Magento 2 Installation with Nginx Webserver and Certbot SSL Certificate

1. Install Ubuntu and set date using dpkg-reconfigure tzdata
2. Install Nginx
   1. apt-get update
   2. apt-get -y install nginx
   3. to test nginx, open browser and open the ip address of web server
   4. create a new file with name superstore in /etc/nginx/sites-available and enter below code in it

upstream fastcgi\_backend {

server unix:/run/php/php7.2-fpm.sock;

}

server {

listen 80;

server\_name superstore.com 192.168.33.10;

set $MAGE\_ROOT /var/www/html/superstore;

set $MAGE\_MODE developer;

error\_log /var/log/nginx/superstore\_error.log;

access\_log /var/log/nginx/superstore\_access.log;

include /var/www/html/superstore/nginx.conf.sample;

}

server {

listen 443 ssl;

server\_name superstore.com 192.168.33.10;

ssl on;

ssl\_certificate /etc/letsencrypt/live/magento.yourdomain/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/magento.yourdomain/privkey.pem;

set $MAGE\_ROOT /var/www/magento2;

set $MAGE\_MODE developer;

include /var/www/magento2/nginx.conf.sample;

}

* 1. create a symbolic link using below command

ln -s /etc/nginx/sites-available/superstore /etc/nginx/sites-enabled/superstore

* 1. restart the nginx and php service

systemctl restart nginx && systemctl restart php7.2-fpm

* 1. test the nginx server by running command

nginx -t

nginx -T

1. Install PHP7.2-FPM , PHP-CLI and Related Modules
   1. apt-get -y install php7.2-fpm php7.2-cli or php-fpm in 20.04 ubuntu
   2. install the below php extensions
      1. sudo apt-get install -y php7.2-common php7.2-gd php7.2-mysql php7.2-curl php7.2-intl php7.2-xsl php7.2-mbstring php7.2-zip php7.2-bcmath php7.2-iconv php7.2-soap zip unzip php7.2-zip
   3. to setup php7.2 as default php version, run below command

update-alternatives --set php /usr/bin/php7.2

* 1. open /etc/php/7.2/fpm/php.ini and /etc/php/7.2/cli/php.ini and check below
     1. memory\_limit = 2G
     2. max\_execution\_time = 1800
     3. zlib.output\_compression = On
     4. opcache.enable=1
     5. opcache.save\_comments=yes
     6. date.timezone = Asia/Dubai
     7. upload\_max\_filesize = 100M
  2. restart php-fpm services
     1. systemctl restart php7.2-fpm

1. Install SQL Server
   1. apt-get install -y mysql-server mysql-client
   2. mysql\_secure\_installation (choose all Yes Option)
   3. if we want to import large data in mysq, edit sql.cnf file
   4. create magento database using below commands
      1. login to mysql
      2. create database database\_name;
      3. create user ‘username’@’localhost’ identified by ‘password’;
      4. grant all on database\_name.\* TO ‘username’@’localhost’ with grant option;
      5. flush privileges;
2. Installing Elasticsearch
   1. apt-get -y update && apt-get install -y openjdk-8-jdk

use above command to install java

* 1. java -version

use above command to check java version

* 1. sudo apt -y install gnupg
  2. wget -qO - <https://artifacts.elastic.co/GPG-KEY-elasticsearch> | sudo apt-key add -
  3. sudo apt -y install apt-transport-https
  4. echo "deb <https://artifacts.elastic.co/packages/oss-7.x/apt> stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list
  5. sudo apt update
  6. sudo apt -y install elasticsearch-oss
  7. sudo systemctl daemon-reload
  8. sudo systemctl enable elasticsearch.service
  9. sudo systemctl start elasticsearch.service
  10. sudo systemctl status elasticsearch.service

use above command from c to l to install elasticsearch

j. curl -XGET 'localhost:9200/\_cat/health?v&pretty'

use above command to check & verify elasticsearch installation

1. Install Composer
   * 1. apt install curl git
     2. curl -sS https://getcomposer.org/installer | sudo php -- --install-dir=/usr/bin --filename=composer –version=2.0
2. composer create-project --repository-url=https://repo.magento.com/ magento/project-community-edition .
   * 1. download magento setup using above command
     2. Obtain authentication keys

Username = 6d304bfc5ac8e8cdac4c60c14247ca39

Password = 65efe00a6ff3fcaf29d8ac13aef70531

Run the below command from magento directive to install magento.

sudo bin/magento setup:install --base-url-secure=https://192.168.10.10/ --db-host=localhost --db-name=magento --db-user=magento --db-password=m@g1t786 --admin-firstname=Rizwan --admin-lastname=Khan --admin-email=rizwan.wakil@gmail.com --admin-user=rizwan --admin-password=m@g1t786 --language=en\_US --currency=USD --timezone=Asia/Dubai --use-rewrites=1

To disable two factor authentication, issue below command

bin/magento mo:di Magento\_TwoFactorAuth

0

1. Setup Permission for Magento files

First try this

* 1. chown -R www-data:www-data /var/www/html
  2. find . -type d -exec chmod 770 {} \;
  3. find .-type f -exec chmod 660 {} \;
  4. chmod u+x bin/magento

1. after installation of magneto, change the permission to of etc directory
   1. chmod -R 440 /var/www/html/app/etc

Then Try this

* 1. Create a user with below command

adduser magento\_user

* 1. Set user password with below command

passwd magento\_user

* 1. Add user to web server group using below command

usermod -aG www-data magento\_user

* 1. Move to magento root directory and using root user run below commands

find var generated vendor pub/static pub/media app/etc -type f

-exec chmod g+w {} +

find var generated vendor pub/static pub/media app/etc -type d

-exec chmod g+ws {} +

chown -R magento\_user:www-data . # Ubuntu

chmod u+x bin/magento

* 1. Now switch to magento\_user and do all work

1. **Installing Sample Data in Magento**
   1. php bin/magento sampledata:deploy
   2. php bin/magento setup:upgrade
2. **Creating Magento Crontab**
   1. Change to magento root directory
   2. Enter below command

php bin/magento crontab:install --force

* 1. To view crontab, enter following command

crontab -l

* 1. To remove crontab, run below command

php bin/magento cron:remove

* 1. To run the crontab from command line

php bin/magento cron:run

php bin/magento cron:run --group index

php bin/magento cron:run --group default

* 1. cron logging

by default cron logging are stored in /var/log directory

failed job with erros or missed are logged to

/var/log/support\_report.log

Critical error are logged in

/var/log/exception.log

* 1. all cron data are written to cron\_schedule table in database

1. Installing Extension
   1. Backup your database
   2. Enable maintenance mode using below command

php bin/magento maintenance:enable

* 1. Get extension from magento marketplace
  2. Get extension composer name & version by click profile 🡪 My Purchase 🡪 Technical Detail
  3. Update composer.json file in magento project with name &version of extension by running below command from magento root directory

composer require extension\_name:version

* 1. Verify extension install properly by running below command

php bin/magento module:status extension\_name

* 1. Enable and configure the extension

To enable the module run below command

php bin/magento module:enable module\_name

To register the module run below command

php bin/magento setup:upgrade

Then run the magent recompile command

php bin/magento setup:di:compile

1. Upgrade an Extension
   1. Copy the composer file to magento root
   2. Run below command to update composer

Composer update vendor/module\_name

Composer require vendor/module-name ^x.x.x

* 1. Run below commands

php bin/magento setup:upgrade –keep-generated

php bin/magento setup:static-content:deploy -f

php bin/magento cache:clean

1. Installing phpMyAdmin
   1. Run below command to install phpMyAdmin

apt-get install phpMyAdmin

* 1. To access the phpMyAdmin, enter below url in browser

192.168.33.10/phpMyAdmin

1. SSL Installation & Configuration

apt update

install certbot using below command

add-apt-repository ppa:certbot/certbot

install certbot nginx package

apt install python-certbot-nginx

obtain SSL certificate using certbot

certbot –nginx -d superstore.com

above command will work for valid domain name only

your certificate will be generated and stored in

/etc/letsencrypt/live/yourdomainname/fullchain.pem

Add the following line below the nginx configuration file

listen 443 ssl;

ssl\_certificate /etc/letsencrypt/live/yourdomain.com/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/yourdomain.com/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

now enable SSL in magento 2 from admin

store🡪 configuration 🡪 General 🡪 web 🡪 Base URLs (Secure)

set URL and enable it.

If we are not able to access our admin panel, we can set using command line

php bin/magento setup:store-config:set –use-secure-admin=1

1. Install Varnish and Configuration
   1. Run below command to install and start and enable varnish

apt-get install varnish

systemctl start varnish

systemctl enable varnish

* 1. By default varnish listen on port 6081 for public and 6082 for backend
  2. Checking varnish service is running

netstat -plntu

* 1. Add the following line below the nginx configuration file

1. **Backup Magento System**
   1. Use the below command to backup magento code, media & database

php bin/magento setup:backup –code

this command will put website in maintenance mode, then take backup

this backup will not include /var and pub/static folder

the backup file will be store in /var directory

* 1. php bin/magneto setup:backup –media

this command will backup media directory and keep the backup file in /var directory

* 1. below command will show list of all backup taken

php bin/magento info:backup:list

1. **Backup Restoration / Rolling back a backup**
   1. Use below command to rolling back a backup

php bin/magento setup:rollback -c backup\_name\_code -m backup\_name\_media -d back\_name\_db

1. **Enabling Performance in Magento 2**
   1. **Configuring Redis for Backend Cache**

Redis is key value storage store in memory 🡪 in magento 1 we used Memcache(d) and Alternative PHP Cache (APC) 🡪

**Redis Installation**

**wget** [**http://download.redis.io/releases/redis-3.0.5.tar.gz**](http://download.redis.io/releases/redis-3.0.5.tar.gz)

**tar xzf redis-3.0.5.tar.gz**

**cd redis-3.0.5**

**make && make install**

run the script from redis-3.0.5/utils directory

**./install-server.sh**

**Checking Rediss**

redis-cli --version

service redis\_6379 status

netstat -anp | grep redis

**Installing PHP module to communicate with Redis**

**git clone https://github.com/phpredis/phpredis.git**

**cd phpredis**

**apt install php7.0-dev**

**phpize**

**./configure**

**make && make install**

**echo "extension=redis.so" | sudo tee /etc/php/7.4/mods-available/ redis.ini**

**ln -s /etc/php/7.4/mods-available/redis.ini /etc/php/7.4/fpm/conf.d/20- redis.ini**

**ln -s /etc/php/7.4/mods-available/redis.ini /etc/php5/cli/conf.d/20- redis.ini**

**service php5-fpm restart**

**to make sure the redis PHP and Redis server are running together,**

**php -r "if (new Redis() == true){ echo \"\r\n OK \r\n\"; }"**

* 1. **Configuring Memcached for session caching**
  2. **Configuring Varnish as a Full page cache**

Setup varnish is simple but configuration is hard.

Magento provide default varnish file, which holds all elements to be cache or not

Varnish includes Varnish Configuration Language (VCL)

Varnish not support HTTPS, We will need SSL Proxy such as NGINX

1. Installing Varnish Server

Run below command to install varnish server

**apt install -y apt-transport-https**

below command will create varnish repository

**echo "deb https://repo.varnish-cache.org/ubuntu/ trusty varnish-4.1" | sudo tee -a /etc/apt/sources.list.d/varnish-cache. list**

Below command will add varnish key to our system

**curl https://repo.varnish-cache.org/GPG-key.txt | apt-key add -**

below command will update system and install varnish

**apt-get update && apt-get install -y varnish**

**service varnish start**

**varnishd -V**

**service varnish status**

**netstat -anp | grep varnish**

By default varnish server is using port no. 6081 and 6082

Run the below command to inform, varnish server that web server is running on which port

**sed -i 's/6081/web-server-port-no/' /etc/default/varnish**

**sed -i 's/6081/ web-server-port/' /lib/systemd/system/varnish.service systemctl daemon-reload**

**service varnish restart && service nginx restart**

run below command to check both varnish & NGINX are running on same port

**netstat -upnlt | egrep 'varnish|nginx'**

now login to Magento backend 🡪 Store 🡪configuration 🡪 Advance 🡪 System 🡪 Full Page cache 🡪 select varnish Caching

Change the /etc/varnish/default.vcl file with following details

Backend defult {

.host = “127.0.0.1”;

.port = “8080”;

}

Now start the varnish service and clean the cache

**service varnish restart**

**php bin/magento cache:clean**

**sphp bin/magento cache:flush**

To test run the below command and check the HIT Value

**Curl -I http://domainname.com**

To Monitor Varnish, use below tool

Varnishstat

Varnishlog

If there is no content in /var/page\_cache directory that mean varish is working

* 1. **Configuring Magento 2 with cloudFlare**
  2. **Configure Optimized Images Magento 2**
  3. **Configuring Magento 2 with HTTP / 2 including SSL**
  4. **Configuring Magento 2 performance testing**