How to configure static ip address

# sudo nano /etc/netplan/50-cloud-init.yaml

network:

ethernets:

enp0s3:

addresses:

- 192.168.8.5/24

gateway4: 192.168.8.1

nameservers:

addresses:

- 8.8.8.8

- 8.8.4.4

version: 2

**How to change datetime**

#timedatectl list-timezones | grep -i Asia/Kuala\_Lumpur

#sudo ln -s /usr/share/zoneinfo/Asia/Kuala\_Lumpur /etc/localtime

How to install cron job

#sudo apt update

Then install cron with the following command:

#sudo apt install cron

You’ll need to make sure it’s set to run in the background too:

#sudo systemctl enable cron

**Install Apache, MySQL, PHP (LAMP) Stack On Ubuntu 18.04 LTS**

https://www.ostechnix.com/install-phpmyadmin-with-lamp-stack-on-ubuntu-18-04-lts/

#### Install Apache web server

First of all, update Ubuntu server using commands:

$ sudo apt update

$ sudo apt upgrade

Next, install Apache web server:

$ sudo apt install apache2

Check if Apache web server is running or not:

$ sudo systemctl status apache2

###### 1.1 Adjust firewall to allow Apache web server

$sudo ufw app list

As you see, “Apache Full” profile has included the rules to enable traffic to the ports **80** and **443**:

Now, run the following command to allow incoming HTTP and HTTPS traffic for this profile:

**$sudo ufw app info "Apache Full"**

**# sudo ufw allow 3306**

###### 1.2 Test Apache Web server

Now, open your web browser and access Apache test page by navigating to **http://localhost/** or **http://IP-Address/**.

#### 2. Install MySQL

To install MySQL On Ubuntu, run:

$sudo apt install mysql-server

Verify if MySQL service is running or not using command:

$ sudo systemctl status mysql

###### 2.1 Setup database administrative user (root) password

By default, MySQL **root** user password is blank. You need to secure your MySQL server by running the following script:

$ sudo mysql\_secure\_installation

###### 2.2 Change authentication method for MySQL root user

$sudo mysql

Run the following command at the mysql prompt to find the current authentication method for all mysql user accounts:

SELECT user,authentication\_string,plugin,host FROM mysql.user;

As you see, mysql root user uses **auth\_socket** plugin for authentication.

To change this authentication to **mysql\_native\_password** method, run the following command at mysql prompt. Don’t forget to replace **“password”** with a strong and unique password of your choice. If you have enabled VALIDATION plugin, make sure you have used a strong password based on the current policy requirements.

Mysql > ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'password';

Update the changes using command:

Mysql> FLUSH PRIVILEGES;

Now check again if the authentication method is changed or not using command:

Mysql> SELECT user,authentication\_string,plugin,host FROM mysql.user;

#### Install PHP

To install PHP, run:

$ sudo apt install php libapache2-mod-php php-mysql

After installing PHP, create **info.php** file in the Apache root document folder. Usually, the apache root document folder will be **/var/www/html/** or **/var/www/** in most Debian based Linux distributions. In Ubuntu 18.04 LTS, it is **/var/www/html/**.

Let us create **info.php** file in the apache root folder:

$ sudo vi /var/www/html/info.php

Add the following lines:

<?php

phpinfo();

?>

$ sudo systemctl restart apache2

###### 3.1 Test PHP

Open up your web browser and navigate to **http://IP-address/info.php** URL.

Usually, when a user requests a directory from the web server, Apache will first look for a file named **index.html**. If you want to change Apache to serve php files rather than others, move **index.php** to first position in the **dir.conf** file as shown below

Move the “index.php” file to first. Once you made the changes, your **dir.conf** file will look like below.

<IfModule mod\_dir.c>

DirectoryIndex **index.php** index.html index.cgi index.pl index.xhtml index.htm

</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

###### 3.2 Install PHP modules

To improve the functionality of PHP, you can install some additional PHP modules.

To list the available PHP modules, run:

$ sudo apt-cache search php- | less

To find the details of any particular php module, for example **php-gd**, run:

$ sudo apt-cache show php-gd

To install a php module run:

$ sudo apt install php-gd

To install all modules (not necessary though), run:

$ sudo apt-get install php\*

## Install phpMyAdmin With LAMP Stack On Ubuntu 18.04 LTS

$ sudo add-apt-repository universe

Now is the time to install phpMyAdmin.

To install phpMyAdmin on Ubuntu 18.04, run:

$ sudo apt update

$ sudo apt install phpmyadmin php-mbstring php-gettext

Now, select the web server that should be automatically configured to run phpMyAdmin. Choose the web server with UP/DOWN arrows and hit the SPACEBAR key. Once you chose the web server, you will see a \* (star) symbol in-front of it. Hit the TAB key to choose OK and again hit ENTER key to continue.

$ sudo phpenmod mbstring

$ sudo systemctl restart apache2

$ sudo phpenmod mbstring

$ sudo systemctl restart apache2

#### Create dedicated user to access phpMyAdmin dashboard

Once phpMyAdmin is installed, a database user named **‘phpmyadmin’** will be automatically created with the administrative password you set during the installation. You can login to phpmyAdmin dashboard using ‘phpmyadmin’ user or mysql **root** user. However, it is recommended to create a dedicated user to manage databases via phpMyAdmin web interface.

To do so, login to mysql shell using command:

$ mysql -u root -p

Enter your mysql root password. You will now be in mysql shell.

Enter the following command to create a new dedicated user for phpmyadmin:

CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY 'password';

Here, **phpmyadmin** is the new user for accessing phpmyadmin dashboard. The password for phpmyadminuser is **password**. Replace these values with your own.

Next give the appropriate privileges to the ‘phpmyadminuser’ using command:

GRANT ALL PRIVILEGES ON \*.\* TO 'phpmyadminuser'@'localhost' WITH GRANT OPTION;

Finally exit from mysql shell:

exit

**Copy the file on the directory** /var/www/html/ using winscp

Then assign the ownership of the directory through the following commands:

sudo chown -R $USER:$USER /var/www/html/busbookingtrial

sudo chmod -R 755 /var/www/html/busbookingtrial.com

**How to create backup mysql database and creating cronjob**

**$ sudo mkdir /usr/local/bin/dbasebackup**

**$ sudo nano /usr/local/bin/dbasebackup/backup.sh**

**Then insert the following code**

**#!/bin/sh**

**#echo "Stopping all Cronjob Backup Procedure"**

**#systemctl stop crond.service**

**mysqldump -u root -pp@ssw0rd busbooking> "/usr/local/bin/dbasebackup/busbooking”**

**echo "Succesfully backup"**

**#systemctl start crond.service**

**#echo "Cronjob Sucessfully started"**

**How to create executable**

**$nano chmod u+x /usr/local/bin/dbasebackup/backup.sh**

**How to create cronjob**

**# crontab –e**

**Then insert**

\* \* \* \* \* cd /usr/local/bin/dbasebackup; ./backup.sh

59 7 \* \* \* rsync -avz -e ssh /usr/local/bin/dbasebackup/ [root@192.168.1.8:/root/mysql\_backup/](mailto:root@192.168.1.8:/root/mysql_backup/)

**=============================================================================**

## Set Up Virtual Hosts in Apache

A virtual host is similar to what you have server blocks in Nginx. It is used to manage configurations for more than one domain from one server. We will present an example of how to set up a virtual host through the Apache server. We will set up a website named sampledomain.com by using the server block that is enabled by default in Apache for Ubuntu 18.

### Step 1: Set up a domain name

The server block that is enabled by default is capable of serving documents from /var/www/html. However, we will create a directory at /var/www/ leaving the default directory intact.

Create this directory through the following command, replacing sampledomain.com by your respective domain name.

sudo mkdir -p /var/www/busbookingtrial/

Then assign the ownership of the directory through the following commands:

sudo chown -R $USER:$USER /var/www/html/busbookingtrial

sudo chmod -R 755 /var/www/html/busbookingtrial.com

Let us now create an index page that we can later access to test if Apache is running our domain name. Create an HTML file either through the Nano editor or any of your favorite text editor.

$ nano /var/www/busbookingtrial.com/index.html

Enter the following HTML for the index page:

<html>

<head>

<title>Welcome to the page sampledomain.com!</title>

</head>

<body>

<h1>You got Lucky! Your sampledomain.com server block is up!</h1>

</body>

</html>

We are using the nano editor to create the HTML file.

You can save a file in nano by using Ctrl+X and then enter Y and hitting Enter.

Apache needs a virtual host file to serve the contents of your server. The default configuration file for this purpose is already created but we will make a new one for our custom configurations.

$ sudo nano /etc/apache2/sites-available/busbookingtrial.com.conf

Enter the following customized configuration details for our domain name:

<VirtualHost \*:80>

ServerAdmin admin@busbookingtrial.com

ServerName busbookingtrial.com

ServerAlias www. busbookingtrial.com

DocumentRoot /var/www/busbookingtrial

ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

</VirtualHost>

We are using the nano editor to create this .conf file.

You can save a file in nano by using ***Ctrl+X*** and then enter Y and hitting Enter.

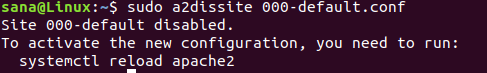
### Step 2: Enable the domain configuration file

Let us enable the configuration file we created with the a2ensite tool:

$ sudo a2ensite busbookingtrial.com.conf

The output will suggest activating the new configuration but we can do it all collectively after running the following command that disables the original configuration file:

$ sudo a2dissite 000-default.conf



Now restart the Apache service:

$ sudo systemctl restart apache2

### Step 3: Test for errors

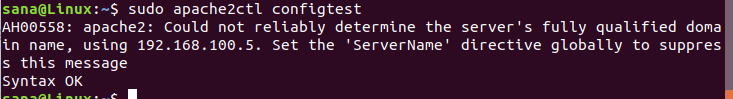
Finally, let us test if there are any configuration errors through the following command:

$ sudo apache2ctl configtest

If you do not get any errors, you will get the following output:

Test configuration

However, the following error is common in Ubuntu 18.04



**Resolve the error:**

Enter the following command in order to resolve the above-mentioned error:

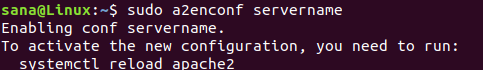
$ echo "ServerName busbookingtrial.com

| sudo tee /etc/apache2/conf-available/ busbookingtrial.conf

Resolve servername error

And then:

$ sudo a2enconf busbookingtrial.com



Now when you check again for errors, you will see this error resolved through the following output:

https://vitux.com/wp-content/uploads/2018/11/word-image-18.png

### Step 4: Test if Apache is serving your domain name

Apache server is now configured to serve your domain name. This can be verified by entering your server name as follows in any of the web browsers running on your system:

***http://sampledomain.com***

## Some Common Apache Management Commands

After setting up the web server, you might have to perform some basic management operations on Apache. Here are the commands that you can enter in your Terminal application for these operations.

sudo systemctl start apache2

Use this command as sudo in order to start the Apache server.

sudo systemctl stop apache2

Use this command as sudo in order to stop the Apache server when it is in start mode.

sudo systemctl restart apache2

Use this command as sudo in order to stop and then start the Apache service again.

sudo systemctl reload apache2

Use this command as sudo in order to apply the configuration changes without restarting the connection.

sudo systemctl enable apache2

Use this command as sudo in order to enable Apache to be started every time you boot your system.

sudo systemctl disable apache2

Use this command as sudo in order to disable if you have set up Apache to be started every time you boot your system.

(Ubuntu 18)

1. On the remote mysql server:
2. vim /etc/mysql/mysql.conf.d/mysqld.cnf
3. comment out like this:
4. #bind-address = 127.0.0.1
5. Restart your mysql server
6. service mysql restart