AWS Ubuntu 20 EC2 Nginx Install

In this tutorial, we will be installing the Nginx web server on an AWS Ubuntu 20 compute instance.

Prerequisites

- AWS Ubuntu 20 EC2 instance
- an AWS Free Tier account
- internet access

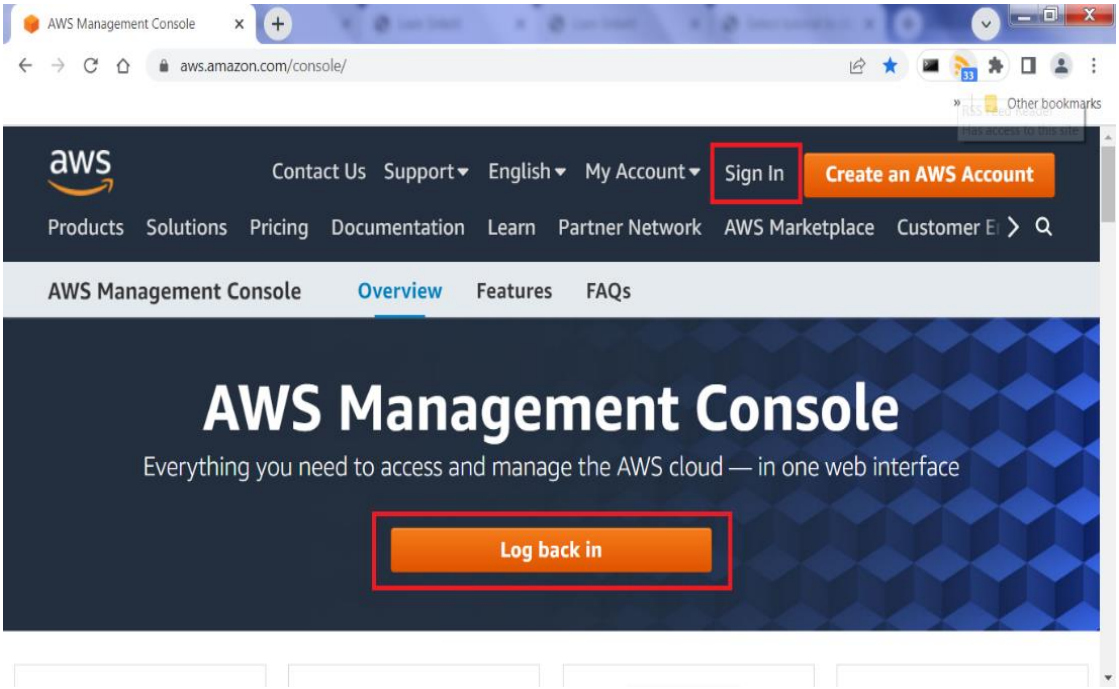
If you do not have an AWS account, you can access my **AWS Create Free Tier Account** tutorial **here**.

If you do not have an AWS Ubuntu 20 EC2 instance, my tutorial **Create AWS Ubuntu 20 EC2 Instance** is **here**.

Steps to complete tutorial:

- Start Ubuntu 20 EC2 Instance
- Update Virtual Private Cloud (VPC)
- Connect to Ubuntu 20
- Install Nginx Web Server
- Configure Nginx to Host a Website

To begin, go to the following website, https://aws.amazon.com/console/ and log in to the console.

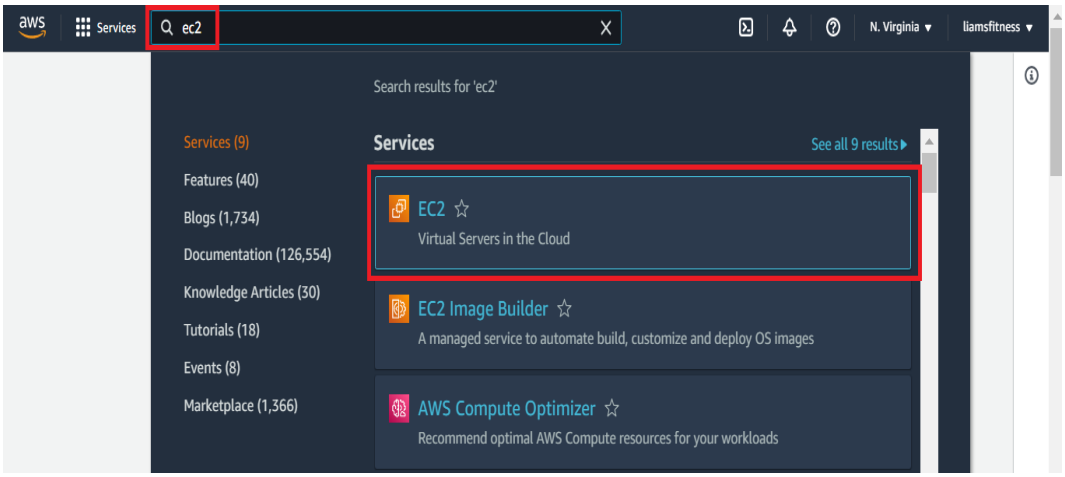


Start Ubuntu 20 EC2 Instance

At the end of my **Create AWS Ubuntu 20 EC2 Instance** tutorial, I suggested shutting down your EC2 instances when they are not being used due to the AWS Free Tier EC2 limit of 750 hours per month.

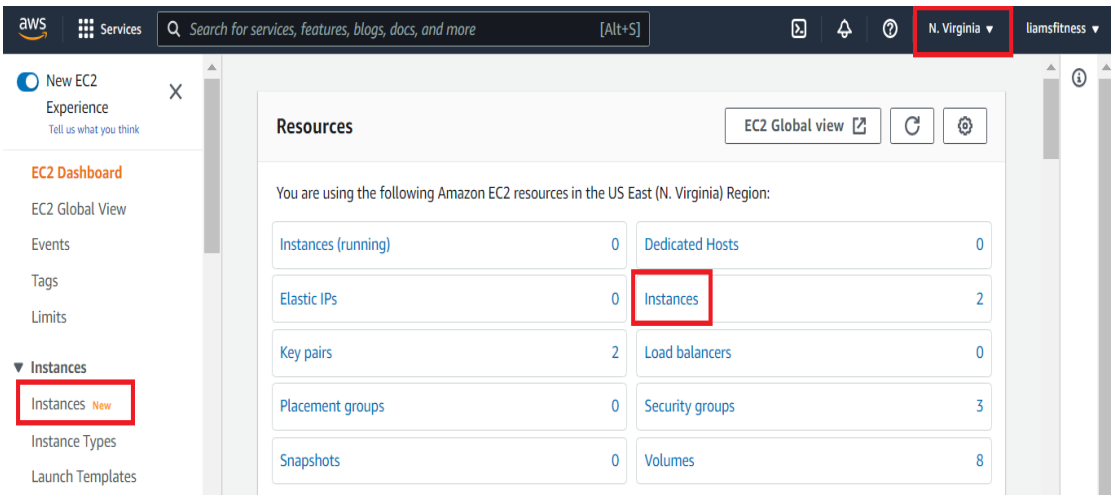
If you already know that your Ubuntu 20 EC2 instance is running, you can skip this step and go directly to **Update Virtual Private Cloud (VPC)**.

Otherwise, we need to ensure that our instance is running. Once on the **Console Home** screen, enter **EC2** in the search bar and select the 1st EC2 listing.

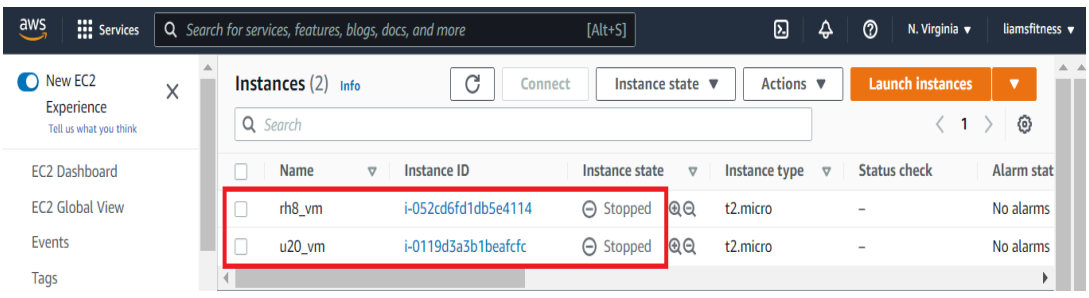


You will be brought to the **EC2** Dashboard. It contains links to the resources being used in the selected AWS region. In my case it's US East (N. Virginia).

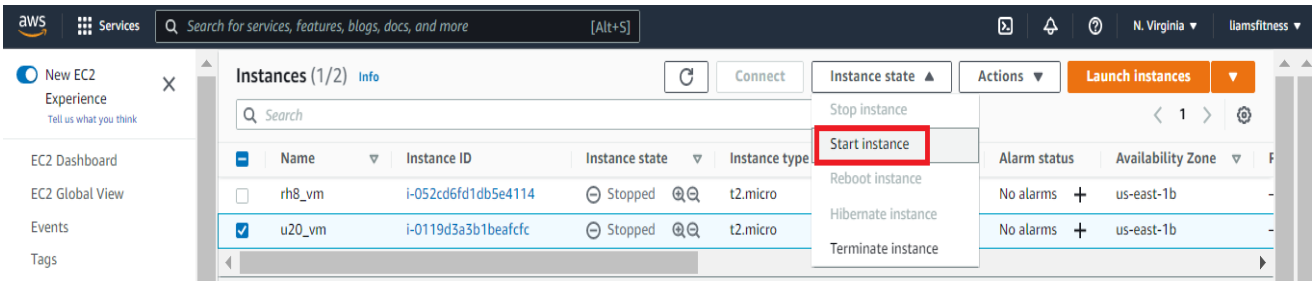
From the EC2 dashboard, click **Instances** (either link will work, your choice).



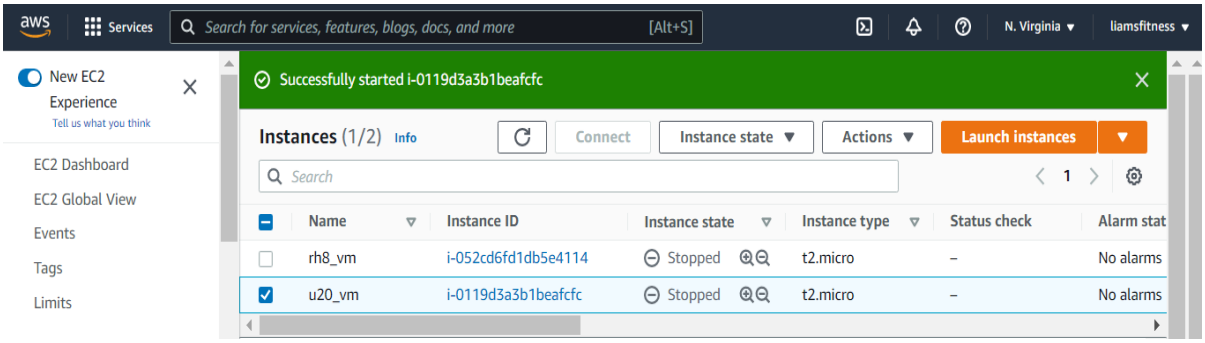
You will notice that both of my instances are in the stopped stated.



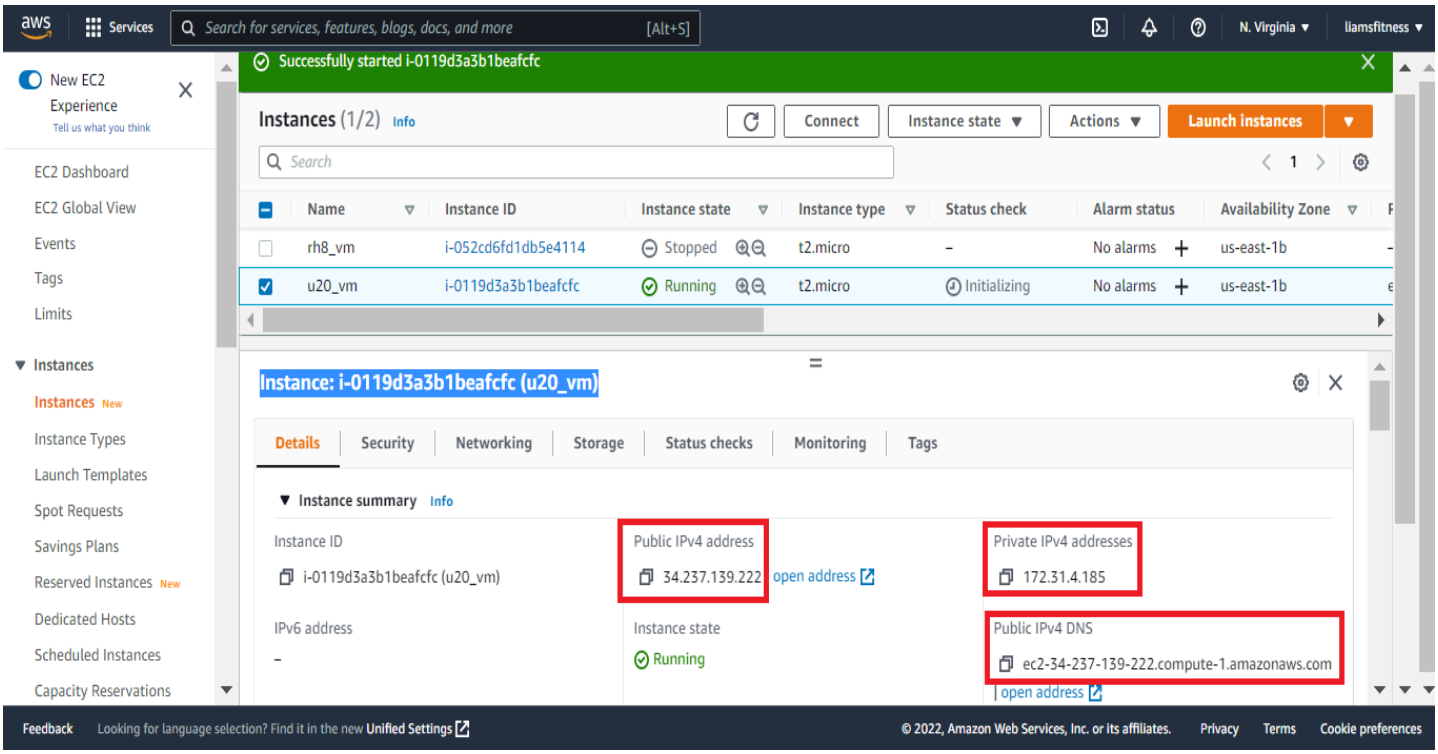
To start my Ubuntu 20 instance, I will ensure that my **u20_vm** is selected (checkbox), then I will click the **Instance state** drop-down menu and click **Start instance**.



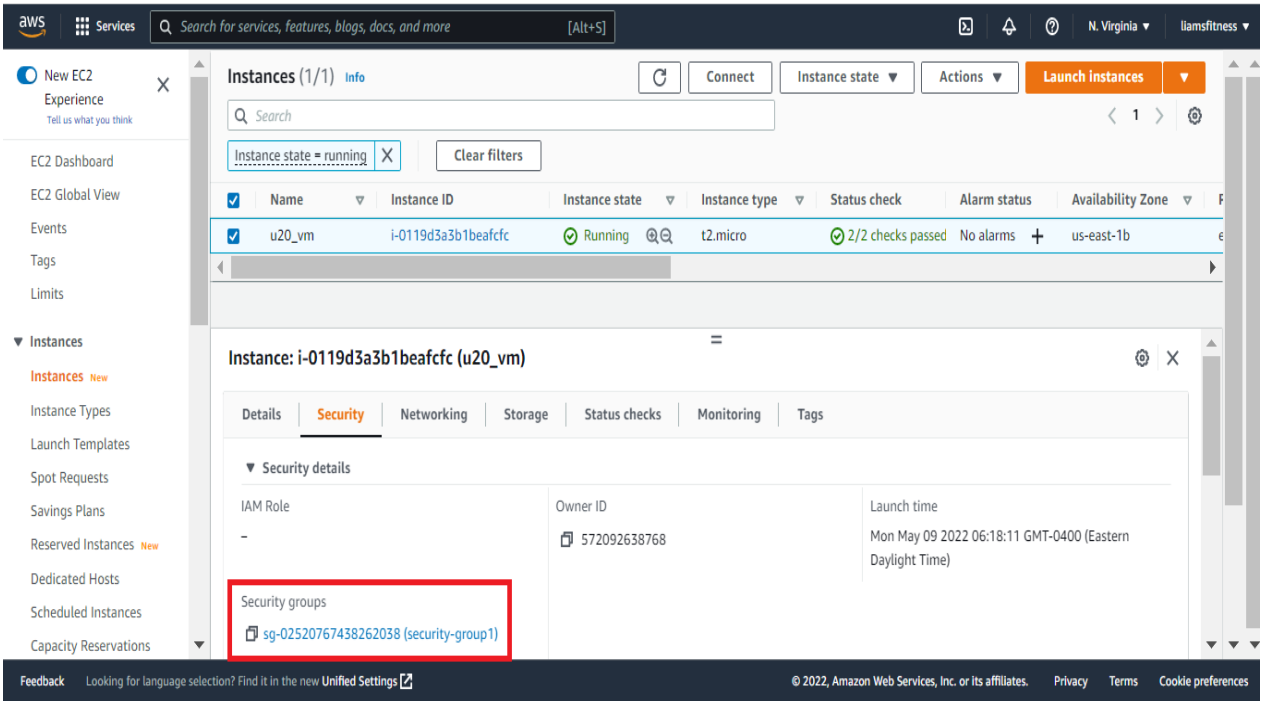
The instance will start and be given new connection details.



Ensure your new instance is selected (**u20_vm**), then on the **Details** tab, note the value for **Public IPv4 DNS**. I usually keep the instance's name, public IP, private IP and public IPv4 DNS stored for easy access. We will need this to connect to the instance.



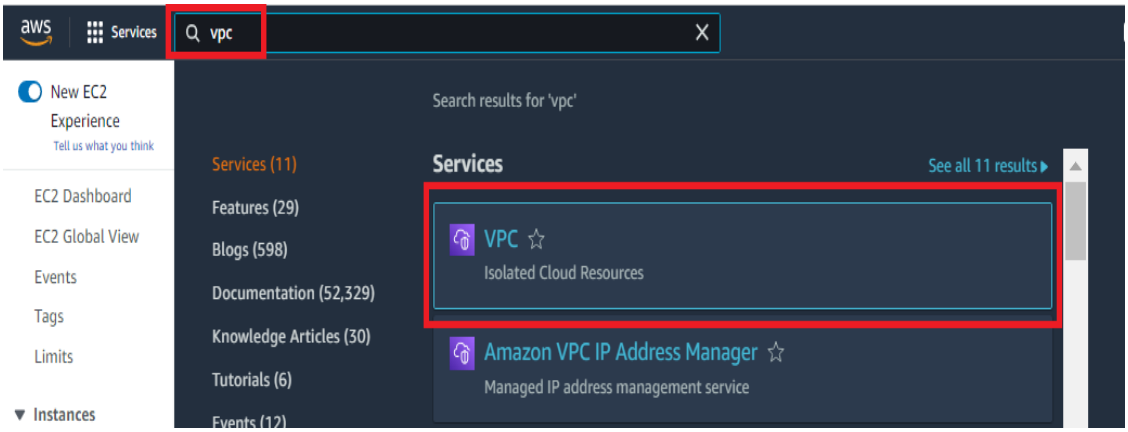
Next, click the **Security** tab and note the Security Group associated with the Ubuntu 20 EC2 instance. In my case, it is **security-group1**.



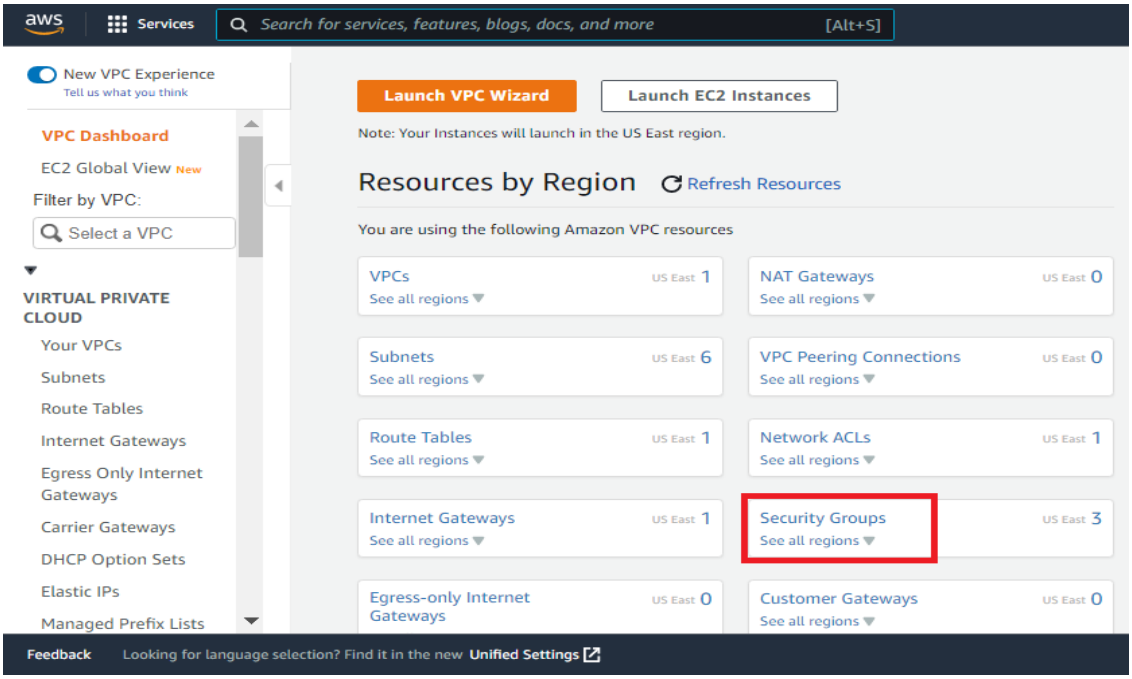
Before we install Nginx, we will need to open a port in our firewall to be able to access the website we create after the Nginx install. To do this we will configure our [VPC (Virtual Private Cloud)] [Security Group] to allow HTTP traffic through the firewall. This will allow access to the Nginx web server that will be running on our Ubuntu 20 EC2 instance in our isolated virtual network (VPC).

Update Virtual Private Cloud (VPC)

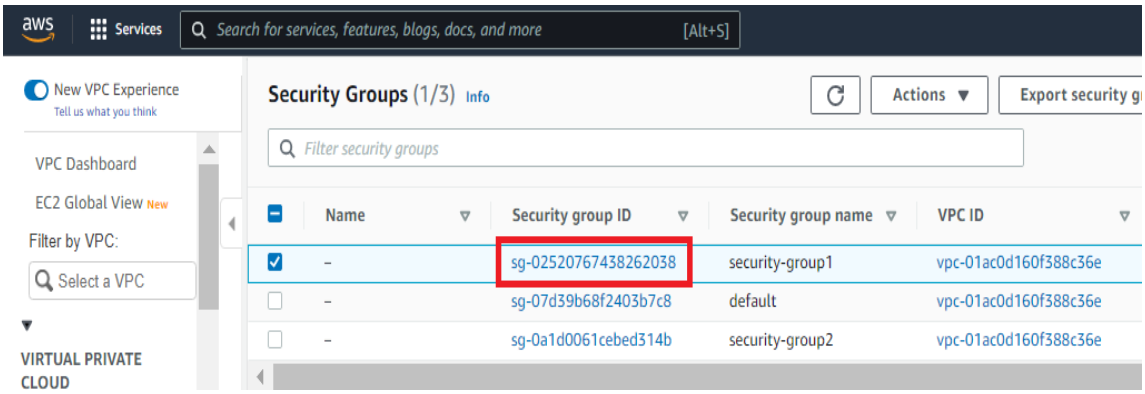
From the EC2 dashboard, enter VPC in the search bar and select **VPC** Isolated Cloud Resources.



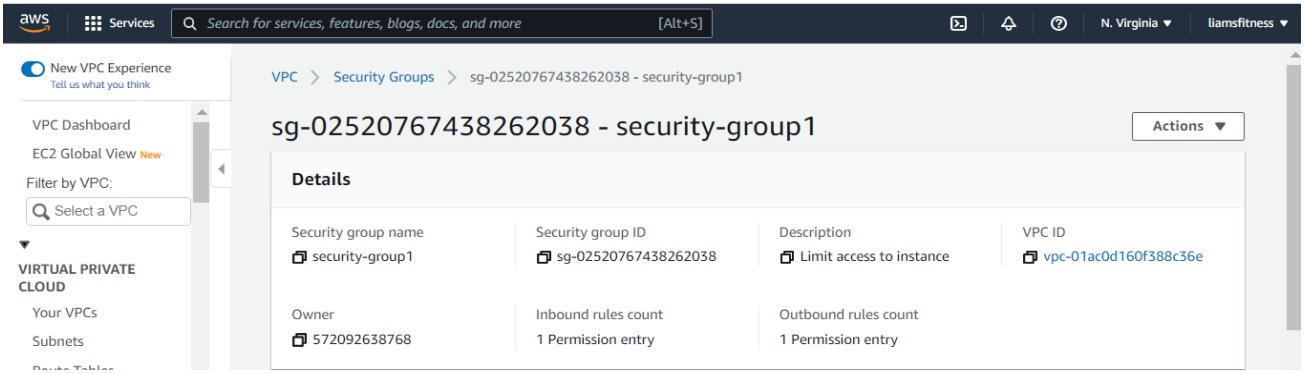
On the VPC Dashboard, select **Security Groups**.

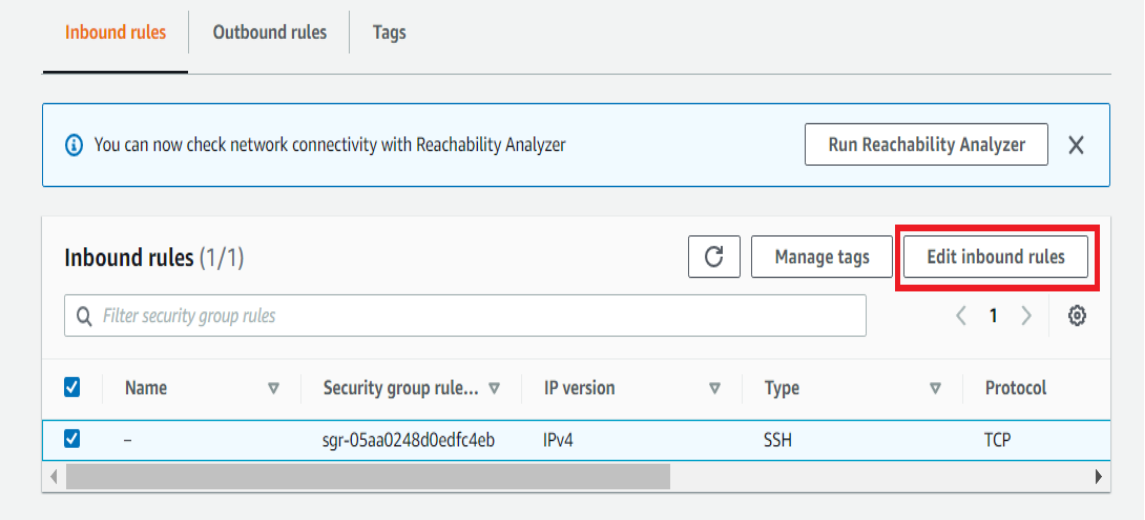


On the **Security Groups** screen, click the **Security group ID** that is associated with your Ubuntu 20 EC2 instance.

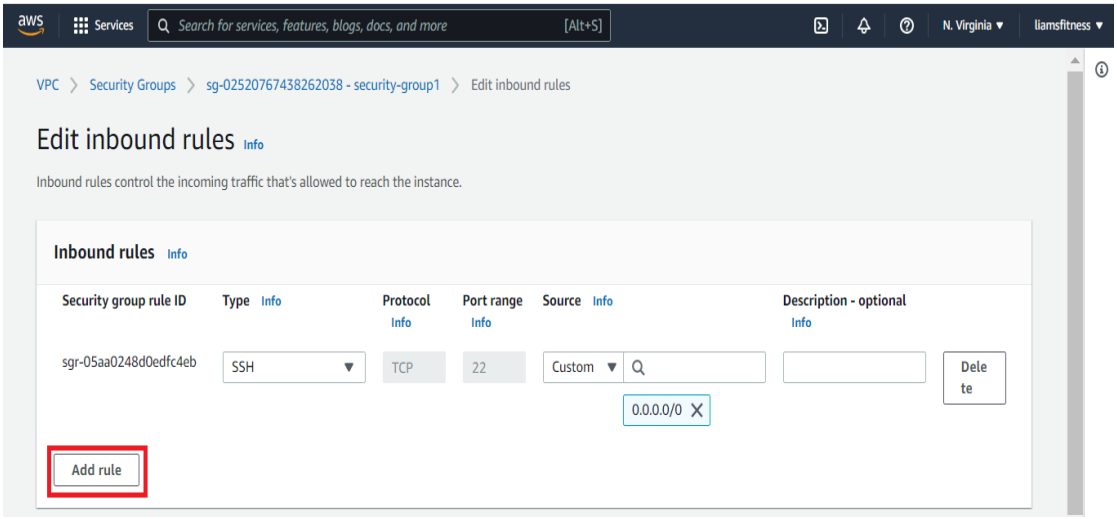


On the security group screen, scroll down and ensure the **Inbound Rules tab** is selected, then click **Edit Inbound Rules**

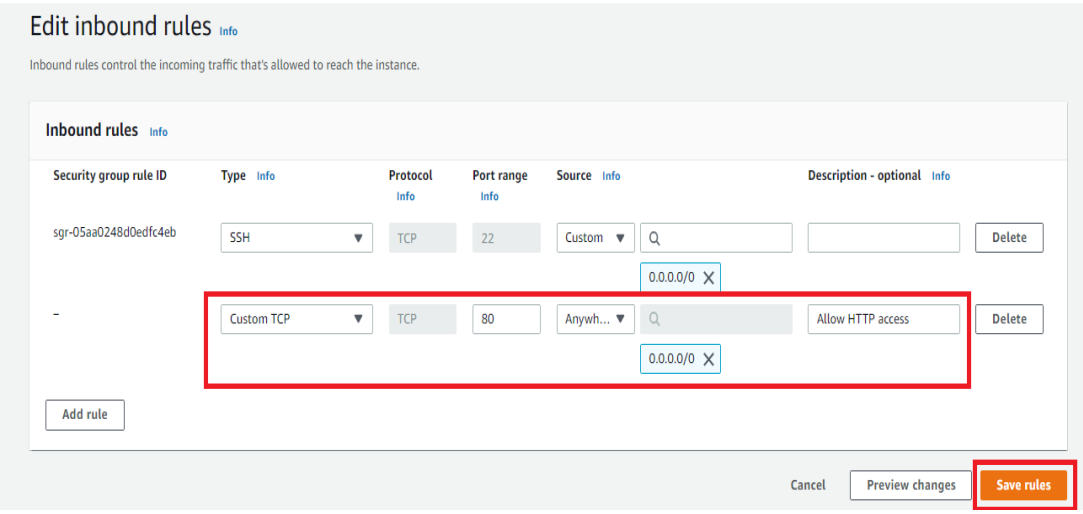


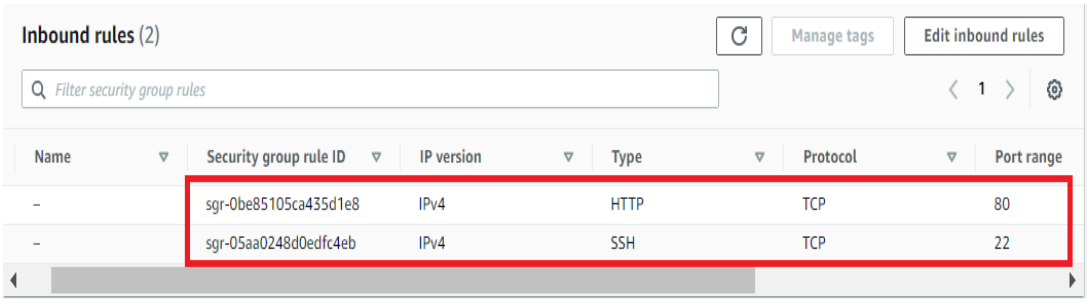


On the **Edit Inbound Rules** screen, click the **Add rule** button



Ensure **Port range** is set to **80** (HTTP) and **Source** is set to **0.0.0.0/0** (Anywhere-IPv4) and click **Save rules**



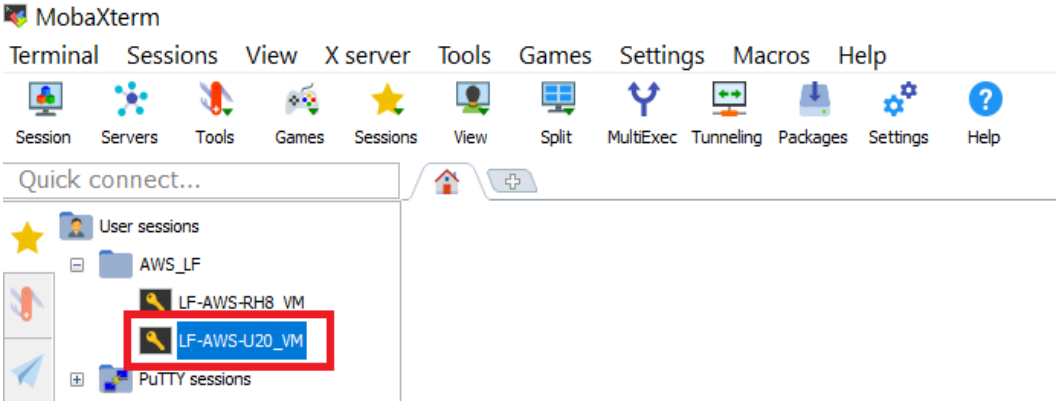


Now that we've opened the firewall for HTTP access, we can connect to the Ubuntu 20 instance and install the Nginx web server.

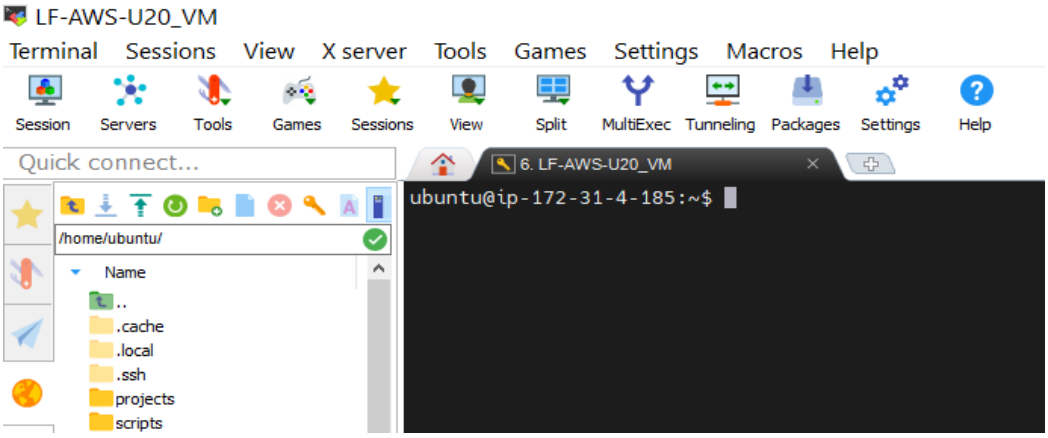
Connect to Ubuntu 20

As I mentioned in my **Create AWS Ubuntu 20 EC2 Instance** tutorial, accessible **here,** I am using **MobaXterm Portable** as my SSH client. Since I restarted my Ubuntu 20 EC2 instance, the connection details changed and I had to update my saved SSH session by changing the **Remote Host:** to the updated **Public IPv4 DNS** of my Ubuntu 20 instance.

After updating my saved SSH connection, from the MobaXterm main interface, I double clicked my SSH session to connect to my Ubuntu 20 EC2 instance.



After the session opened, I executed the **`clear`** command to clear the terminal.



We are now ready to install the Nginx web server.

Install Nginx Web Server

The Nginx web server package is available in Ubuntu's repositories. To install it, issue the following:

```
$ sudo apt install nginx -y
```

```
ubuntu@ip-172-31-4-185:~$ sudo apt install nginx -y
```

The Nginx web server runs automatically after installation. We will confirm this with the following:

```
$ sudo systemctl status nginx
```

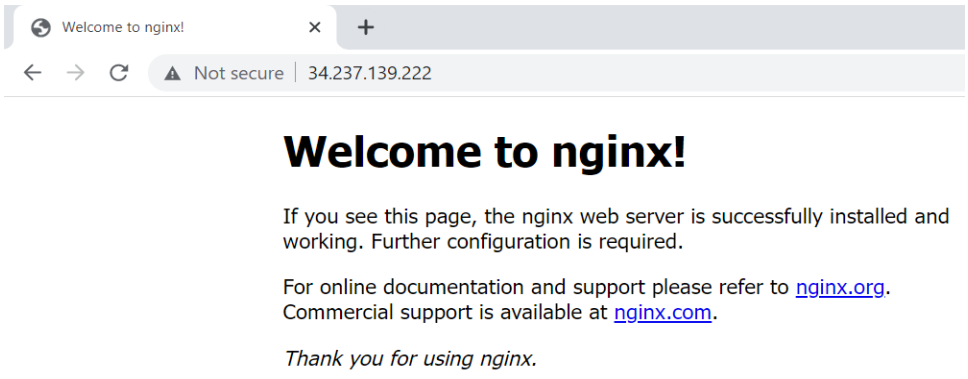
```
Setting up libnginx-mod-http-image-filter (1.18.0-0ubuntu1.3) ...
Setting up nginx-core (1.18.0-0ubuntu1.3) ...
Setting up nginx (1.18.0-0ubuntu1.3) ...
Processing triggers for ufw (0.36-6ubuntu1) ...
Processing triggers for systemd (245.4-4ubuntu3.15) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.7) ...
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo systemctl status nginx
● 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) since Mon 2022-05-09 11:49:10 UTC; 1min 25s ago
       Docs: man:nginx(8)
   Main PID: 2109 (nginx)
      Tasks: 2 (limit: 1145)
     Memory: 4.2M
     CGroup: /system.slice/nginx.service
             ├─2109 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
             └─2110 nginx: worker process

May 09 11:49:10 ip-172-31-4-185 systemd[1]: Starting A high performance web server and a reverse proxy server...
May 09 11:49:10 ip-172-31-4-185 systemd[1]: Started A high performance web server and a reverse proxy server.
ubuntu@ip-172-31-4-185:~$
```

We will also verify whether, or not, the nginx service is enabled on system startup with the following:

```
$ sudo systemctl is-enabled nginx
```

```
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo systemctl is-enabled nginx
enabled
ubuntu@ip-172-31-4-185:~$
```

Since we've already added an inbound rule to allow HTTP traffic, we can test it out by using the **Public IPv4 address** associated with the Ubuntu 20 EC2 instance.
*(**NOTE**: **your Public IPv4 address will be different**).*



Configure Nginx to Host a Website

Now we will configure Nginx to host a basic website using a Nginx server block. This will override the default test page that was returned when we initially accessed the web server via the public IP address of our AWS Ubuntu 20 instance.

To begin, we will create a directory to store the website content with the following:

```
$ sudo mkdir -p /var/www/test_site/html
```

Next, we will create a basic html page at the following location:

```
sudo vi /var/www/test_site/html/index.html
```

This page will be rendered when we access the site.

```
<html>
    <head>
        <title>Welcome to my Test Site!</title>
    </head>
    <body>
        <p>It works! Thanks for visiting my
             <b><span style="color:blue">Test Site</span></b>!
         </p>
    </body>
</html>
```

Next, create the following file: **`/etc/nginx/sites-available/test_site`**

This file contains a Nginx server block using the public IP address of my AWS Ubuntu 20 instance.

*(**NOTE**: your **Public IPv4 address** will be different).*

```
server {
  listen 80;
  server_name 3.227.9.214; # refers to Public IPv4 address of AWS instance
  root /var/www/test_site/html;
}
```

```
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo mkdir -p /var/www/test_site/html
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ cat /var/www/test_site/html/index.html
<html>
    <head>
        <title>Welcome to my Test Site!</title>
    </head>
    <body>
        <p>It works! Thanks for visiting my <b><span style="color:blue">Test Site</span></b>!</p>
    </body>
</html>
ubuntu@ip-172-31-4-185:~$ cat /etc/nginx/sites-available/test_site
server {
  listen 80;
  server_name 3.227.9.214;
  root /var/www/test_site/html;
}
ubuntu@ip-172-31-4-185:~$
```

Link this file with the Nginx enabled sites directory to ensure that this website will be served when accessed.

```
$ sudo ln -s /etc/nginx/sites-available/test_site /etc/nginx/sites-enabled/
```

Verify that the server block in the configuration file is valid with the following:

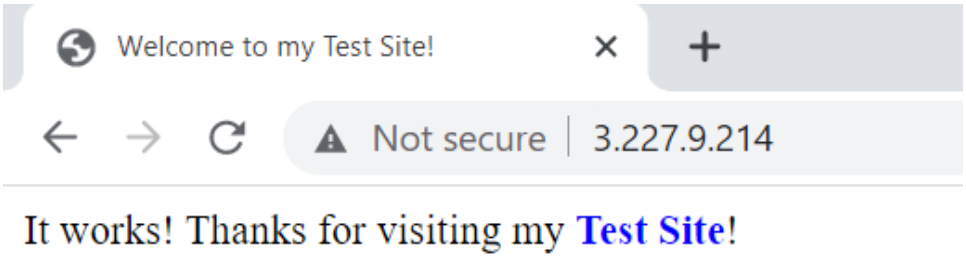
```
$ sudo nginx -t
```

Now, restart the Nginx service

```
$ sudo systemctl restart nginx
```

```
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo ln -s /etc/nginx/sites-available/test_site /etc/nginx/sites-enabled/
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ ls -l /etc/nginx/sites-enabled/
total 0
lrwxrwxrwx 1 root root 34 May  9 11:49 default -> /etc/nginx/sites-available/default
lrwxrwxrwx 1 root root 36 May 11 10:31 test_site -> /etc/nginx/sites-available/test_site
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ sudo systemctl restart nginx
ubuntu@ip-172-31-4-185:~$
```

Open your browser and enter **your** instance's public IP address as the URL to confirm that it's working.



Please note that if you shutdown your Ubuntu 20 AWS instance, the connection details will change (**Public IPv4 address** & **Public IPv4 DNS**). Therefore, the corresponding change will need to be made in:

/etc/nginx/sites-enabled/test_file

```
server {
  listen 80;
  server_name 3.227.9.214; # refers to Public IPv4 address of AWS instance
                           # MUST UPDATE after restarting AWS instance
  root /var/www/test_site/html;
}
```

We've successfully installed the Nginx web server and configured it to host a single website using the Ubuntu 20 instance's IP address. To host more than one site, we would need to use domain names. I will be demonstrating this process in a future tutorial.

I hope you've enjoyed this tutorial.

I have another tutorial where I install the Nginx web server on my **RHEL 8** EC2 (Elastic Compute Cloud) instance, accessible **here**. Although the Nginx installations are practically identical, I wanted to have two different instances for testing purposes. This will enable me to test my shell scripts that monitor services locally, as well as, remotely. If you're interested, the scripts can be accessed via my Linux tutorials page, **here**, while my main tutorials page is accessible **here**.

Please note that the free tier allows for 750 hours per month of Amazon EC2. You can create many EC2 instances but beware of the limit. If you go over that limit, you will pay the cost. My advice to you is to shutdown your instance/s after you've done your work.

Back to Top