How To Install And Configure Nginx on Ubuntu 20.04 LTS

Nginxpronounced as **Engine X**is an open-source **popular web server**used to host websites and applications. NGINX is a high-performance load balancer, web server, and also used as reverse-proxy for other servers including Apache Web Server, Apache Tomcat, and NodeJS Applications. It's also considered to handle websites with high traffic due to its load-balancing capabilities. This tutorial provides the steps required to install Nginx on the popular Linux distribution Ubuntu. It provides all the steps required to install and use Nginx on Ubuntu 20.04 LTS. The steps should be similar for other Linux systems and Ubuntu versions.

Prerequisites

This tutorial assumes that you have already installed Ubuntu 20.04 LTS desktop or server version either for local or production usage. You can follow Install Ubuntu 20.04 LTS Desktop, Install Ubuntu 20.04 LTS On Windows Using VMware, and Spin Up Ubuntu 20.04 LTS Server On Amazon EC2to install Ubuntu 20.04 LTS. It also assumes that you have either root privileges or a regular user with sudo privileges.
In case you are installing **NGINX**for production usage, it assumes that the ports 80 and 443 are publicly open. On AWS EC2, we can explicitly open these ports by updating the security group associated with the servers. Similar to EC2, DigitalOcean (Cloud Firewalls) and UpCloud (L3 firewall) also provide firewall interface at an additional cost. On other cloud service providers, you may use the standard firewall i.e. UFW (ships by default with Ubuntu) to secure the server. In such cases, enable the appropriate apache profile based on your production usage.

Install Nginx - Default

We can install Nginx directly from the Ubuntu repositories using the commands as shown below.

```
# Refresh packages index
sudo apt update

# Install nginx
sudo apt install nginx

# Installation output
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
libjpeg-turbo8 libjpeg8 libnginx-mod-http-image-filter
libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream
libtiff5 libwebp6 libxpm4 nginx-common nginx-core
Suggested packages:
libgd-tools fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjbig0
libjpeg-turbo8 libjpeg8 libnginx-mod-http-image-filter
libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream
libtiff5 libwebp6 libxpm4 nginx nginx-common nginx-core
...
...
Setting up libnginx-mod-stream (1.17.10-0ubuntu1) ...
Setting up libtiff5:amd64 (4.1.0+git191117-2build1) ...
Setting up libfontconfig1:amd64 (2.13.1-2ubuntu3) ...
Setting up libgd3:amd64 (2.2.5-5.2ubuntu2) ...
Setting up libnginx-mod-http-image-filter (1.17.10-0ubuntu1) ...
Setting up nginx-core (1.17.10-0ubuntu1) ...
Setting up nginx (1.17.10-0ubuntu1) ...
Processing triggers for ufw (0.36-6) ...
Processing triggers for systemd (245.4-4ubuntu3) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9) ...
```

If required, enable Nginx to auto start on system boot and check it's status as shown below.

```
# Check and enable Nginx - Auto start
sudo systemctl is-enabled nginx
sudo systemctl enable nginx

# Output
Synchronizing state of nginx.service with SysV service script with
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx

# Check Nginx Status
sudo systemctl status nginx

# Output
```

```
● nginx.service - A high performance web server and a reverse proxy
server
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor
preset: enabled)
Active: active (running) 2020-06-06 11:20:01 UTC; 1min ago
Docs: man:nginx(8)
Main PID: 2669 (nginx)
Tasks: 3 (limit: 1119)
Memory: 4.0M
CGroup: /system.slice/nginx.service
----
----
```

Now verify the installation by checking the Nginx version as shown below.

```
# Check Nginx Version
nginx -v

# Output
nginx version: nginx/1.17.10 (Ubuntu)
```

The above-mentioned commands will install the Nginx server and its dependencies. It installed **Nginx 1.17.10**on **Ubuntu 20.04 LTS**while writing this tutorial. It may vary based on your system. The important configurations and directories of Nginx installed from the Ubuntu repositories are listed below.

- **Configurations Directory**- /etc/nginx/conf.d - Additional server configurations. It's empty by default.
- **Modules Available Directory**- /etc/nginx/modules-available - The directory to store the modules configurations.
- **Modules Enabled Directory**- /etc/nginx/modules-available - The directory the symlinks of the modules configurations. The symlinks of image filter, xslt filter, mail, and stream modules are available by default.
- **Sites Available Directory**- /etc/nginx/sites-available - The directory to store the server blocks. The default server block is available within this directory.
- **Sites Enabled Directory**- /etc/nginx/sites-enabled - The directory having the symlinks of the server blocks available at /etc/nginx/sites-available. We can enable or disabled the server blocks available at /etc/nginx/sites-available by creating or deleting the symlinks. The symlink of **default**server block is available within this directory.
- **Main Configuration**- /etc/nginx/nginx.conf - The main configuration of Nginx. View it on GitHub.
- **Default Server Block**- /etc/nginx/sites-available/default - The default server block. View it on GitHub.

We can also install the most recent version of Nginx as explained in the next section.

Install Nginx - Latest

We can also install the most recent version of Nginx using the official Nginx repository. You can remove or uninstall the existing installation of NGINX completely using the commands as shown below. Make sure to take the backup of existing configuration files in case you have used Nginx to host the applications.

```
# Backup main configuration
sudo cp /etc/nginx/nginx.conf /<backups path>/nginx.conf.bak

# Backup server blocks

# Completely Uninstall Nginx
sudo apt purge nginx nginx-full nginx-common nginx-core

# Autoclean
sudo apt-get autoclean

# Autoremove
sudo apt-get autoremove
```

Now create the repository source file using the commands as shown below. I have used the nano editor to create and save the file. You can use any editor of your choice.

```
# Create the repository source file
sudo nano /etc/apt/sources.list.d/nginx.list

# Update the file
deb https://nginx.org/packages/mainline/ubuntu/ focal nginx
deb-src https://nginx.org/packages/mainline/ubuntu/ focal nginx

# Save and close the editor
```

Save the file using the Nano text editor by pressing **CTRL + O**, then press **Enter**to write the file. Press **CTRL + X**to close the editor. Now verify the package integrity using the commands as shown below.

```
# Get the signing key
wget http://nginx.org/keys/nginx_signing.key

# Add the key
sudo apt-key add nginx_signing.key

# Output
```

```
OK
```

The above commands confirm the addition of the official Nginx repository to the system. Now refresh the packages index and install the latest Nginx using the commands as shown below.

```
# Refresh packages index
sudo apt update

# Install nginx
sudo apt install nginx

# Installation output
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
nginx
....
....
Unpacking nginx (1.19.0-1~focal) ...
Setting up nginx (1.19.0-1~focal) ...
Created symlink /etc/systemd/system/multi-
user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3) ...
```

Now verify the installation by checking the Nginx version as shown below.

```
# Check Nginx Version
nginx -v

# Output
nginx version: nginx/1.19.0
```

It installed **Nginx 1.19.0**on **Ubuntu 20.04 LTS**. The important configurations and directories of Nginx installed from the Nginx repositories are listed below.

- **Configurations Directory**- /etc/nginx/conf.d - Additional server configurations. The default server block is available within this directory.
- **Modules Directory**- It's a symlink to /usr/lib/nginx/modules.
- **Main Configuration**- /etc/nginx/nginx.conf - The main configuration of Nginx. View it on GitHub.
- **Default Server Block**- /etc/nginx/conf.d/default.conf - The default server block. View it on GitHub.

We can see that there are several differences on comparing the Nginx installed from the Ubuntu repositories. The Nginx installed from the Nginx repositories does not have configuration directories including modules available, modules enabled, sites available, sites enabled. Also the default server block is available as configuration within the /etc/nginx/conf.d directory.

Server Blocks and Server Tokens

This section provides the configurations to read the additional configurations from /etc/nginx/conf.d/ and active server blocks from /etc/nginx/sites-enabled/. We can also include server block files by updating the main configuration as shown below.

```
# Include server block files
sudo nano /etc/nginx/nginx.conf

# Scroll down and make sure that the below mentioned lines are there
in the http block
include /etc/nginx/conf.d/*.conf;
include /etc/nginx/sites-enabled/*;
```

Turn off the server token for production usage.

```
# Update config
sudo nano /etc/nginx/nginx.conf

# Update/Add server tokens config - http block
server_tokens off;

# Save and exit the editor
```

Install Certbot Plugin

Optionally, install the Certbox plugin required to generate Let's Encrypt SSL certificates. We can use the Certbox to generate and install SSL certificates to secure the website or application.

```
# Install Certbot Plugin
sudo apt install python3-certbot-nginx
```

You can follow Configure Virtual Host Or Server Block On Nginxand How To Install Let's Encrypt For Nginx On Ubuntuto learn more about configuring Server Blocks and generate Let's Encrypt SSL certificates.

Verify Nginx

We can verify the Nginx installed by us in the previous sections using the commands as shown below.

```
# Check version
# Ubuntu repositories
nginx version: nginx/1.17.10 (Ubuntu)

# Nginx repository
nginx version: nginx/1.19.0

# Detailed version details
nginx -V

# Output
nginx version: nginx/1.19.0
built by gcc 9.3.0 (Ubuntu 9.3.0-10ubuntu2)
built with OpenSSL 1.1.1f 31 Mar 2020
TLS SNI support enabled
....
....
```

Now enable Nginx to auto start on system boot, check the configuration file, and check the status of Nginx using the command as shown below.

```
# Check and enable Nginx - Auto start
sudo systemctl is-enabled nginx
sudo systemctl enable nginx

# Check configuration
sudo nginx -t

# Output
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful


# Check status
systemctl status nginx

# It will show status
● nginx.service - nginx - high performance web server
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor
preset: enabled)
Active: inactive (dead)
Docs: http://nginx.org/en/docs/
....
....
```

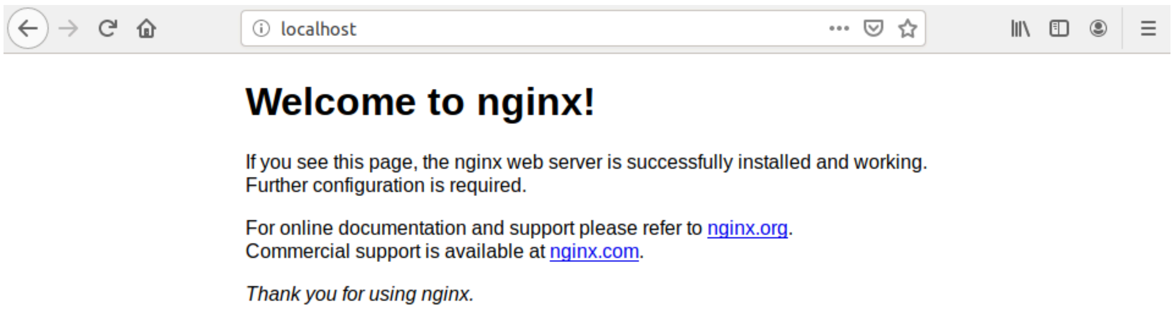
We can see that the Nginx is installed successfully and the configuration is ok.

```
# Start Nginx
sudo systemctl start nginx

# Again check Status
systemctl status nginx

# NGINX status
● nginx.service - nginx - high performance web server
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor
preset: enabled)
Active: active (running) since Sat 2020-06-06 11:49:02 UTC; 3s ago
Docs: http://nginx.org/en/docs/
----
----
```

The default www directory of Nginx is **/usr/share/nginx/html**. It serve the static files from this directory using the default server block.
The default www directory used by the NGINX Web Server is **/var/www/html**. Now open the NGINX's Home Page in your favorite browser by using the server's public IP address (**http://xx.xx.xx.xx**) or the DNS name assigned by the service provider. It should show the Home Page similar to Fig 1. It can also be accessed using **http://localhost**or**http://127.0.01**in case it's installed on a local desktop or server system.



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.

Fig 1
In case you are unable to see the page, as shown above, make sure that your firewall is configured properly and port 80 is available for Nginx.

Configure Firewall (Production Usage)

If your cloud service provider provides a firewall interface to configure the firewall, and you have configured your server to use the firewall interface provided by the service provider, open the ports 80 or 443 or both based on your server requirements. In case you are not using the firewall interface provided by the service provider and using UFW on your system, you can enable the appropriate profile as shown below.

Notes: It's mandatory to assign Security Group to the AWS EC2 instances and we do not need to configure UFW for EC2 instances for most of the use cases.

```
# UFW NGINX Profiles
sudo ufw app list

# Output
Available applications:
Nginx Full
Nginx HTTP
Nginx HTTPS
OpenSSH
```

The NGINX profiles can be used to configure UFW as listed below.
Nginx Full Profile- It opens port 80 and port 443 to allow unencrypted and TLS/SSL encrypted traffic.
NginxHTTP Profile- It only opens port 80 to allow unencrypted traffic.
NginxHTTPS Profile- It only opens port 443 to allow TLS/SSL encrypted traffic.
Now activate the Nginx Profile to open the ports 80 and 443 as shown below.

```
# Activate NGINX Full Profile
sudo ufw allow 'Nginx Full'

# Check NGINX Profile
sudo ufw status

# Output
Status: active

To Action From
-- ------ ----
OpenSSH ALLOW Anywhere
Nginx Full ALLOW Anywhere
OpenSSH (v6) ALLOW Anywhere (v6)
Nginx Full (v6) ALLOW Anywhere (v6)
```

We can see that the **Nginx Full Profile**is activated and open to accept connections from anywhere. If you are sure about your usage, enable the Nginx HTTPS profile to only accept the HTTPS request on port 443.

Common Commands

This section lists the commonly used commands to manage the Nginx process. These commands are listed below.

```
# Nginx Status
```

```
sudo systemctl status nginx

# Start Nginx
sudo systemctl start nginx

# Check status
sudo systemctl status nginx

# Status Output after starting the server
● nginx.service - nginx - high performance web server
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor
preset: enabled)
Active: active (running) since Sat 2020-06-06 11:49:02 IST; 31min ago
Docs: http://nginx.org/en/docs/
Process: 21131 ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf
(code=exited, status=0/SUCCESS)
Main PID: 21141 (nginx)
Tasks: 2 (limit: 4624)
Memory: 2.1M
CGroup: /system.slice/nginx.service
├─21141 nginx: master process /usr/sbin/nginx -c
/etc/nginx/nginx.conf
└─21142 nginx: worker process
...
...

# Stop Nginx
sudo systemctl stop nginx

# Reload Nginx
sudo systemctl reload nginx

# Enable Nginx on system boot
sudo systemctl enable nginx

# Disable Nginx on system boot
sudo systemctl disable nginx
```

Disable Default Server Block

This step is optional and you can follow it in case you want to disable the default server block which shows the **Default Page**as shown in Fig 1. It can be done as shown below.

```
# Disable default configs
sudo nano /etc/nginx/nginx.conf
```

```
# Scroll down and update the http block
....
....
keepalive_timeout 65;

#gzip on;

#include /etc/nginx/conf.d/*.conf;
include /etc/nginx/sites-enabled/*;
}

# Save and exit the editor

# Optional - Only if sites-available and sites-enabled directories
exist
# Remove default config symlink from sites-enabled
sudo rm /etc/nginx/sites-enabled/default

# Reload Nginx
sudo systemctl reload nginx
```

After reloading NGINX, the default server block gets disabled. We can also delete the default config file in case the configurations directory **/etc/nginx/conf.d**is required for other configs except for server blocks.

```
# Backup the file - /etc/nginx/conf.d/default.conf
sudo mkdir /etc/nginx/sites-available
sudo mv /etc/nginx/conf.d/default.conf /etc/nginx/sites-
available/default

# Enable default configs
sudo nano /etc/nginx/nginx.conf

# Scroll down and update the http block
....
....
keepalive_timeout 65;

#gzip on;

include /etc/nginx/conf.d/*.conf;
include /etc/nginx/sites-enabled/*;
}
```

```
# Save and exit the editor

# Remove default server block - if exists
sudo rm /etc/nginx/conf.d/default.conf

# Optional - Only if sites-available and sites-enabled directories
exist
# Remove default config symlink from sites-enabled
sudo rm /etc/nginx/sites-enabled/default

# Reload Nginx
sudo systemctl reload nginx
```

In this way, we can keep the configurations within the directory **/etc/nginx/conf.d/**instead of completely disabling it. Now if you try to open the default URL in the browser, it will show an error message - **Unable to connect**.

Add and Configure Server Block

This section explains about adding the server block and configuring it separately for each site. We can start the site configuration using the default config as shown below. I have used the domain **example.com**for the examples. Make sure to replace it with your own domain.

```
# Create the directories if does not exist
sudo mkdir /etc/nginx/sites-available
sudo mkdir /etc/nginx/sites-enabled

# Copy the default config
sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-
available/example.com
# OR
sudo cp /etc/nginx/conf.d/default.conf /etc/nginx/sites-
available/example.com
```

Now update the server block to configure your domain and the system path having site files. The configuration should look like the one as shown below.

```
# Update server block
sudo nano /etc/nginx/sites-available/example.com

# Content
server {
listen 80;
server_name example.com www.example.com;
#charset koi8-r;
#access_log /var/log/nginx/host.access.log main;

location / {
root /var/www/example.com/html;
index index.html index.htm;
}

#error_page 404 /404.html;

# redirect server error pages to the static page /50x.html
#
error_page 500 502 503 504 /50x.html;
location = /50x.html {
root /usr/share/nginx/html;
}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80
#
#location ~ \.php$ {
# proxy_pass http://127.0.0.1;
#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
#
#location ~ \.php$ {
# root html;
# fastcgi_pass 127.0.0.1:9000;
# fastcgi_index index.php;
# fastcgi_param SCRIPT_FILENAME /scripts$fastcgi_script_name;
# include fastcgi_params;
#}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
#location ~ /\.ht {
# deny all;
#}
}

# Save and exit the editor
# Create the site directory and update permissions
sudo mkdir /var/www/example.com/html
sudo chmod -R 755 /var/www/example.com
```

Use the below-mentioned commands to enable or disable the site configuration.

```
# Enable Site - Create the symlink
sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-
enabled/

# Disable Site
sudo rm /etc/nginx/sites-enabled/example.com
# OR
sudo unlink /etc/nginx/sites-enabled/example.com
```

Apply the configuration using the below-mentioned commands.

```
# Check configurations
sudo nginx -t

# Reload configuration
sudo nginx -s reload
# OR
sudo systemctl reload nginx
```

Now add the **index.html**file having content as shown below.

```
# Add index.html
sudo nano /var/www/example.com/html/index.html

# Content
<!DOCTYPE html>
<html lang="en">
<head>
<title>My Domain</title>
</head>
<body>
<h1>Welcome to My Domain.</h1>
</body>
</html>

# Save and exit the editor
```

This is all about configuring the site. You may fine-tune the configuration by adding error pages, SSL certificate, etc based on your site needs.

Hash Bucket Memory Problem

You may face the hash bucket memory problem in case you are using server blocks. It can be avoided by updating the configuration file as shown below.

```
# Update config
sudo nano /etc/nginx/nginx.conf

# Update the file

...
http {
...
server_names_hash_bucket_size 64;
...
}
...
```

Apply the configuration using the below-mentioned commands.

```
# Check configurations
sudo nginx -t

# Restart nginx
sudo systemctl restart nginx
```

Now open your site using the URL - **http://example.com**or**http://www.example.com**. It should show your site. Make sure that the **domain record**is configured properly to use your server. The domain should point to the IP address of your server.

Summary

This tutorial provided all the steps required to install and configure NGINX on Ubuntu 20.04 LTS. It also provided to configure the server blocks and the basic commands required to start, stop, or reload NGINX.