Installing Apache2 web server, MySQL database server, and PHP language interpreter, on Ubuntu Server

You can install these with LAMP or XAMPP. You can also install these each separately using apt from the command line.

As of April, May June, 2020, Let's install Apache 2.4, MySQL 8.0 and PHP 7.4 on Ubuntu 20.04 LTS.

This is done with root or sudo privilege user access to the system. Login to your Ubuntu server either at your command line interface, using console within a GUI, or remotely using SSH. Putty is good app for remote SSH access to Linux Ubuntu from a Windows operating system.

Installing PHP

The latest PHP 7.4 packages are available in the default repositories for Ubuntu 20.04 LTS. First Update the index and then install PHP on the Ubuntu Server operating system.

```
sudo apt update
sudo apt install -y php7.4
```

Here are some additional PHP modules to install that may be required for your applications. [However, before you install these modules, see the discussion below regarding additional modules required to run NextCloud.]

```
sudo apt install php7.4-curl php7.4-gd php7.4-json php7.4-mbstring php7.4-
xml
```

If you are planning to install NextCloud on your Ubuntu Server, then the above generally required modules are included in the expanded list of PHP modules that are recommended or required to run NextCloud from Apache and using PHP.

Install the PHP Modules that are Prerequisites for NextCloud installation

If you plan to install NextCloud to run on a virtual host from Apache web server, then you should install the following PHP modules that are either recommended or required as prerequisites for the proper functioning of NextCloud. Ensure that you manually enable each of these modules in the php.ini file for purposes of running NextCloud. This involves removing the appropriate #comment marks in php.ini to enable the appropriate modules. There should obviously be other PHP modules to enable for the general operation of PHP as necessary for other websites' applications running from Apache web server on this Ubuntu server.

```
sudo apt install php-imagick php7.4-common php7.4-mysql php7.4-fpm php7.4-gd
php7.4-json php7.4-curl  php7.4-zip php7.4-xml php7.4-mbstring php7.4-bz2
php7.4-intl
```

After the installation of these PHP modules, you may see a notice like the following:

Notice: Not enabling PHP 7.4 FPM by default. Notice: To enable PHP 7.4 FPM in Apache2 do: Notice: a2enmod procy fcgi set enfiv Notice: a2enconf php7.4*fpm Notice: You are seeing this message because you have apache2 package installed.

The apache2 package that is installed and the notice is referring to is the package that was just installed with the installation of PHP 7.4.

Note that if Apache server was already installed before you found this guide, then you should **reload Apache2 servic**e to use these newly installed PHP modules:

```
sudo systemctl reload apache2
```

Next we will install Apache2 web server and configure Virtual Hosts.

Installing Apache2 and Creating Apache VirtualHost

Installing Apache2 (version 2.4)

```
sudo apt-get update

sudo apt-get install apache2
```

Also install libapache2-mod-php module to work PHP with Apache2.

Enter the following command to install it:

```
sudo apt install apache2 libapache2-mod-php7.4
```

Some help sites also include “php-mysql” module appended to the end of the above command. I think we can install that later after we install MySQL server. So keep in mind for later, after MySQL server is installed, the step of entering a command like the following.

```
sudo apt install apache2 php-mysql
```

Creating Apache VirtualHost

In Apache on Ubuntu, the virtual host configuration files are stored under /etc/apache2/sites-available/ directory. With the new Apache installation you will find a default virtual host file there. Create a new Virtual Host configuration file by copying the default file, as follows:

```
cd /etc/apache2/sites-available/

sudo cp 000-default.conf www1.example.com.conf
```

Edit the virtual host configuration file for your requirements, using your favorite editor such as vim. A configuration for www1.example.com configuration can be something like the following:

```
vim www.example.com.conf
```

```
<VirtualHost *:80>
        ServerAdmin webmaster@site1.example.com
        ServerName www1.example.com
        DocumentRoot /var/www/www.example.com/httpdocs

        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
        </Directory>

        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        LogLevel warn
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Create the appropriate directory structure in the file system, in accordance with the path specified in the above virtual host.conf file, and assign or grant the appropriate file permissions for the Apache webserver user (namely www-data:www-data).

```
mkdir -p /var/www/www1.example.com/httpdocs
chmod 755 /var/www/www1.example.com/httpdocs
chown www-data.www-data /var/www/www1.example.com/httpdocs
```

You can now upload your project files (website) on **/var/www/www1.example.com/httpdocs/** directory. For example, create a default page **index.html** file to test virtual host configuration, such as follows:

```
echo "<h1>site1.example.com</h1>" >
/var/www/www1.example.com/httpdocs/index.html
```

Next - Enable the First VirualHost

Remember that we configured the first VirtualHost file under the /etc/apache2/sites-available/ directory. However, it is necessary to enable or activate the VirtualHost under the /etc/apache2/sites-enabled/ directory (an apache directory to which there is no permission for direct access).

Adjust the site-name and use the following command to enable this VirtualHost so that Apache can load this enabled configuration file when the Apache service is restarted (reloaded). so, to enable the new configuration VitualHost, adjust the site-name and run the following from the command line:

```
a2ensite site1.example.com
```

This enables the site in **/etc/apache2/sites-enabled**/

To activate the new configuration by reloading Apache (to bring the site 'live' in the Apache web service), enter the following command:

```
service apache2 reload
```

You will need to setup DNS address entries for this 'domain' and host header.

If you do not have the DNS ready, you can temporarily map the domain by making an entry in the /etc/hosts files.

```
192.168.2.163  www1.example.com
```

Creating Additional Virtual Hosts

Repeat the above steps for site www1 by changing the site-name to whatever it will be, such as www2.exmample.com

Install MySQL Database Server

The apt repositories for Ubuntu 20.04 LTS contain MySQL server version 8.0.

Install mysql-server packages for the MySQL database. Also, install the php-mysql package to use MySQL support using PHP. Use the following command to install it.

```
sudo apt install mysql-server php7.4-mysql
```

The installer will prompt for the root password. This password will work for your MySQL root user.

After installing MySQL execute the following command for initial settings of MySQL server. The script will ask more questions for settings than earlier MySQL versions, such as password validation policy etc.

Enter this command:

```
sudo mysql_secure_installation
```

> *Securing the MySQL server deployment.*
>
> *Connecting to MySQL using a blank password.*
>
> *VALIDATE PASSWORD COMPONENT can be used to test passwords and improve security. It checks the strength of password and allows the users to set only those passwords which are secure enough. Would you like to setup VALIDATE PASSWORD component?*
>
> **Press y|Y for Yes, any other key for No: y**
>
> *There are three levels of password validation policy:*
>
> *LOW Length >= 8*
>
> *MEDIUM Length >= 8, numeric, mixed case, and special characters*
>
> *STRONG Length >= 8, numeric, mixed case, special characters and dictionary file*
>
> **Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 2**
>
> *Please set the password for root here.*
>
> **New password:**
>
> **Re-enter new password:**
>
> *Estimated strength of the password: 100*
>
> **Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : y**
>
> *By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.*
>
> **Remove anonymous users? (Press y|Y for Yes, any other key for No) : y**
>
> *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? (Press y|Y for Yes, any other key for No) : y** *(n should be considered)*

> *Success.*
>
> *By default, MySQL 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? (Press y|Y for Yes, any other key for No) : y**
>
> *- 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? (Press y|Y for Yes, any other key for No) : y**
>
> *Success.*
>
> *All done!*

Installing phpMyAdmin

phpMyAdmin is a web interface to manage MySQL databases and user privileges. Install phpMyAdmin on Ubuntu 20.04 LTS as follows:

```
sudo apt install phpmyadmin
```

When prompted during installation, select “Apache” as the web server running phpMyAdmin.

When prompted to create the phpMyAdmin database, provide your admin user access details and take the remaining steps to complete the installation.

Workaround Errors in setting up phpmyadmin@localhost with all privileges involving dbconfig-common

Are you getting errors when trying to login to PhpMyAdmin as root?

> Cannot log in to the MySQL server
>
> mysqli_real_connect(): (HY000/1045): Access denied for user 'phpmyadmin'@'localhost' (using password: YES)
>
> Connection for controluser as defined in your configuration failed.

> mysqli_real_connect(): (HY000/1698): Access denied for user 'root'@'localhost'
>
> or
>
> #1698 - Access denied for user 'root'@'localhost'

Create a new admin user or new root user and password with access from any IP remotely

Using SSH access to the Ubuntu Server, log into the mysql console using as the user root with the password for root that was established when installing MySQL database server on Ubuntu. This will put us into the console mysql>

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

Enter password:

The SSH prompt should change to the MySQL console prompt that looks like this:

<BLOCK>mysql></BLOCK>

Now, create a new MySQL admin/root user with all privileges including GRANT privileges and make sure that the user can login from any IP remotely rather than only having privileges from @localhost. There are 3 choices of commands in the code block below for creating a new user from the MySQL console prompt; however, only the third choice may actually allow for logging into PhpMyAdmin as a root user from any remote IP.

```
CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';

CREATE USER 'username'@'ip_address' IDENTIFIED BY 'password';

CREATE USER 'username'@'%' IDENTIFIED BY 'password';
```

So, for example, this should work (and ensure that you are logged into the console prompt at mysql>

```
CREATE USER 'rootuser'@'%' IDENTIFIED BY 'YourComplex#54321Pwd!';
```

Now, let's grant the new user named 'rootuser' all privileges including the Grant Option, like this:

```
GRANT ALL PRIVILEGES ON *.* TO 'rootuser'@'%' WITH GRANT OPTION;
```

And lastly, run this command:

```
FLUSH PRIVILEGES;
```

Exit the MySQL console and return to the SSH user prompt, with the exit command:

<BLOCK>mysql> exit<BLOCK>

Now, open your web browser and log into PhpMyAdmin through a webpage on a remote device with

the username 'rootuser' and the password you installed for rootuser. You should have success now.

From:
https://installconfig.com/ - **Install Config Wiki**

Permanent link:
https://installconfig.com/doku.php?id=install_apache_mysql_php_linux_ubuntu_server&rev=1591658014

Last update: **2020/06/08 23:13**