**Epel - aws linux 2**

------------------

sudo wget -r --no-parent -A 'epel-release-\*.rpm'

https://dl.fedoraproject.org/pub/epel/7/x86\_64/Packages/e/

sudo rpm -Uvh dl.fedoraproject.org/pub/epel/7/x86\_64/Packages/e/epel-release-\*.rpm

sudo yum-config-manager --enable epel\*

sudo yum repolist all

# **Configure SSL Nginx Lets Encrypt Centos 7**

## *Let’s Encrypt is a free, automated, and open certificate authority brought to you by the nonprofit Internet Security Research Group (ISRG).*

## Lets Encrypt is a free and open certificate provided by ISRG (Internet Security Reaserch Group ) . We are going to integrate Lets Encrypt with Centos 7 and Using Nginx as a reverse proxy .

We assume you already have installed Nginx and CentOs 7 . Lets Encrypt Certificates are also used by lot of hosting providers now a days .

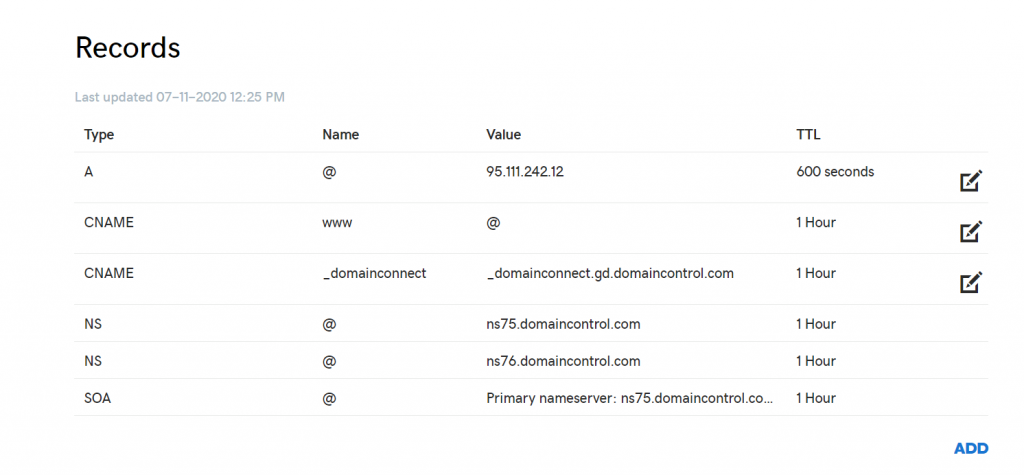
Make sure you have a domain and a Static IP , in order to proceed further .

## **Install Certbot**

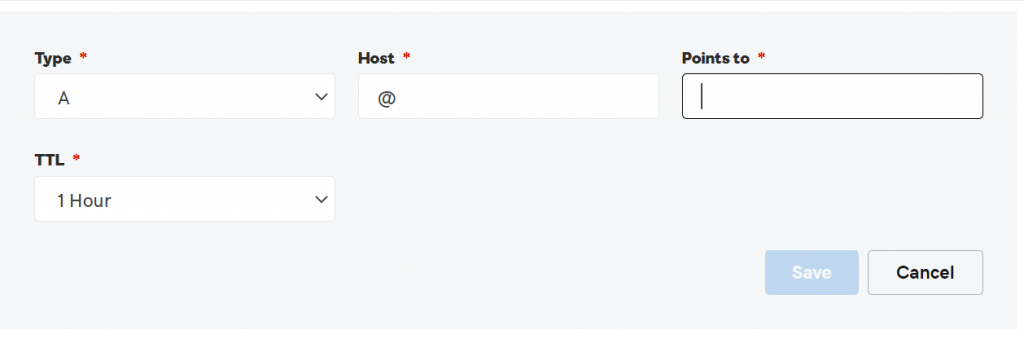
sudo yum install certbot

## **Configure GoDaddy Domain to Run Nginx**

Now Make sure you have purchased a GoDaddy domain and you have access to its Doamin Management . Once you Move to Domain Management , you would see something like below .



Click on ADD button , Select A record and enter details as below



**Now You need to enter your VPS IP in place of Points To\*.**

## **Generate –** [**Hellman key exchange**](https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange) **(DH)**

As Wrote in Wiki “**DH is a way to securely exchange cryptographic keys over public internet .DH is one of the earliest practical examples of public** [**key exchange**](https://en.wikipedia.org/wiki/Key_exchange) **implemented within the field of** [**cryptography**](https://en.wikipedia.org/wiki/Cryptography).”

sudo openssl dhparam -out /etc/ssl/certs/dhparam.pem 2048

## **Setup acme-challenge Lets Encrypt**

It is one of the way to validate your domain ownership , lets encrypt exchanges token and validates the authority before generating a SSL certificate .

Now we need to define a path where your Nginx and Lets Encrypt can Collaborate and store token to validate . Once lets encrypt servers token Nginx has to know the URL .

Have a look at the configuration below

sudo mkdir -p /var/lib/letsencrypt/.well-known

sudo chgrp nginx /var/lib/letsencrypt

sudo chmod g+s /var/lib/letsencrypt

sudo mkdir /etc/nginx/snippets

**Create following file /etc/nginx/snippets/letsencrypt.conf**

location ^~ /.well-known/acme-challenge/ {

allow all;

root /var/lib/letsencrypt/;

default\_type "text/plain";

try\_files $uri =404;

}

Create File /etc/nginx/snippets/ssl.conf

ssl\_dhparam /etc/ssl/certs/dhparam.pem;

ssl\_session\_timeout 1d;

ssl\_session\_cache shared:SSL:50m;

ssl\_session\_tickets off;

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

ssl\_ciphers 'ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS';

ssl\_prefer\_server\_ciphers on;

ssl\_stapling on;

ssl\_stapling\_verify on;

resolver 8.8.8.8 8.8.4.4 valid=300s;

resolver\_timeout 30s;

add\_header Strict-Transport-Security "max-age=15768000; includeSubdomains; preload";

add\_header X-Frame-Options SAMEORIGIN;

add\_header X-Content-Type-Options nosniff;

## **Configure Domain SSL With Nginx**

We are going to configure SSL for domain **www.abcd.com** . Nginx needs to know the location where lets encrypt going to put files and serve.

Create a file under /etc/nginx/conf.d/yourdoamin.com.conf

**server {**

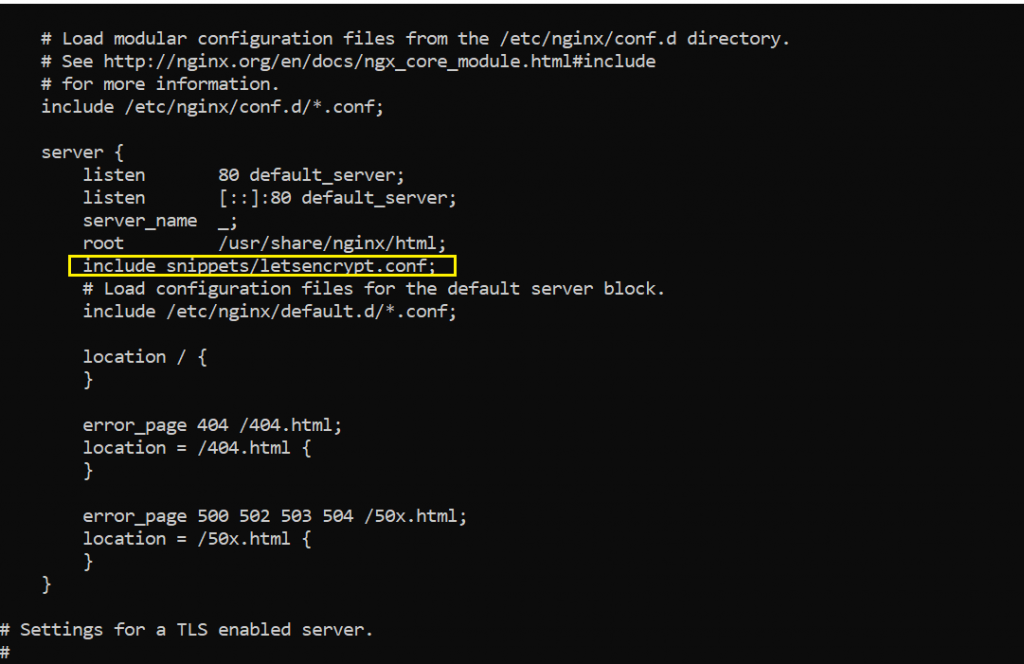
**listen 80;**

**server\_name example.com www.mydnslookup.com;**

**include snippets/letsencrypt.conf;**

**}**

Now before moving to next step to generate certificate using Certbot plugin . You can do this optional step.

Navigate to /etc/nginx and open your default nginx.conf configuration and update the following line marked below .

## **Generate SSL Certificate**

Now we will be moving ahead to generate SSL certificate from lets encrypt . Make sure you reload Nginx before moving ahead .

$ sudo systemctl reload nginx

$ sudo certbot certonly --agree-tos --email mydnslookup@gmail.com --webroot -w /var/lib/letsencrypt/ -d mydnslookup.com -d www.mydnslookup.com

## **Configuring Nginx SSL With Lets Encrypt**

Now that we have generated the certificate in above step we are now going to Creating configure and enable Nginx to use that certificate and route to https .

Now what we are going to do is any request that comes from , **www.mydomain.com** or http://mydomain.com has to be routed to **https://mydomain.com** . Hence we have used a 301 redirect in nginx configuration .

**Have a look at the configuration below**

server {

listen 80;

server\_name www.mydnslookup.com mydnslookup.com;

include snippets/letsencrypt.conf;

return 301 https://$host$request\_uri;

}

server {

listen 443 ssl http2;

server\_name www.example.com;

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

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

ssl\_trusted\_certificate /etc/letsencrypt/live/mydnslookup.com/chain.pem;

include snippets/ssl.conf;

include snippets/letsencrypt.conf;

return 301 https://mydnslookup.com$request\_uri;

}

server {

listen 443 ssl http2;

server\_name mydnslookup.com;

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

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

ssl\_trusted\_certificate /etc/letsencrypt/live/mydnslookup.com/chain.pem;

include snippets/ssl.conf;

include snippets/letsencrypt.conf;

# . . . other code

}

sudo systemctl reload nginx

Bingo !!! Now You should be able to run and use SSL with Nginx .

Point to note here is Nginx Can act as a reverse proxy or as a load balancer . You Can do anything you want to using any domain and VPS. Weather it is **Centos or Ubuntu or AWS** the process of configuring SSL exactly same .

## **Troubleshooting Guide**

1. In Case You are facing issuess like “**Job for nginx.service failed because the control process exited with error code. See “systemctl status nginx.service” and “journalctl -xe” for details.”** Make Sure You have the Port 80 free .

lsof -i :80

If its blocked kill the port listed from the above command using

kill -2 <ProcessID>

2. Lets Encrypt CertBox Certificate generation failed or ACME challenge Failed

Make sure you check the logs of Nginx **/var/log/nginx** and check **error,log** .

Navigate to your nginx logs folder at **/var/log/nginx** and check are you are getting 404 while generating certificate . If yes then follow option in section **Configure Domain SSL With Nginx**