THE PERFECT NGINX SERVER

UBUNTU 20.04

LEMP STACK

NGINX | MARIADB | PHP

INSTALL | HARDEN | OPTIMIZE

UPDATED DECEMBER 2020

php 8.0

IMPORTANT NOTICE - PHP 8.0

As of December 2020, I don't recommend php 8 and WordPress 5.6 be used on a production server.

Time is needed for the theme and plugin developers to release versions that are fully compatible with WordPress 5.6 and php 8.

For this reason, **test WordPress 5.6 and php8.0 on a test server first, not your production server.**

Your production server is not used to test new software releases!!!

INSTALL NGINX MARIADB & PHP

There are a few commands we need to cover regarding the Advanced Package Manager, `apt`:

APT COMMANDS:

Installing Packages:

```
sudo apt install package_name(s)
```

Removing Packages: configuration files are not removed

```
sudo apt remove package_name(s)
```

Purging Packages: configuration files are removed

```
sudo apt purge package_name(s)
```

Find a Package

```
sudo apt-cache search package_name
```

Package Information

```
sudo apt-cache show package_name
```

INSTALLING NGINX

Always ensure the package list is up to date before installing any packages:

```
sudo apt update
```

We are going to install nginx and the libnginx-mod-http-headers-more-filter packages.

```
sudo apt install nginx libnginx-mod-http-headers-more-filter
```

After installing nginx, check the status of nginx:

```
sudo systemctl status nginx
```

INSTALLING MARIADB

```
sudo apt install mariadb-server
```

Check the status of MariaDB.

```
sudo systemctl status mariadb
```

INSTALLING PHP7.4

The following php modules are recommended for WordPress:

```
php7.4-fpm
php7.4-gd
php7.4-json
php7.4-mbstring
php7.4-mysql
php7.4-xml
php7.4-xmlrpc
php7.4-opcache
php7.4-cli
php7.4-zip
php7.4-soap
php7.4-intl
php7.4-bcmath
php7.4-curl
php-ssh2
```

php-imagick

Please make a note of the php-imagick package. Even if WordPress site health indicates its missing, only install the package if it's needed. The package php-imagick has pretty much been replaced by the php-gd package. If a plugin requires this package, install it as sudo apt install php-imagick. Be aware that this module can cause issues with certain plugins.

Refer to the video lecture regarding the installation of the php packages:

```
sudo apt install php7.4-{fpm,gd,json,mbstring,mysql,xml,xmlrpc,opcache,cli,zip,soap,intl,bcmath,curl} php-ssh2
```

Check the status of php-fpm:

```
sudo systemctl status php7.4-fpm
```

INSTALLING PHP 8.0

ADD REPOSITORY CONTAINING PHP8.0

```
sudo apt install software-properties-common
sudo add-apt-repository ppa:ondrej/php
sudo apt update && sudo apt upgrade
```

Press ENTER when prompted after typing the add-apt-repository command

INSTALL PHP MODULES

```
sudo apt install php8.0-{fpm,gd,mbstring,mysql,xml,opcache,cli,zip,soap,intl,bcmath,curl} php-ssh2
```

Check the status of php-fpm:

```
sudo systemctl status php8.0-fpm
```

SET PHP VERSION IN YOUR NGINX CONFIGURATION FILES

EXISTING

```
    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php7.4-fpm.sock;
        include /etc/nginx/include_files/fastcgi_optimize.conf;
    }
```

MODIFIED FOR PHP 8.0 - change php7.4 to php8.0

```
    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.0-fpm.sock;
        include /etc/nginx/include_files/fastcgi_optimize.conf;
    }
```

HARDEN NGINX MARIADB & PHP

NGINX

The nginx defaults are actually very secure and further hardening of the server is a combination of securing and optimizing. We will complete this step, in detail, in the next section.

For now, we need to prevent information leakage, in this case it's to prevent the nginx version number from being displayed on various pages and to remove the server name from the http headers.

nginx.conf is the main nginx configuration file and is located in the `/etc/nginx` directory. Make a backup copy of the file prior to editing the file.

Refer to the video lecture regarding the initial changes that need to be made to the nginx.conf file

```
more_clear_headers 'Server';
```

After making any changes to a nginx configuration file, you need to test the configuration syntax and then reload nginx to enable the changes in configuration.

Test the nginx configuration syntax:

```
sudo nginx -t
```

Reload nginx to enable the changes in configuration:

```
sudo systemctl reload nginx
```

MARIADB

Securing MariaDB means that we need to remove any dangerous default settings that have been applied during the installation process. This is accomplished by running the security script, `mysql_secure_installation`.

```
sudo mysql_secure_installation
```

PHP 7.4 and PHP 8.0

The main php configuration file is `php.ini` and is located in the following directory: /etc/php/7.4/fpm/

As before, make a backup copy of php.ini file.

We need to make the following changes in the php.ini file:

```
allow_url_fopen = Off

cgi.fix_pathinfo = 0

expose_php = Off
```

After making any changes to the php.ini file, the php7.4-fpm service needs to be restarted to enable the changes you have made to the php-fpm configuration:

```
sudo systemctl restart php7.4-fpm
```

After making any changes to the php.ini file, the php8.0-fpm service needs to be restarted to enable the changes you have made to the php-fpm configuration:

```
sudo systemctl restart php8.0-fpm
```

As php.ini is a large file, use Nano's Search Function [CTRL + W] to locate the directive you want to change.

OPTIMIZE NGINX MARIADB PHP7.4 and PHP8.0

NGINX

Optimizing nginx is an involved process. Initially we need to optimize the directives in the main, events, and http contexts. The next section of the course deals with optimizing nginx exclusively. So, in the section we will only optimize MariaDB and php.

In depth and detailed nginx optimization follows in the next section.

MARIADB

Initially we are going to implement both a performance and sys schema which will aid the recommendations made by an optimization script, called MySQLTuner.

The main MariaDB configuration file is 50-server.cnf. It's located in the /etc/mysql/MariaDB.conf.d/ directory. As always, make a backup copy of the file before editing.

```
# Performance Schema
performance_schema=ON
performance-schema-instrument='stage/%=ON'
performance-schema-consumer-events-stages-current=ON
performance-schema-consumer-events-stages-history=ON
performance-schema-consumer-events-stages-history-long=ON
```

Restart mariadb

```
sudo systemctl restart mariadb
```

UPDATED URL: Download the following Sys Schema from github.com to your home directory

```
cd
wget https://github.com/FromDual/mariadb-sys/archive/master.zip
```

Install zip and unzip

```
sudo apt update

sudo apt install zip unzip
```

Extract the master.zip file:

```
unzip mariadb-sys-master.zip
```

UPDATED: Change to the extracted directory and install the Sys Schema

```
cd mariadb-sys-master/

sudo mysql -u root < ./sys_10.sql
```

Delete the sys schema directory and zip file.

Download mysqltuner:

```
wget http://mysqltuner.pl/ -O mysqltuner.pl
```

Give MySQLTuner executable permissions:

```
chmod +x mysqltuner.pl
```

PHP

We are going to be editing the main php configuration file, php.ini.

```
upload_max_filesize = 100M
post_max_size = 100M
max_execution_time = 30
max_input_time = 60
max_input_vars = 3000
memory_limit = 256M
```

OPCACHE

```
opcache.memory_consumption=192
opcache.interned_strings_buffer=16
opcache.max_accelerated_files=7963
opcache.validate_timestamps=0
```

After making any changes to the php.ini file, you must restart the php7.4-fpm process to enable the changes.

```
sudo systemctl restart php7.4-fpm
```

After making any changes to the php.ini file, the php8.0-fpm service needs to be restarted to enable the changes you have made to the php-fpm configuration:

```
sudo systemctl restart php8.0-fpm
```

To calculate the opcache.max_accelerated_files.

```
cd /var/www
find . -type f -print | grep php | wc -l
```

CLOSEST PRIME NUMBER

https://www.dcode.fr/closest-prime-number