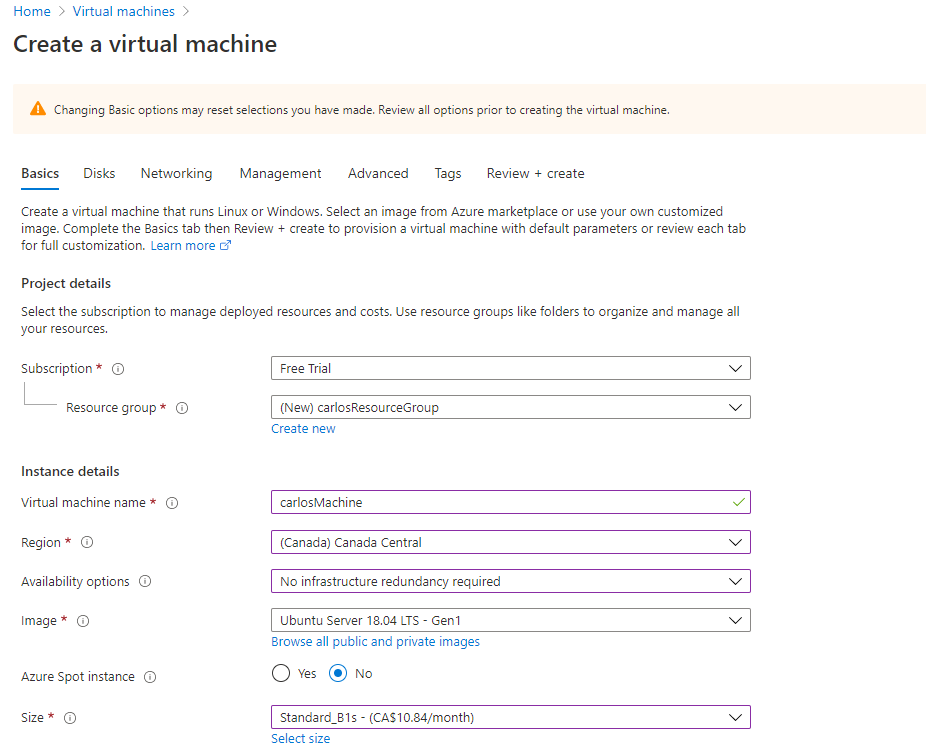
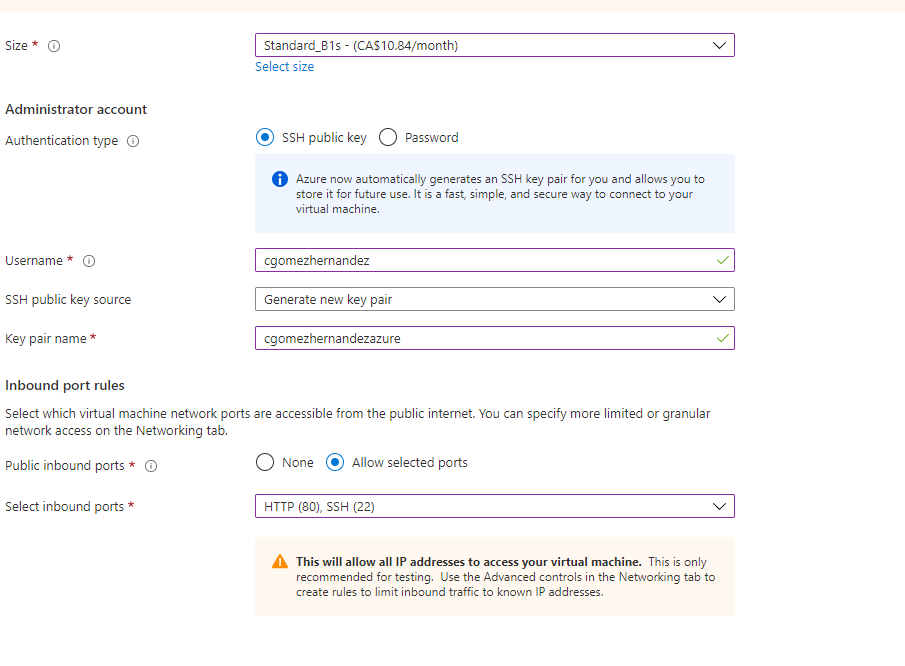
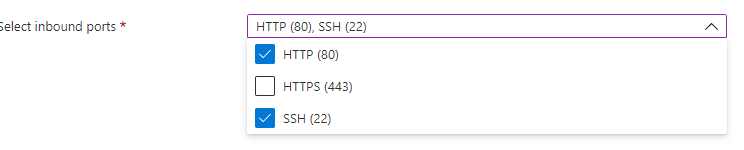
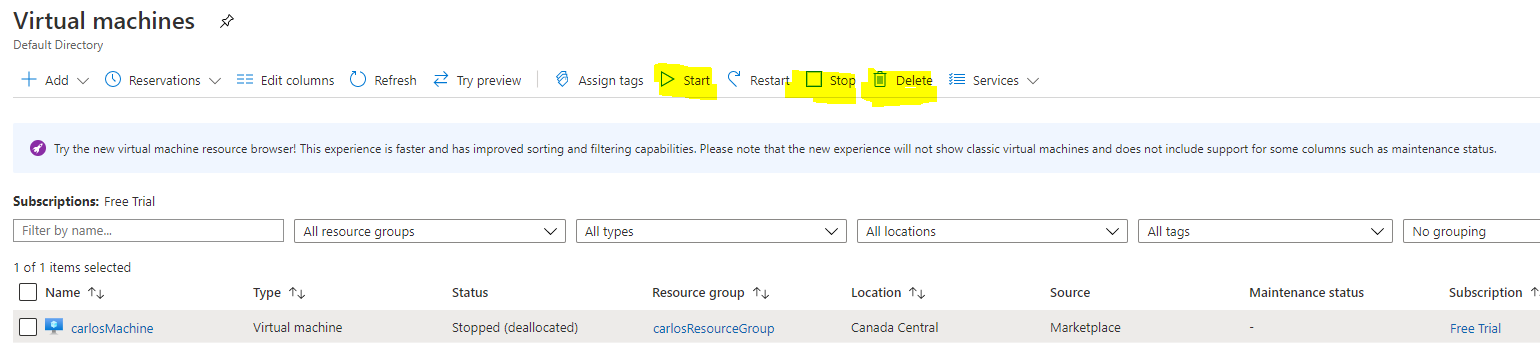
**CREATE A VIRTUAL MACHINE CONFIGURATIONS**

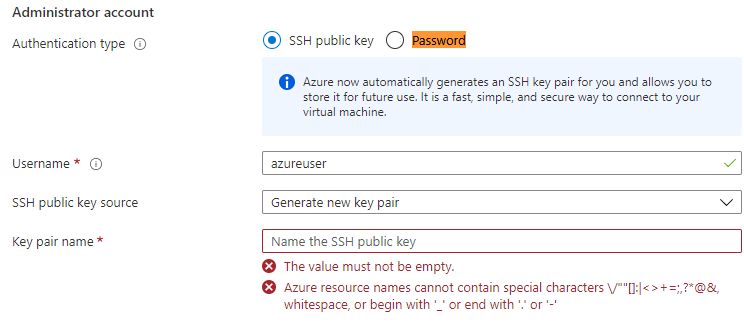






The rest of the tabs we left as Default

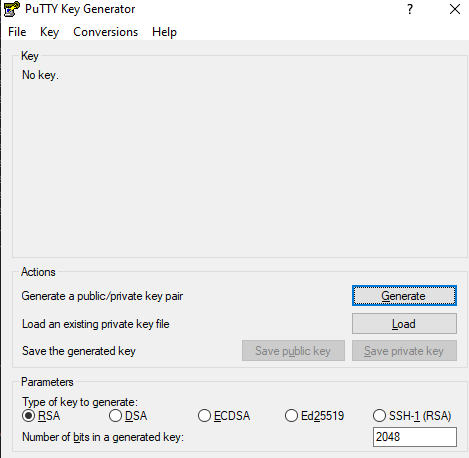
  
  
You are paying for:

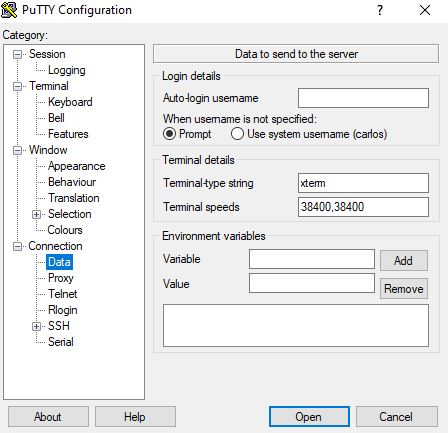
- The virtual Machine Running  
- The SSD containing your data  
  
If you want Azure to stop charging you for SSD you need to DELETE the machine.  
If you want Azure to stop charging you for runtime you need to stop the machine.  
  
Now that we have configured the machine we need to :

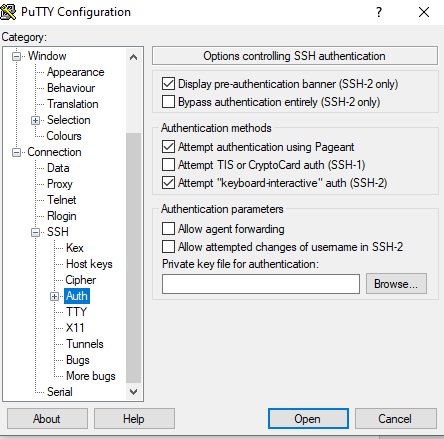
We created an account using an SSH public key and Azure gave us a key file.  

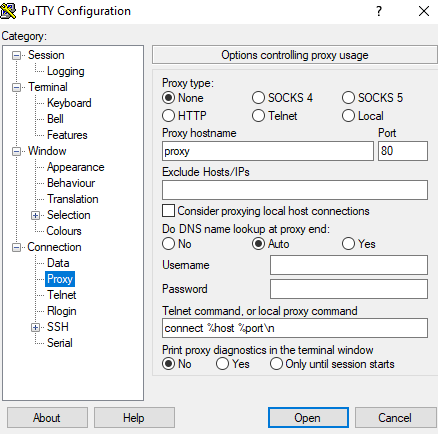

However Windows is really bad for connecting to the server directly:  

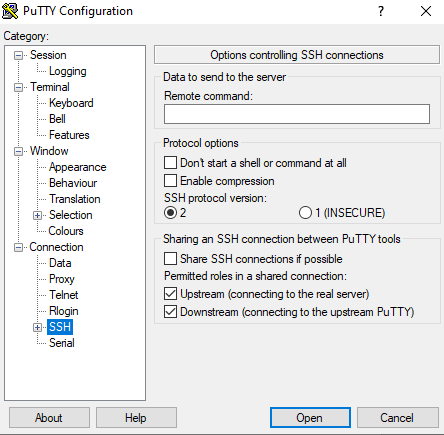

So you will need to download **Putty** and **puttygen here:**  
**https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html**



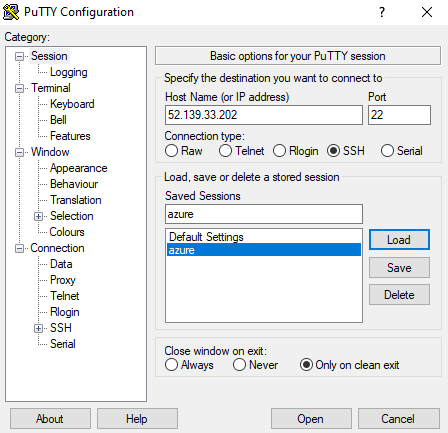
Load the key that you downloaded from Azure here, Generate and Save Private Key.   
  
**PUTTY CONFIGURATIONS:**







**LOGG INTO YOUR VIRTUAL MACHINE (REMEMBER TO HAVE THE MACHINE ON IN AZURE)**



**SETTUP YOUR VIRTUAL MACHINE and NGINX WEB SERVER**

sudo apt update -- these are the basic linux configurations

sudo apt upgrade -- this are the basic linux configurations python etc

sudo apt install nginx – this is the web server

**INSTALL GITHUB REPO**

git clone <https://github.com/Carlos-Gomez-Hernandez/git-public.git>

user: Carlos-Gomez-Hernandez

**CREATE YOUR VIRTUAL ENVIRORMENT:**

**Install pip**

sudo python3 -m pip install --user --upgrade pip

**Check python and pip are running**

python3 -m pip --version

python3 -m pip install --user --upgrade pip

python3 -m venv myNewWebApp

source myNewWebApp/bin/activate

**CREATE YOUR VIRTUAL ENVIRORMENT:**

python -m pip install Django

pip install django-extensions

pip install djangorestframework

**FOR CORS POLICY (OPEN API)**

python -m pip install django-cors-headers

**SETUP NGINX AS REVERSE PROXI**

[**https://linuxize.com/post/nginx-reverse-proxy/**](https://linuxize.com/post/nginx-reverse-proxy/)

cd /etc/nginx/sites-enabled

sudo vim default

The following screenshot displays the first attempt, to configure the reverse proxi. It workes however django rest framework won’t because ngnx do not like when applications call urls from different networks.

In order to fix this we needed to add the if statements in the second image to allow Access control **CORS policy settings**:

This code will tell NGNX to trust the Django-Rest Framework USLs. Later we will need to tell Django to trust **CORS**  
<https://www.ryadel.com/en/nginx-access-control-allow-origin-cors-policy-settings/>

server {

if ($request\_method = 'OPTIONS') {

add\_header 'Access-Control-Allow-Origin' '\*';

add\_header 'Access-Control-Allow-Credentials' 'true';

add\_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';

add\_header 'Access-Control-Allow-Headers' 'DNT,X-CustomHeader,Keep-Alive,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type';

add\_header 'Access-Control-Max-Age' 1728000;

add\_header 'Content-Type' 'text/plain charset=UTF-8';

add\_header 'Content-Length' 0;

return 204;

}

if ($request\_method = 'POST') {

add\_header 'Access-Control-Allow-Origin' '\*';

add\_header 'Access-Control-Allow-Credentials' 'true';

add\_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';

add\_header 'Access-Control-Allow-Headers' 'DNT,X-CustomHeader,Keep-Alive,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type';

}

if ($request\_method = 'GET') {

add\_header 'Access-Control-Allow-Origin' '\*';

add\_header 'Access-Control-Allow-Credentials' 'true';

add\_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';

add\_header 'Access-Control-Allow-Headers' 'DNT,X-CustomHeader,Keep-Alive,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type';

}}





**SETUP NGINX AS REVERSE PROXI**  
At this point we should have NGINX configurations ready now we should be able to manage it.

**From Any Directory:**

**Every time you make a change in your nginx settings you need to restart**

sudo systemctl restart

**Active means all is good:**

sudo systemctl status

**Every time you make a change in your application**

sudo systemctl start carlosApp

sudo systemctl status carlosApp

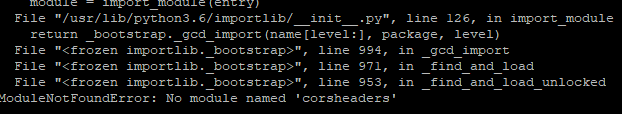
At this point you should be able to see your application in the internet using azure path.



**APLICATION DEBUGGING:  
  
To Test For Application Problems you need to first stop the server:**

sudo systemctl stop carlosApp

/home/cgomezhernandez/myNewWebApp/bin/python /home/cgomezhernandez/espacios/git-public/core/manage.py runserver



<https://medium.com/@dtkatz/3-ways-to-fix-the-cors-error-and-how-access-control-allow-origin-works-d97d55946d9>

<https://stackoverflow.com/questions/35760943/how-can-i-enable-cors-on-django-rest-framework>

<https://www.ryadel.com/en/nginx-access-control-allow-origin-cors-policy-settings/>

<https://github.com/adamchainz/django-cors-headers#configuration>

<https://stackoverflow.com/questions/58558804/django-cors-headers-and-nginx-config-preflight-response-missing-cors-headers>

**Automate NGINX to run the application even when the linux session cgomezhernandez is offline:  
  
Node Exporter Service File**https://www.digitalocean.com/community/tutorials/how-to-install-prometheus-on-ubuntu-16-04

**CHECK WHERE PYTHON IS LOCATED**

which python

*/usr/bin/python*

*(remember we have a virtual environment)*  
  
**GO TO**

cd etc/systemmd/system/

**CREATE AN APPLICATION CONFIGURATION FILE**

sudo vim carlosAPP.service

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++  
ADD THE FOLLOWING CONFIGURATION SETTINGS AND ADJUST AS NEEDED  
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

[Unit]

Description=carlosApp

Wants=network-online.target

After=network-online.target

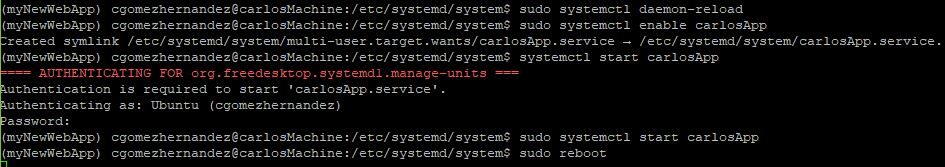
[Service]

Type=simple

ExecStart=**/home/cgomezhernandez/myNewWebApp/bin/python** **/home/cgomezhernandez/espacios/git-public/core/manage.py runserver**

[Install]

WantedBy=multi-user.target

TELL THE SYSTEM TTO READ THE UNIT FILE

**API CORS CANNOT FETCH PROBLEM:**



**INSTALL CORS HEADERS:**

**INSTALL THE DJANGO EXTENSION DJANGO-CORS-HEADERS**

python -m pip install django-cors-headers

*(remember we have a virtual environment)*  
  
**GO TO /git-public/core/core/settings.py**

INSTALLED\_APPS = [

...

'corsheaders',

...

]

MIDDLEWARE = [

'corsheaders.middleware.CorsMiddleware',

'django.middleware.common.CommonMiddleware',

...

]

*(THESE TWO SETTINGS MUST GO ON TOP OF EVERY OTHER SETTING)*

**OPEN THE PORT TO YOUR FRONT END SERVER**

CORS\_ORIGIN\_WHITELIST = [

'http://52.139.33.202'

]

python -m pip install django-cors-headers

**BACKUP:**

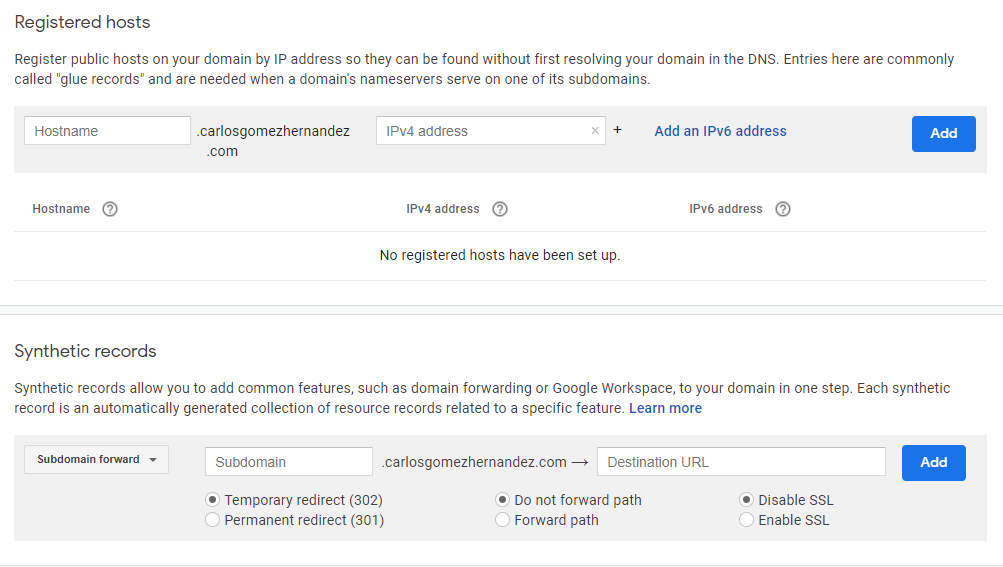
**Recursive Directory**

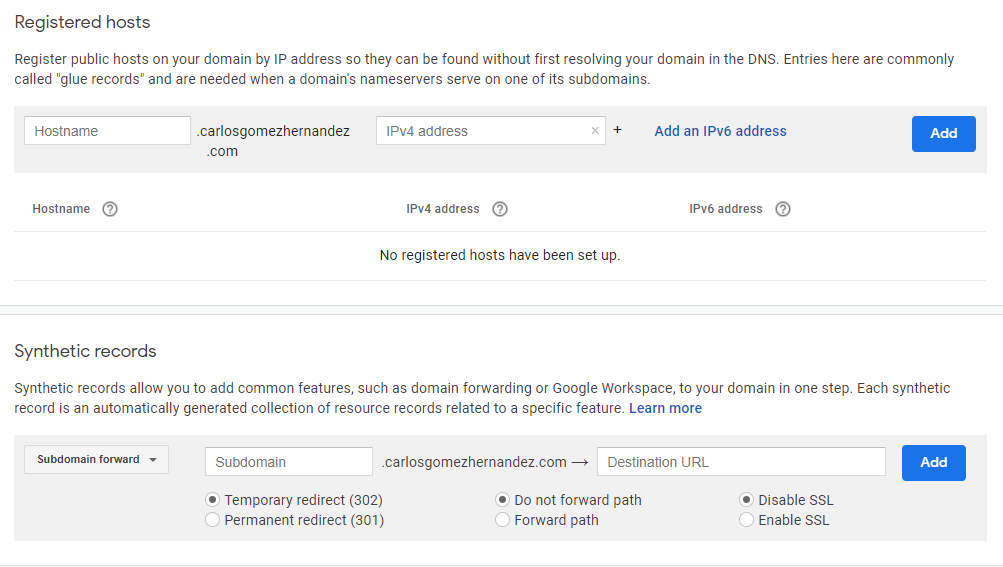
cp -r git-public/ git-public.back

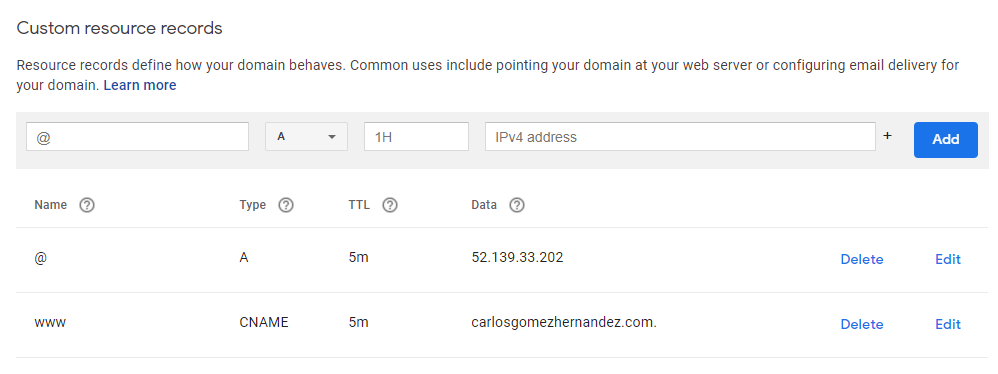
**Single file**

cp fileName fileName.back

**SETTUP THE DNS SERVERS DOMAIN:**

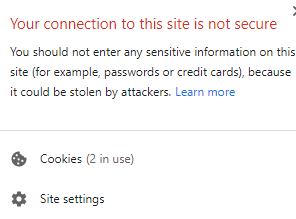
After buying the public Domain with Google Domains you can use their DNS servers (they come paid with the Domain subscription: ****

****

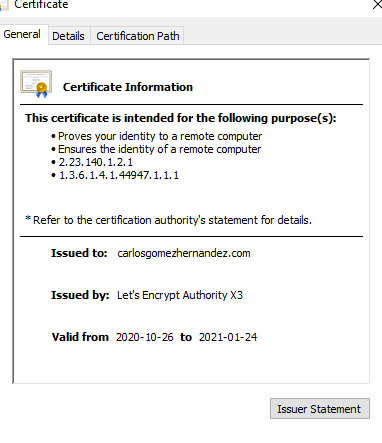
****

At this point you should be able to go to your website by typing www**.carlosgomezhernandez.com**

**FIXING THE SECURE CONNECTION TO YOUR WEB APPLICATION:**

****

This issue appears because your NGINX Ubuntu machine is does not have a certificate of Autority (CA) that provides encryption.



In order to obtain this certificate, you can manually perform a bunch of configurations in your NGINX or you can use a CERTBOT that does that for you.

[**https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-18-04**](https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-18-04)

**INSTALL CERTBOT**

**Add the repository for the certbot**

sudo add-apt-repository ppa:certbot/certbot

**Install Certbot Packages**

sudo apt install python-certbot-nginx

**CONFIGURE THE NGINX FILE**

cd /etc/nginx/sites-available

sudo vim default

**CHANGE THIS LINE TO THE FILE**

server\_name carlosgomezhernandez.com www.carlosgomezhernandez.com;

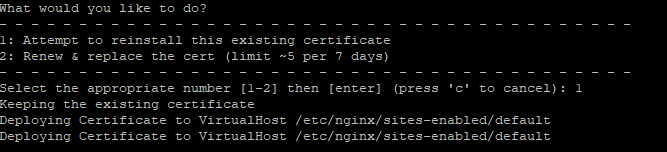
**RELOAD NGINX**

sudo systemctl reload nginx

**OBTAIN SSL CERTIFICATE**

sudo certbot --nginx -d carlosgomezhernandez.com -d www.carlosgomezhernandez.com

This is the part where certbot grants you a new SSL certificate.  
I choose 1

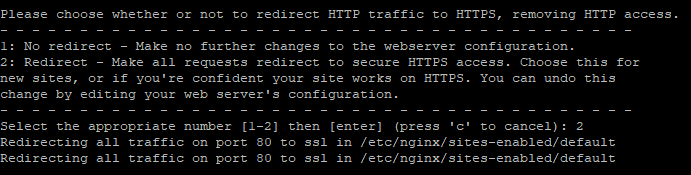
****

The following question has to do with HTTPS and HTTP responses of your website.

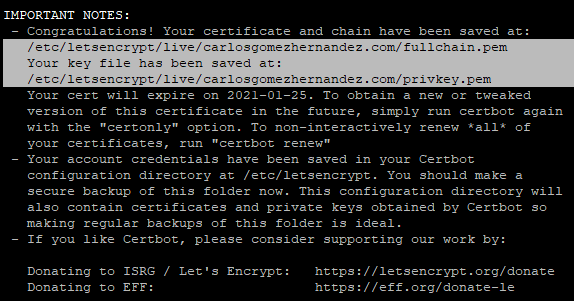
If you select 1 what will happen is the configuration file /etc/nginx/sites-available/default will not have the configurations needed for a secure connection to https:\\carlosgomezhernandez.com calls only for http:\\carlosgomezhernandez.com

Therefore if the user types in the browser [www.carlosgomezhernandez.com](http://www.carlosgomezhernandez.com) it will redirect them to a non-secure connection.

In order to avoid this, choose option 2:

****

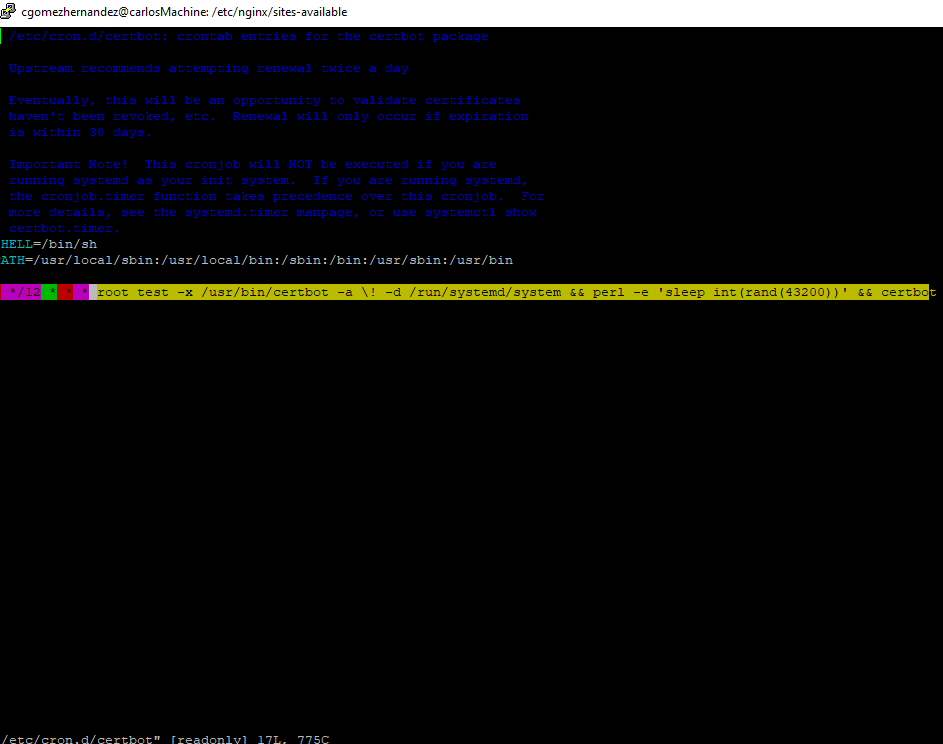
After this command ends running you will get two keys saved on your server. These keys can be reinstalled when if the machine ever goes down by running the command again.

****

**SETTUP AUTORENEWAL EVERY 12 DAYS**

The SSL certificate only works for 2 months and after that you need to request a new one.  
In order to avoid having to do this manually, you can type this command and it will generate a new SSL for you automatically.

sudo certbot renew --dry-run

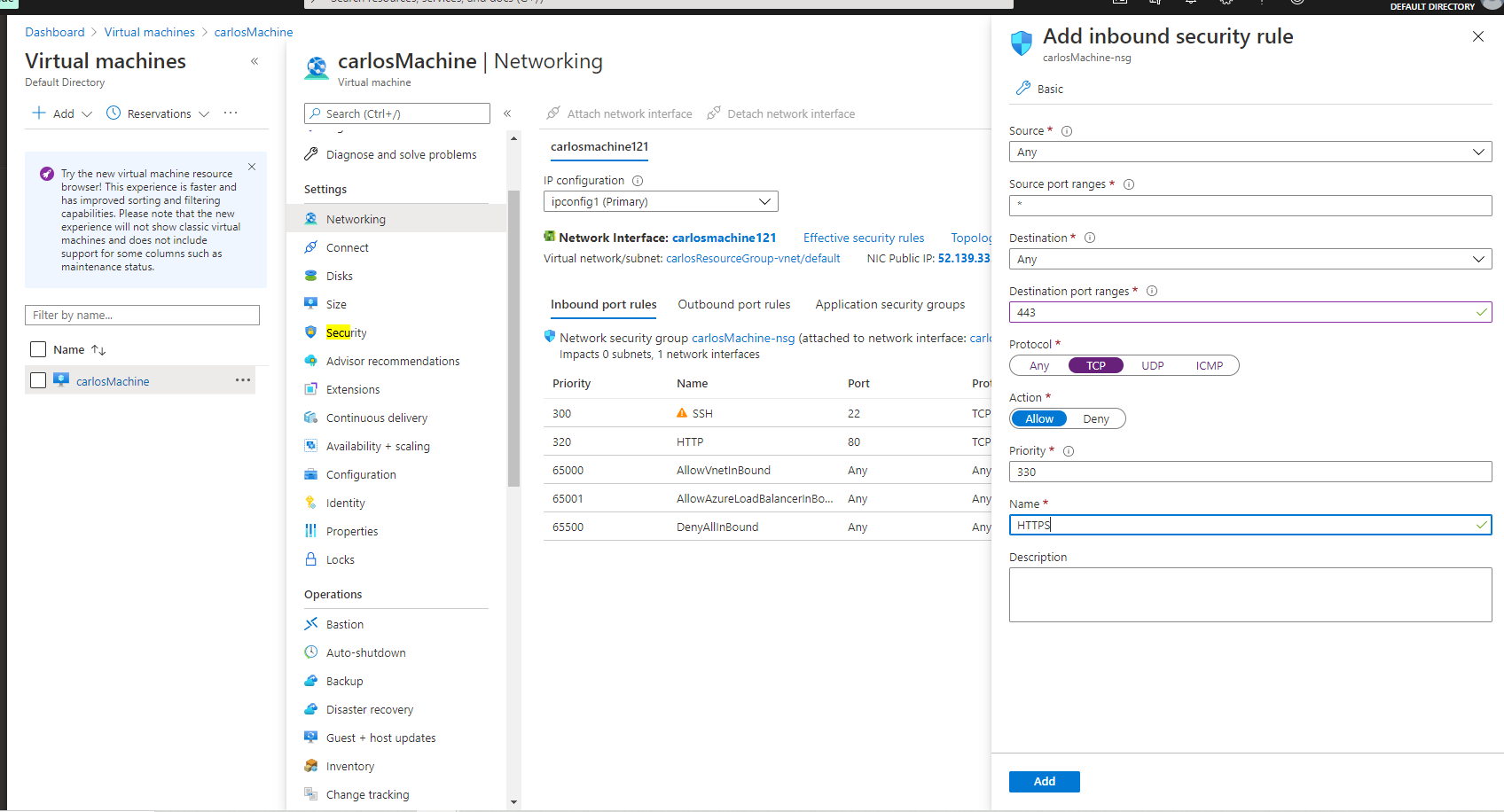
You can check the automation for your credential renewal here:   
vim /etc/cron.d/certbot  


**OPEN THE SECURE PORTS IN THE AZURE VIRTUAL MACHINE**

Ports are like Suite Numbers and IP addresses are like postal codes. The application is running on an specific IP address with the insecure port.

At this point you have the virtual machine running your application in port 80 which is the unsecure port, you need to create a new secure port for it 4043 so NGINX can use it.

<https://docs.microsoft.com/en-us/azure/virtual-machines/windows/nsg-quickstart-portal>

****

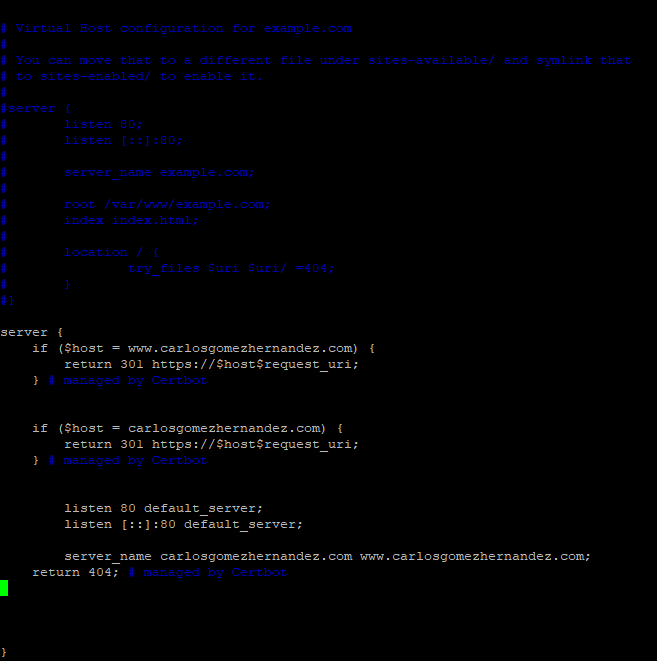
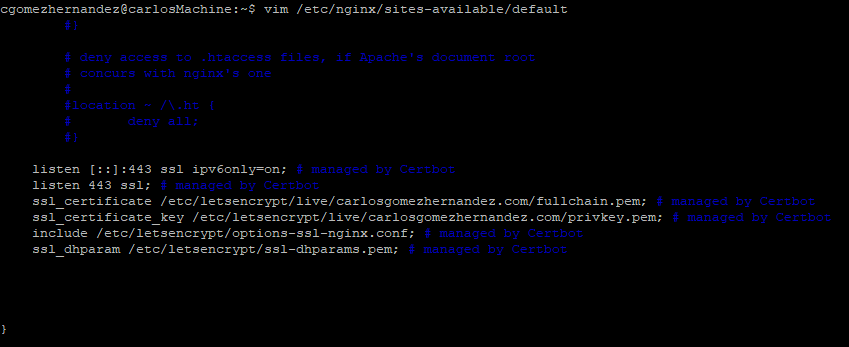
**RELOAD YOUR NGINX**

sudo systemctl reload nginx

**YOU ARE ALL SETTUP THIS IS THE LAST SCREENSHOT OF THE NGINX CONFIGURATION FILE FOR THE RECORDS:**

cd /etc/nginx/sites-available

sudo vim default

****