How to Install LEMP Stack on Ubuntu 22.04 Server/Desktop

Last Updated: September 15th, 2022 Xiao Guoan (Admin) 22 Comments Ubuntu

This tutorial is going to show you how to install **LEMP stack** (Nginx, MariaDB, and PHP8.1) on **Ubuntu 22.04**. A software stack is a set of software tools bundled together. LEMP stands for Linux, Nginx (Engine-X), MariaDB/MySQL, and PHP, all of which are open source and free to use.

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.
- PHP is the server-side scripting language responsible for generating dynamic web pages.

Requirements

To follow this tutorial, you need an Ubuntu 22.04 OS running on your local computer or on a remote server.

If you are looking for a virtual private server (VPS), I recommend Kamatera VPS, which features:

- 30 days free trial.
- Starts at $4/month (1GB RAM)
- High-performance KVM-based VPS

- 9 data centers around the world, including United States, Canada, UK, Germany, The Netherlands, Hong Kong, and Isreal.

Follow the tutorial linked below to create your Linux VPS server at Kamatera.

- How to Create a Linux VPS Server on Kamatera

Once you have a VPS running Ubuntu 22.04, follow the instructions below.

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.

Step 1: Update Software Packages

Before we install the LEMP stack, it's a good practice to update the repository and software packages by running the following commands on your Ubuntu 22.04 OS.

```
sudo apt update

sudo apt upgrade -y
```

Step 2: Install Nginx Web Server

Nginx is a high-performance web server and is 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
```

After it's installed, we can enable Nginx to auto-start at boot time by running the following command.

```
sudo systemctl enable nginx
```

Then start Nginx with this command:

```
sudo systemctl start nginx
```

Now check out its status.

```
sudo systemctl status nginx
```

Output:

```
● nginx.service - A high performance w
eb server and a reverse proxy server
     Loaded: loaded (/lib/systemd/syst
em/nginx.service; enabled; vendor pres
et: enabled)
     Active: active (running) since Fr
i 2022-04-10 14:11:43 UTC; 3s ago
       Docs: man:nginx(8)
    Process: 8533 ExecStartPre=/usr/sb
in/nginx -t -q -g daemon on; master_pr
ocess on; (code=exited, status=0/SUCCE
SS)
    Process: 8545 ExecStart=/usr/sbin/
nginx -g daemon on; master_process on;
(code=exited, status=0/SUCCESS)
   Main PID: 8549 (nginx)
      Tasks: 3 (limit: 9451)
     Memory: 3.9M
     CGroup: /system.slice/nginx.servi
ce
```

```
             ├─8549 nginx: master proc
ess /usr/sbin/nginx -g daemon on; mast
er_process on;
             ├─8550 nginx: worker proc
ess
             └─8551 nginx: worker proc
ess
```

"**Enabled**" indicates that auto-start at boot time is enabled and we can see that Nginx is running. You can also see how much RAM Nginx is using from the output. If the above command doesn't immediately quit after running. You need to press "**q**" to make it quit.

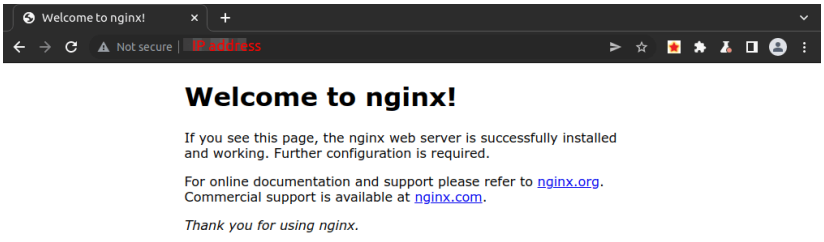
Check Nginx version.

```
nginx -v
```

Output:

```
nginx version: nginx/1.18.0 (Ubuntu)
```

Now type in the public IP address of your Ubuntu 22.04 server in the browser address bar. You should see the "Welcome to Nginx" Web page, which means Nginx Web server is running properly. If you are installing LEMP on your local Ubuntu 22.04 computer, then type `127.0.0.1` or `localhost` in the browser address bar.



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 the UFW firewall, then run this command to open TCP port 80.

```
sudo ufw allow http
```

Finally, we need to make `www-data` (Nginx user) as the owner of web directory. By default, it's owned by the root user.

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

Step 3: Install MariaDB Database Server

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 Ubuntu 22.04.

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

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

```
systemctl status mariadb
```

Output:

```
● mariadb.service - MariaDB 10.6.7 dat
abase server
     Loaded: loaded (/lib/systemd/syst
em/mariadb.service; enabled; vendor pr
eset: enabled)
     Active: active (running) since Fr
i 2022-04-10 14:19:16 UTC; 18s ago
       Docs: man:mysqld(8)
             https://mariadb.com/kb/e
n/library/systemd/
   Main PID: 9161 (mysqld)
     Status: "Taking your SQL requests
now..."
      Tasks: 31 (limit: 9451)
     Memory: 64.7M
     CGroup: /system.slice/mariadb.ser
vice
             └─9161 /usr/sbin/mysqld
```

> **Hint**: If the above command doesn't immediately quit after running. You need to press "**q**" to make it quit.

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

```
sudo systemctl start mariadb
```

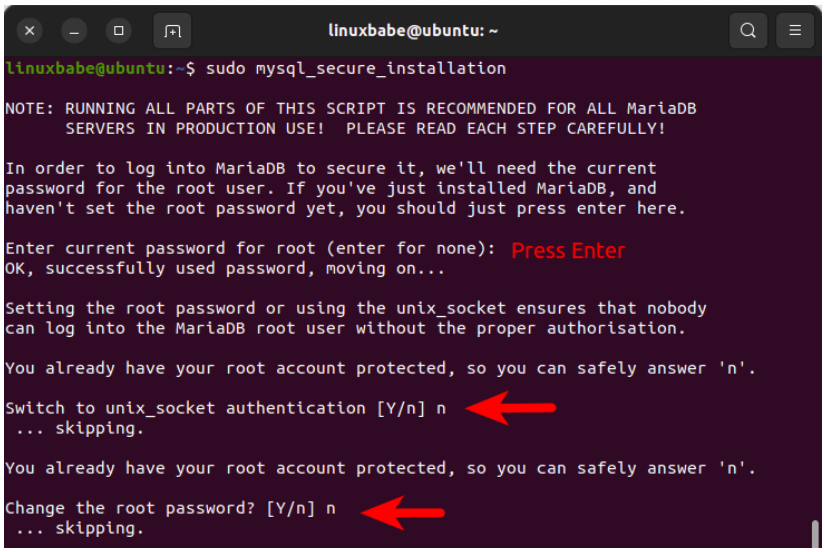
To enable MariaDB to automatically start at boot time, run

```
sudo systemctl enable mariadb
```

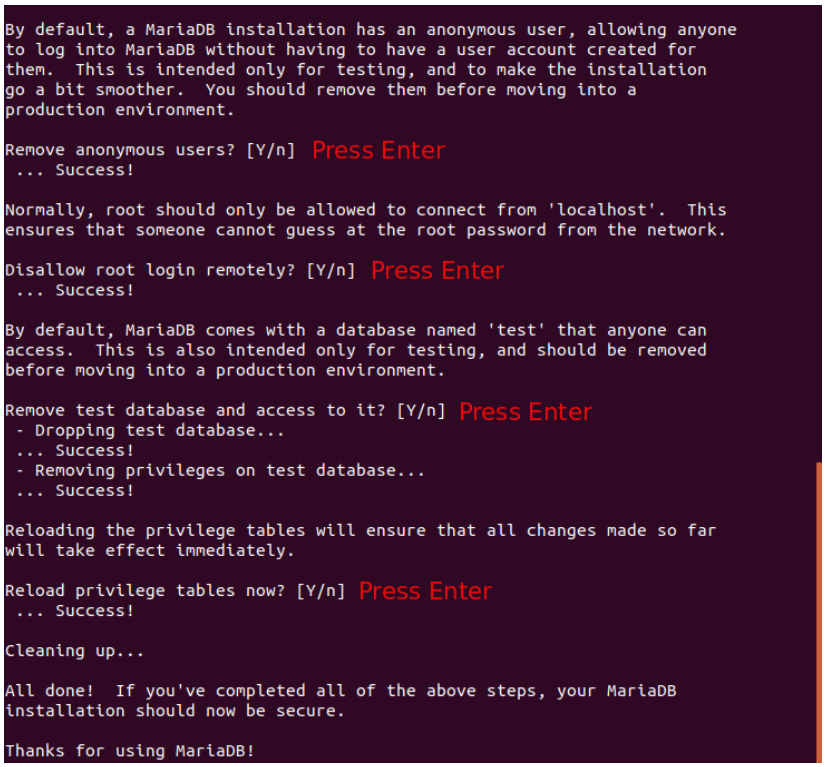
Now run the post-installation security script.

```
sudo mysql_secure_installation
```

- When it asks you to enter MariaDB root password, press `Enter` key as the root password isn't set yet.
- Don't switch to unix_socket authentication because MariaDB is already using unix_socket authentication.
- Don't change the root password, because you don't need to set root password when using unix_socket authentication.



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. (Notice that `Y` is capitalized, which means it is the default answer.)



By default, the MaraiDB package on Ubuntu 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 log in without providing MariaDB root password.

```
sudo mariadb -u root
```

To exit, run

```
exit;
```

Check MariaDB server version information.

```
mariadb --version
```

As you can see, we have installed MariaDB 10.6.7.

```
mariadb Ver 15.1 Distrib 10.6.7-MariaD
B, for debian-linux-gnu (x86_64) using
```

```
EditLine wrapper
```

Step 4: Install PHP8.1

PHP8.1 is included in Ubuntu 22.04 repository and has a minor performance improvement over PHP8.0. Enter the following command to install PHP8.1 and some common extensions.

```
sudo apt install php8.1 php8.1-fpm php
8.1-mysql php-common php8.1-cli php8.1
-common php8.1-opcache php8.1-readline
php8.1-mbstring php8.1-xml php8.1-gd p
hp8.1-curl
```

PHP extensions are commonly needed for content management systems (CMS) like WordPress. For example, if your installation lacks `php8.1-xml`, then some of your WordPress site pages may be blank and you can find an error in Nginx error log like:

```
PHP message: PHP Fatal error:  Uncaugh
t Error: Call to undefined function xm
l_parser_create()
```

Installing these PHP extensions ensures that your CMS runs smoothly. Now start php8.1-fpm.

```
sudo systemctl start php8.1-fpm
```

Enable auto-start at boot time.

```
sudo systemctl enable php8.1-fpm
```

Check status:

```
systemctl status php8.1-fpm
```

Sample output:

```
● php8.1-fpm.service - The PHP 8.1 Fas
tCGI Process Manager
     Loaded: loaded (/lib/systemd/syst
em/php8.1-fpm.service; enabled; vendor
pr>
     Active: active (running) since Fr
i 2022-04-10 14:40:26 UTC; 12s ago
       Docs: man:php-fpm8.1(8)
    Process: 21019 ExecStartPost=/usr/
lib/php/php-fpm-socket-helper install
 /ru>
   Main PID: 21012 (php-fpm8.1)
     Status: "Processes active: 0, idl
e: 2, Requests: 0, slow: 0, Traffic: 0
req>
      Tasks: 3 (limit: 9451)
     Memory: 9.4M
     CGroup: /system.slice/php8.1-fpm.
service
             ├─21012 php-fpm: master p
rocess (/etc/php/8.1/fpm/php-fpm.conf)
             ├─21017 php-fpm: pool www
             └─21018 php-fpm: pool www
```

If the above command doesn't immediately quit after running. You need to press "**q**" to make it quit.

Step 5: Create an Nginx Server Block

An 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/defau
lt
```

Then use a command-line text editor like Nano to create a new server block file under **/etc/nginx/conf.d/** directory.

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

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/php8.1-
fpm.sock;
    fastcgi_param SCRIPT_FILENAME $doc
ument_root$fastcgi_script_name;
```

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

 # A long browser cache lifetime can s
peed 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-FPM with Nginx Web server, we need to create a `info.php` file in the webroot directory.

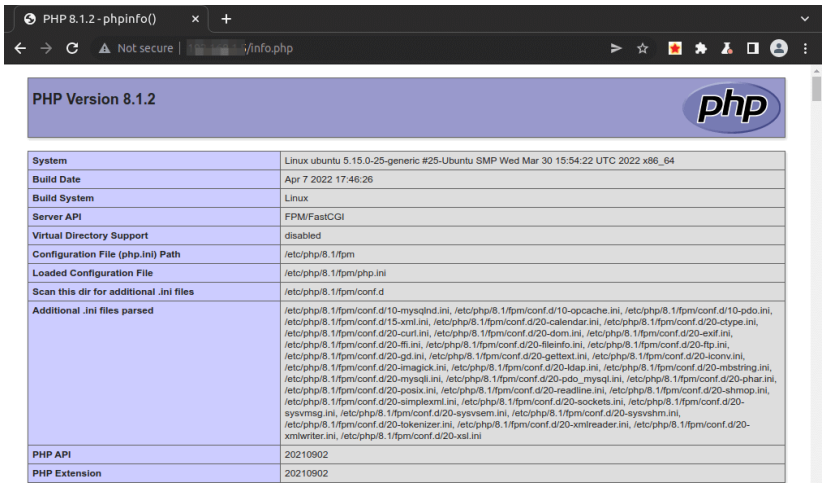
```
sudo nano /usr/share/nginx/html/info.php
```

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 `sever-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.



Step 7: Improve PHP Performance

The default PHP configurations (`/etc/php/8.1/fpm/php.ini`) are made for servers with very few resources (like a 256MB RAM server). To improve web application performance, you should change some of them.

We can edit the PHP config file (`php.ini`), but it's a good practice to create a custom PHP config file, so when you upgrade to a new version of PHP8.1, your custom configuration will be preserved.

```
sudo nano /etc/php/8.1/fpm/conf.d/60-c
ustom.ini
```

In this file, add the following lines.

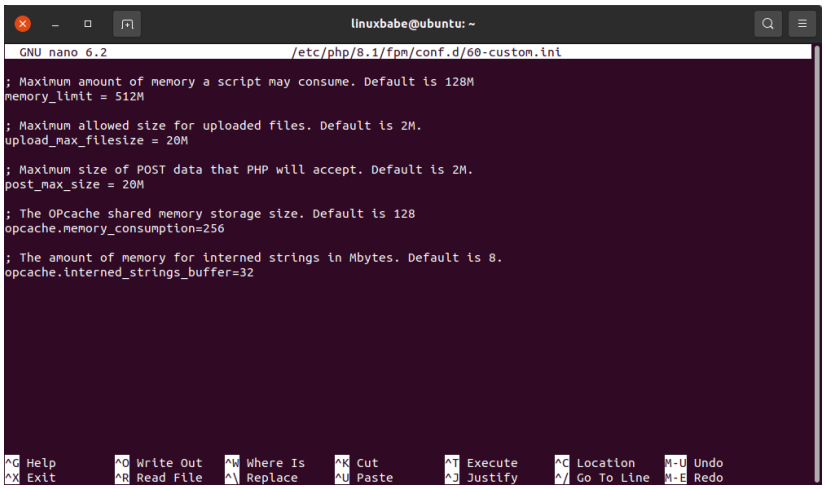
```
; Maximum amount of memory a script ma
y consume. Default is 128M
memory_limit = 512M

; Maximum allowed size for uploaded fi
les. Default is 2M.
upload_max_filesize = 20M

; Maximum size of POST data that PHP w
ill accept. Default is 2M.
post_max_size = 20M

; The OPcache shared memory storage si
ze. Default is 128
opcache.memory_consumption=256

; The amount of memory for interned st
rings in Mbytes. Default is 8.
opcache.interned_strings_buffer=32
```



Save and close the file. Then reload PHP8.1-FPM for the changes to take effect.

```
sudo systemctl reload php8.1-fpm
```

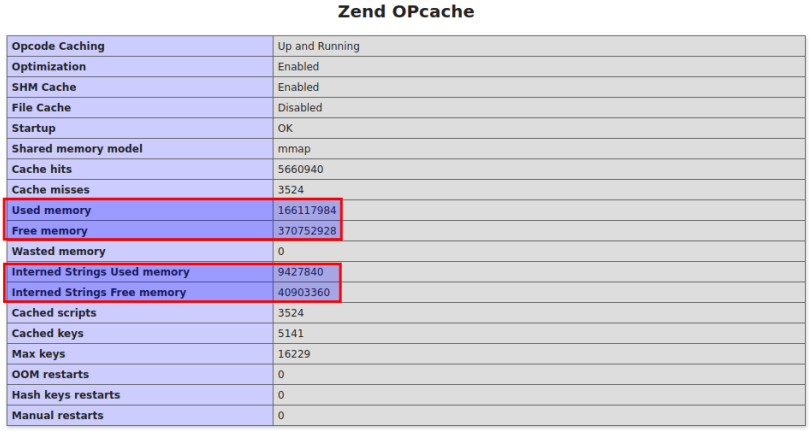
OPcache improves the performance of PHP applications by caching precompiled bytecode. You can view OPcache stats via the `info.php` page. Below is a before and after comparison on one of my servers.

Before

Zend OPcache

Opcode Caching	Up and Running
Optimization	Enabled
SHM Cache	Enabled
File Cache	Disabled
Startup	OK
Shared memory model	mmap
Cache hits	717842
Cache misses	3145
Used memory	103145752
Free memory	27546000
Wasted memory	3525976
Interned Strings Used memory	6291000
Interned Strings Free memory	8
Cached scripts	3111
Cached keys	4374
Max keys	16229
OOM restarts	3
Hash keys restarts	0
Manual restarts	4

After



Zend OPcache

Opcode Caching	Up and Running
Optimization	Enabled
SHM Cache	Enabled
File Cache	Disabled
Startup	OK
Shared memory model	mmap
Cache hits	5660940
Cache misses	3524
Used memory	166117984
Free memory	370752928
Wasted memory	0
Interned Strings Used memory	9427840
Interned Strings Free memory	40903360
Cached scripts	3524
Cached keys	5141
Max keys	16229
OOM restarts	0
Hash keys restarts	0
Manual restarts	0

As you can see, before applying the custom PHP configuration, the RAM allocated to OPcache is almost used up. After applying the custom PHP configurations, OPcache is able to use more RAM for caching precompiled bytecode.

Congrats! You have successfully installed Nginx, MariaDB, and PHP8.1 on Ubuntu 22.04. For your server's security, you should delete `info.php` file now to prevent hackers from seeing it.

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

Troubleshooting Tip

If you encounter errors, you can check the Nginx error log (`/var/log/nginx/error.log`) to find out what's wrong.

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/ngin
x.service.d/
```

Then create a file under this directory.

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

Add the following lines in the file, which will make Nginx automatically restart **5 seconds** after a failure is detected. The default value of `RetartSec` is **100ms**, which is too small. Nginx may complain that "start request repeated too quickly" if `RestartSec` is not big enough.

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

Save and close the file. Then reload systemd for the changes to take effect.

```
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
```

MariaDB Automatic Start

By default, MariaDB is configured to automatically restart `on-abort` (`/lib/systemd/system/mariadb.service`). However, if your server runs out of memory (oom) and MariaDB is killed by the oom killer, it won't automatically restart. We can configure it to restart no matter what happens.

Create a directory to store custom configurations.

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

Create a custom config file.

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

Add the following lines in the file.

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

Save and close the file. Then reload systemd for the changes to take effect.

```
sudo systemctl daemon-reload
```

Next Steps

As always, if you found this post useful, then subscribe to our free newsletter to get more tips and tricks. You can also install

WordPress on top of the LEMP stack to create your own website or blog.

- Install WordPress on Ubuntu 22.04 with Nginx, MariaDB, PHP8.1 (LEMP)

Related Nginx tutorials:

- How to Fix Common Nginx Web Server Errors

Backup is important in case of hacking, data center disasters, etc. You should have a backup strategy for your server.

- Back Up and Restore MariaDB Databases From the Command line
- Use Duplicati to Back Up Files on Debian, Ubuntu, Linux Mint

Linux Server Performance Tuning and Monitoring

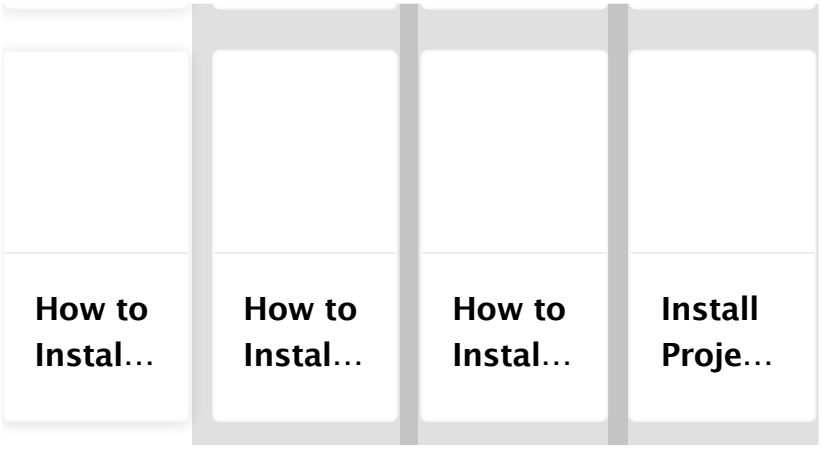
- Easily Boost Ubuntu Network Performance by Enabling TCP BBR
- What is HTTP/2 and How to Enable it on Nginx
- Linux Server Performance Monitoring with Netdata (2022)

Take care 🙂

Rate this tutorial

[Total: 79 Average: 4.7]

You may also like:


How to Instal…
How to Instal…
How to Instal…
Install Proje…