

How to Install LEMP Stack on Debian 10 Buster Server

This tutorial is going to show you how to install Nginx, MariaDB and PHP7.3 (LEMP stack) on [Debian 10 Buster](#). A software stack is a set of software tools bundled together. LEMP stands for **L**inux, **N**ginx, **M**ariaDB/**M**ySQL and **P**HP, all of which are open source and free to use. It is a very 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.

All of the four components are free and open-source. However, since MySQL is now owned by Oracle and there's a chance that Oracle turns it to a closed-source product, we will choose MariaDB instead of MySQL.

Prerequisites of Installing LEMP Stack on Debian 10 Buster

To follow this tutorial, you need a Debian 10 OS running on your local computer or on a remote server.

If you are looking for a VPS (Virtual Private Server), then you can register an account at Vultr via [this special link](#) to get \$50 free credit (for new users only). And if you need to set up LEMP stack with a domain name, I recommend buying domain names from [NameCheap](#) because the price is low and they give whois privacy protection free for life.

Please note that you need to have root privilege when installing software on Debian. You can add **sudo** at the beginning of a command, or use `su -` command to switch to root user.

Step 1: Update Software Packages

Before we install the LEMP stack, it's a good idea to update repository and software packages. Run the following command on your Debian 10 OS.

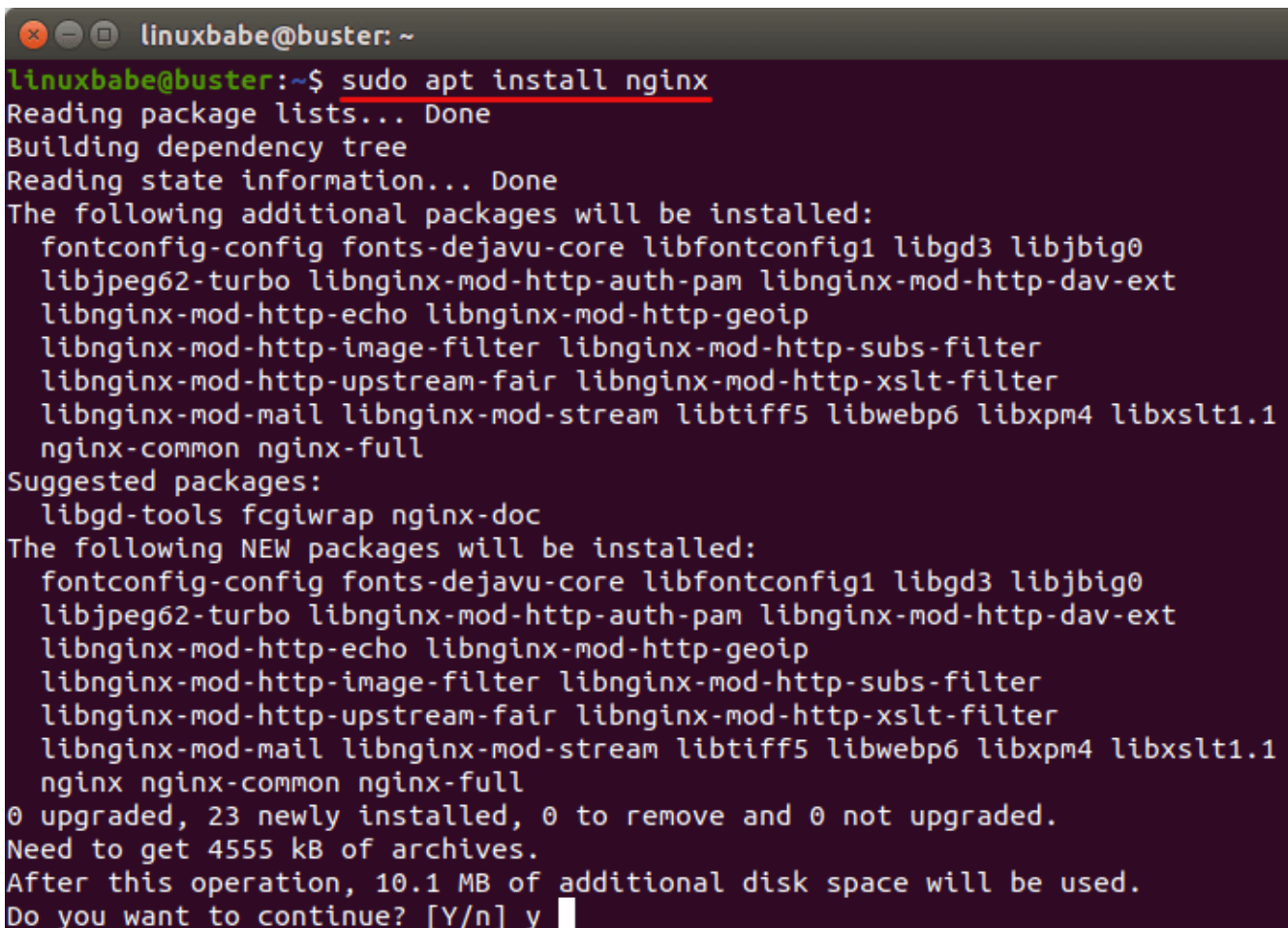
```
sudo apt update
```

```
sudo apt upgrade
```

Step 2: Install Nginx Web Server on Debian 10

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 the following command to install Nginx Web server.

```
sudo apt install nginx
```



```
linuxbabe@buster: ~  
linuxbabe@buster:~$ sudo apt install nginx  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjbig0  
  libjpeg62-turbo libnginx-mod-http-auth-pam libnginx-mod-http-dav-ext  
  libnginx-mod-http-echo libnginx-mod-http-geoip  
  libnginx-mod-http-image-filter libnginx-mod-http-subst-filter  
  libnginx-mod-http-upstream-fair libnginx-mod-http-xslt-filter  
  libnginx-mod-mail libnginx-mod-stream libtiff5 libwebp6 libxpm4 libxslt1.1  
  nginx-common nginx-full  
Suggested packages:  
  libgd-tools fcgiwrap nginx-doc  
The following NEW packages will be installed:  
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjbig0  
  libjpeg62-turbo libnginx-mod-http-auth-pam libnginx-mod-http-dav-ext  
  libnginx-mod-http-echo libnginx-mod-http-geoip  
  libnginx-mod-http-image-filter libnginx-mod-http-subst-filter  
  libnginx-mod-http-upstream-fair libnginx-mod-http-xslt-filter  
  libnginx-mod-mail libnginx-mod-stream libtiff5 libwebp6 libxpm4 libxslt1.1  
  nginx nginx-common nginx-full  
0 upgraded, 23 newly installed, 0 to remove and 0 not upgraded.  
Need to get 4555 kB of archives.  
After this operation, 10.1 MB of additional disk space will be used.  
Do you want to continue? [Y/n] y
```

After it's installed, Nginx should be automatically started. Check its status with `systemctl`.

```
systemctl status nginx
```

Sample output:

```
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: en
   Active: active (running) since Sat 2019-08-10 06:20:26 UTC; 54s ago
     Docs: man:nginx(8)
  Main PID: 19713 (nginx)
    Tasks: 2 (limit: 1149)
   Memory: 4.6M
   CGroup: /system.slice/nginx.service
           └─19713 nginx: master process /usr/sbin/nginx -g daemon on; master_pr
              └─19714 nginx: worker process
```

Hint: If the above command doesn't quit immediately, you can press Q key to gain back control of the terminal window.

If it's not running, use systemctl to start it.

```
sudo systemctl start nginx
```

It's also a good idea to enable Nginx to automatically start at boot time.

```
sudo systemctl enable nginx
```

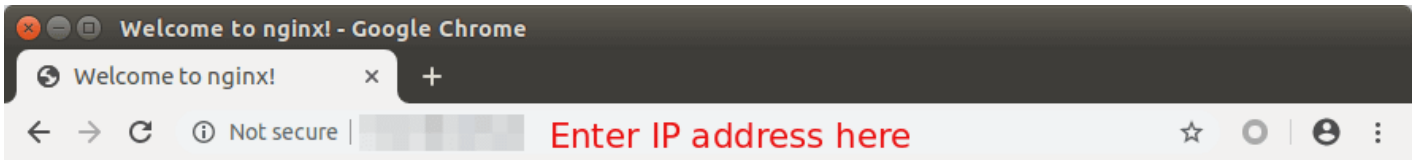
Check Nginx version:

```
sudo nginx -v
```

Output:

```
nginx version: nginx/1.14.2
```

Now type in the public IP address of your Debian 10 server in the browser address bar. You should see the default "Welcome to nginx" Web page, which means Nginx Web server is running properly. If you are installing LEMP on your local Debian 10 computer, then you should type `|127.0.0.1|` or `|localhost|` in the browser address bar.



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

If the connection is refused or failed to complete, there might be a firewall preventing incoming requests to TCP port 80. If you are using iptables firewall, then you need to run the following command to open TCP port 80.

```
sudo iptables -I INPUT -p tcp --dport 80 -j ACCEPT
```

If you are using [UFW firewall](#), then run this command to open TCP port 80.

```
sudo ufw allow http
```

Now we need to set `www-data` (Nginx user) as the owner of document root (also known as web root). By default it's owned by the root user. (Note that Nginx by default uses `/usr/share/nginx/html/` as web root, whereas Apache web server uses `/var/www/html/` as we root.)

```
sudo chown www-data:www-data /usr/share/nginx/html/ -R
```

Step 3: Install MariaDB Database Server on Debian 10

MariaDB is a drop-in replacement for MySQL. Enter the following command to install it on Debian 10.

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

After it's installed, MariaDB server should be automatically started. Use **systemctl** to check its status.

```
systemctl status mariadb
```

Output:

```
● mariadb.service - MariaDB 10.3.15 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset:
   Active: active (running) since Sat 2019-08-10 06:38:58 UTC; 13s ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
  Main PID: 20669 (mysqld)
    Status: "Taking your SQL requests now..."
     Tasks: 31 (limit: 1149)
    Memory: 77.7M
    CGroup: /system.slice/mariadb.service
            └─20669 /usr/sbin/mysqld
```

If it's not running, start it with this command:

```
sudo systemctl start mariadb
```

To enable MariaDB to automatically start at boot time, run

```
sudo systemctl enable mariadb
```

Now run the post installation security script.

```
sudo 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.

```
linuxbabe@buster: ~  
linuxbabe@buster:~$ sudo mysql_secure_installation  
  
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB  
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!  
  
In order to log into MariaDB to secure it, we'll need the current  
password for the root user. If you've just installed MariaDB, and  
you haven't set the root password yet, the password will be blank,  
so you should just press enter here.  
  
Enter current password for root (enter for none): Press Enter  
OK, successfully used password, moving on..  
  
Setting the root password ensures that nobody can log into the MariaDB  
root user without the proper authorisation.  
  
Set root password? [Y/n] y Enter y to set MariaDB root password  
New password:
```

Next, you can just press Enter to answer all remaining questions. This 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.)

```
linuxbabe@buster: ~  
By default, a MariaDB installation has an anonymous user, allowing anyone  
to log into MariaDB 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? [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!  
linuxbabe@buster:~$
```

By default, the MariaDB package on Debian uses `unix_socket` to authenticate user login, which basically means you can use username and password of the OS to log into MariaDB console. So you can run the following command to login without providing MariaDB root password.

```
sudo mariadb -u root
```

or

```
sudo mysql -u root
```

To exit, run

```
exit;
```

Check MariaDB server version information.

```
mariadb --version
```

Output:

```
mariadb Ver 15.1 Distrib 10.3.15-MariaDB, for debian-linux-gnu (x86_64) using readline 5.2
```

Step 4: Install PHP7.3 on Debian 10

At the the time of this writing, PHP7.3 is the latest stable version of PHP and has minor performance improvement over previous versions. Enter the following command to install PHP7.3 and some common PHP extensions from Debian 10 repository.

```
sudo apt install php7.3 php7.3-fpm php7.3-mysql php-common php7.3-cli php7.3-common php7.3-json php7.3-opcache php7.3-readline
```

Check PHP version information.

```
php --version
```

Output:

```
PHP 7.3.4-2 (cli) (built: Apr 13 2019 19:05:48) ( NTS )  
Copyright (c) 1997-2018 The PHP Group  
Zend Engine v3.3.4, Copyright (c) 1998-2018 Zend Technologies  
with Zend OPcache v7.3.4-2, Copyright (c) 1999-2018, by Zend Technologies
```

Now start php7.3-fpm.

```
sudo systemctl start php7.3-fpm
```

Enable auto-start at boot time.

```
sudo systemctl enable php7.3-fpm
```

Check status:

```
systemctl status php7.3-fpm
```

Step 5: Create a Nginx Server Block

A Nginx server block is like a virtual host in Apache. We will not use the default server block because it's inadequate to run PHP code and if we modify it, it becomes a mess. So remove the `default` symlink in `sites-enabled` directory by running the following command. (It's still available as `/etc/nginx/sites-available/default`.)

```
sudo rm /etc/nginx/sites-enabled/default
```

Then create a brand new server block file under `/etc/nginx/conf.d/` directory with a command line text editor, such as Nano.

```
sudo nano /etc/nginx/conf.d/default.conf
```

Paste the following text into the file. The following snippet will make Nginx listen on IPv4 port 80 and IPv6 port 80 with a catch-all server name.

```
server {
    listen 80;
    listen [::]:80;
    server_name _;
    root /usr/share/nginx/html;
    index index.php index.html index.htm index.nginx-debian.html;

    location / {
        try_files $uri $uri/ /index.php;
    }

    location ~ \.php$ {
        fastcgi_pass unix:/run/php/php7.3-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }
}
```

```
include fastcgi_params;
include snippets/fastcgi-php.conf;
}

# A long browser cache lifetime can speed up repeat visits to your page
location ~* \.(jpg|jpeg|gif|png|webp|svg|woff|woff2|ttf|css|js|ico|xml)$ {
    access_log      off;
    log_not_found   off;
    expires         360d;
}

# disable access to hidden files
location ~ /\.ht {
    access_log off;
    log_not_found off;
    deny all;
}
}
```

Save and close the file. (To save a file in Nano text editor, press `Ctrl+O`, then press Enter to confirm. To exit, press `Ctrl+X`.) Then test Nginx configurations.

```
sudo nginx -t
```

If the test is successful, reload Nginx.

```
sudo systemctl reload nginx
```

Step 6: Test PHP

To test PHP scripts with Nginx server, we need to create a `info.php` file in the Web root directory.

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

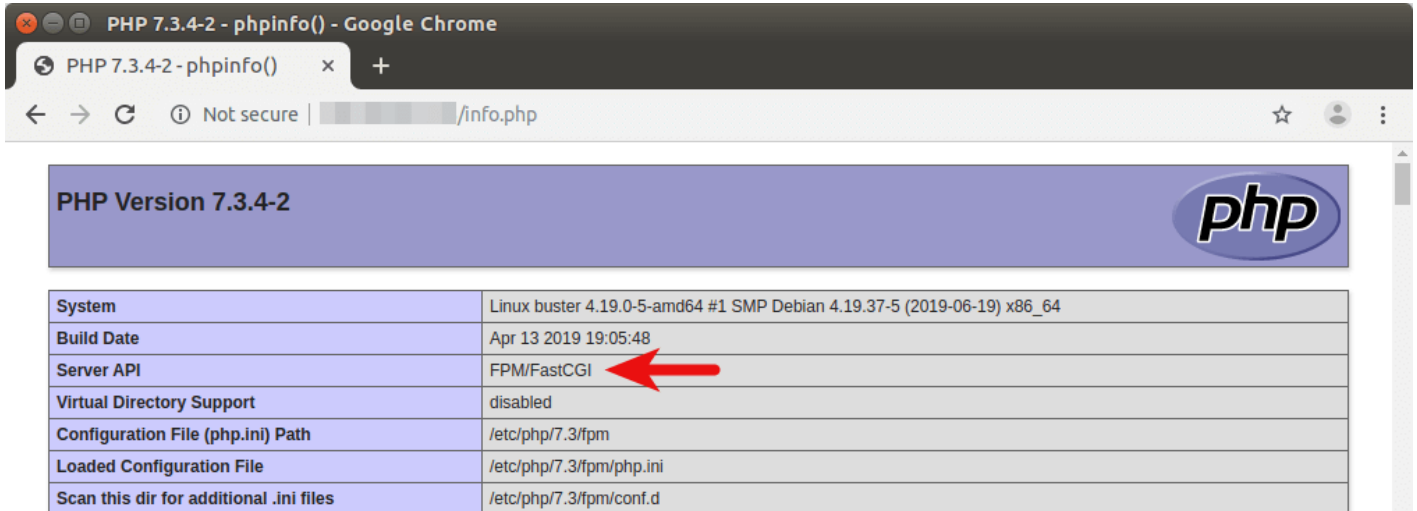
Paste the following PHP code into the file.

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

Save and close the file.

Now in the browser address bar, enter `server-ip-address/info.php`. Replace `server-ip-address` with your actual IP. If you follow this tutorial on your local computer, then type `127.0.0.1/info.php` or `localhost/info.php`.

You should see your server's PHP information. This means PHP scripts can run properly with Nginx web server. You can find that Zend OPcache is enabled.



System	Linux buster 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5 (2019-06-19) x86_64
Build Date	Apr 13 2019 19:05:48
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.3/fpm
Loaded Configuration File	/etc/php/7.3/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/7.3/fpm/conf.d

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 18:56:49 by Admin

Updated 11 July 2021 18:57:14 by Admin