be an issue for low RAM VPS and if you are using aggressive page caching then it is memory being used unnecessarily that could be used by MariaDB MySQL or other critical processes.

Open and edit PHP-FPM configuration by running `nano /etc/php-fpm.d/www.conf` and adjust the following values:

```
pm = ondemand
pm.max_children = 75

;pm.start_servers = 20
;pm.min_spare_servers = 10
;pm.max_spare_servers = 30

pm.process_idle_timeout = 10s;
pm.max_requests = 750
```

Calculate `pm.max_children` by ram's ((available Mb - (database + other workload)) / 60Mb) = number of max_children.

Test the validity of your php-fpm configuration syntax by running:

```
# php-fpm -t
```

Finally restart the php-fpm by running: `# systemctl restart php-fpm`.

Updating PHP Parameters

While using the php in different CMS (Content Management System) like WordPress, Drupal, Joomla and MediaWiki etc we might have to update default PHP parameters to improve the performance. These parameters can be alerted from its configuration file `/etc/php.ini`.

```
upload_max_filesize = 32M
post_max_size = 48M
memory_limit = 256M
max_execution_time = 600
max_input_vars = 3000
max_input_time = 1000
```

Save and exit the file, To make the above changes into the effect restart your apache server by running: `# systemctl restart httpd`.

Installing PHP Extensions for WordPress

The following extensions are required to install and run WordPress on your CentOS 8 machine. WordPress recommends PHP v7.3 for the installation.

```
# dnf install php-dom php-simplexml php-ssh2 php-xml php-xmlreader php-curl php-date php-exif php-filter php-ftp php-gd php-hash php-iconv php-json php-libxml php-pecl-imagick php-mbstring php-mysqlnd php-openssl php-pcre php-posix php-sockets php-spl php-tokenizer php-zlib php-zip php-mcrypt mod_ssl
```

Download and Setup WordPress

Navigate to `/tmp` directory using below command:

```
# cd /tmp
```

Download the latest WordPress setup using below `curl` command:

```
# curl -O https://wordpress.org/latest.tar.gz
```

You should extract the downloaded file using below command:

```
# tar xzvf latest.tar.gz
```

Now create the configuration file for WordPress using below command:

```
# cp /tmp/wordpress/wp-config-sample.php /tmp/wordpress/wp-config.php
```

Then copy all the files to `<example.com>` directory using below command:

```
# cp -a /tmp/wordpress/. /var/www/example.com
```

Now change the ownership of `<example.com>` directory using below command:

```
# chown -R apache:apache /var/www/example.com
```

To set up the WordPress configuration file you should generate some configuration files for it. Do so by running below command:

```
# curl -s https://api.wordpress.org/secret-key/1.1/salt/
```

The generated output keys should be something like:

```
define('AUTH_KEY',         'm=w)!7{-EEc&JYU~$wd@jTrqFseaZ0D-4Vd/?!>_hcF*BmQ+S2Do!QP>>O-|OI21');
define('SECURE_AUTH_KEY',  'S?lk-{RG 5K~sd*1$N<aZ18jy|^0n#-@eGqBhk3#dJy2M-|jUruu[T+ cYfJ^@2-');
define('LOGGED_IN_KEY', '>i*8?IA#h/.@?6MezjmoBWm&&b+h1YP?T.]Y=&*^h9[Bm`ThdbJ5zepb824LUd;-');
define('NONCE_KEY',        'cPim1L6}H1rQLtLj|FrN1DO:LZVsh`rr}5 `}k,f~%u)papX4|_J^Q%PKJ44uF[l');
define('AUTH_SALT',        ',+Aa_iZ/%yj5?-0F.O>Ogd6jCLU+2_2M$+1Zo-hUog70lLa$)YI@wbzkN<~v!Acd');
define('SECURE_AUTH_SALT', '-9sQ8iLS}1-iEX)b<A6(JNuPIGv2SV5ZiHV])4i+@oi6FG76$4{A@c*fj8[ *Uc-');
define('LOGGED_IN_SALT', 'K$i5b^g?TK4M|w;mqlh>m9ZJ5eVAq0X;we}jvw:JNkKm-O|-=GdH-{I><`J(ZgKB');
define('NONCE_SALT', 'c_VY?z=E}2r0A&r!F/qk*rtM3>K-Id+z*qG*^2g#4/-sR2%GP>b|{<97nL4uP8K/');
```

Now open the `/var/www/<example.com>/wp-config.php` file,

```
# nano /var/www/<example.com>/wp-config.php
```

Find the section and replace it with the above data.

Now you should also update configuration file as given below for database connection by replacing `password-you-provided` with the password you entered in above instructions:

```
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpressuser');

/** MySQL database password */
define('DB_PASSWORD', 'password-you-provided');

define('FS_METHOD', 'direct');
define('WP_MEMORY_LIMIT', '128M');
```

Configure Apache

Now create and open `/etc/httpd/conf.d/<example.com>.conf` file using below command:

```
# nano /etc/httpd/conf.d/<example.com>.conf
```

And paste the following code in the above file:

```
<VirtualHost *:80>
    ServerName <your-domain.com>
    RewriteEngine On
    RewriteCond %{HTTPS} off
    RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
</VirtualHost>

<VirtualHost *:443>
    ServerName <your-domain.com>

    DirectoryIndex index.html index.php
    DocumentRoot /var/www/<your-domain.com>

    SSLEngine On
    SSLCertificateFile /etc/letsencrypt/live/<your-domain.com>/cert.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/<your-domain.com>/privkey.pem
    SSLCertificateChainFile /etc/letsencrypt/live/<your-domain.com>/chain.pem

    ErrorLog /var/log/httpd/<your-domain.com>-error.log
    CustomLog /var/log/httpd/<your-domain.com>-access.log combined

    Options FollowSymLinks
    <Directory "/var/www/<your-domain.com>">
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

Restart Apache by using the command: `# systemctl restart httpd`

Comments (0)

What would you like to say?