|  |
| --- |
| AWARE DASHBOARD SETUP INSTRUCTIONS ON AWS |
| Version 2.0 |
|  |

TABLE OF CONTENTS

[**1** **INTRODUCTION 5**](https://docs.google.com/document/d/1F_eyJYlwOFsRL6pgkvfnVPP6bAImqKiY6PtJRzcX1IQ/edit#heading=h.1hmsyys)

[1.1 references 5](https://docs.google.com/document/d/1F_eyJYlwOFsRL6pgkvfnVPP6bAImqKiY6PtJRzcX1IQ/edit#heading=h.41mghml)

[**2**](#_tyjcwt) **SETTING UP THE SERVER ON AWS 5**

[2.1](#_3dy6vkm) Create Security Group 5

[2.2](#_1t3h5sf) Create aN AWS ROLE 7

[2.3](#_4d34og8) Launch Instance 9

[2.4](#_2s8eyo1) Create and Assign an Elastic IP Address 12

[2.5](#_17dp8vu) Get A Name assigned to the IP on the DNS server 14

[**3**](#_3rdcrjn) **INSTALL REMAINING ELEMENT OF THE LAMP STACK 14**

[3.1](#_26in1rg) Install Apache 15

[3.2](#_lnxbz9) Install MYSQL and Setup the database 15

[3.3](#_35nkun2) Install PHP 17

[**4**](#_1ksv4uv) **SETTING UP THE AWARE DASHBOARD 17**

[4.1](#_44sinio) Pulling down the latest version of the aware Server 18

[4.2](#_2jxsxqh) Certbot and SSL Cert Install 18

[4.3](#_z337ya) Securing the AWARE Dashboard 19

[4.4](#_3j2qqm3) set up public certificates in the public folder for the http virtual host 21

[4.5](#_1y810tw) MAKE adjustments to PHP.ini 21

[**5**](#_4i7ojhp) **MYSQL CONFIGURATION 22**

[5.1](#_2xcytpi) Copy LetsEncrypt certificates and allow access to them by MySQL 22

[5.2](#_1ci93xb) Create a MySQL database and DB USER for the AWARE Dashboard 22

[5.3](#_3whwml4) Load AWARE Dashboard core database 23

[5.4](#_2bn6wsx) Set MySQL configuration on your AWARE Dashboard 23

[**6**](#_qsh70q) **SET UP MOSQUITTO MQTT SERVER 23**

[6.1](#_3as4poj) Download and install Mosquitto 23

[6.2](#_1pxezwc) Download source and configure MySQL-Mosquitto adapter 24

[6.3](#_49x2ik5) Compile Mosquitto-MySQL adapter 24

[6.4](#_2p2csry) Configure Mosquitto to use SSL for all communications and authenticate all clients 27

[6.5](#_147n2zr) Installing Mosquitto-PHP library 28

[**7**](#_3o7alnk) **INSTALL THE ANDROID SDK 29**

[**8**](#_23ckvvd) **CONFIGURING AWARE DASHBOARD 29**

[8.1](#_ihv636) Add your server to Google OAUTH credentials 29

[8.2](#_32hioqz) Final AWARE Dashboard configuration 30

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 9/10/2018 | 1.0 | Initial Version of the Document | Abhijit |
| 4/21/2021 | 1.1 | Updates for Ubuntu 18 | D. Bellew |
| 10/7/2024 | 2.0 | Updates for Ubuntu 22, MariaDB | D. Bellew |
|  |  |  |  |
|  |  |  |  |

# **INTRODUCTION**

The purpose of this document is to outline the install steps followed to setup the aware dashboard so that UPenn can test out the aware framework. Due to network Issues faced in trying to setup this install in-house, a decision was taken to try and install the dashboard on AWS. The online guide located at http://www.awareframework.com/hosting-your-own-aware-dashboard/ was used as a basis for the install and this document

## REFERENCES

The following resources were leveraged to complete the install steps and test out the deployment

* Sharath Chandra Guntuku
* Salvatore Giorgi
* http://www.awareframework.com/hosting-your-own-aware-dashboard/

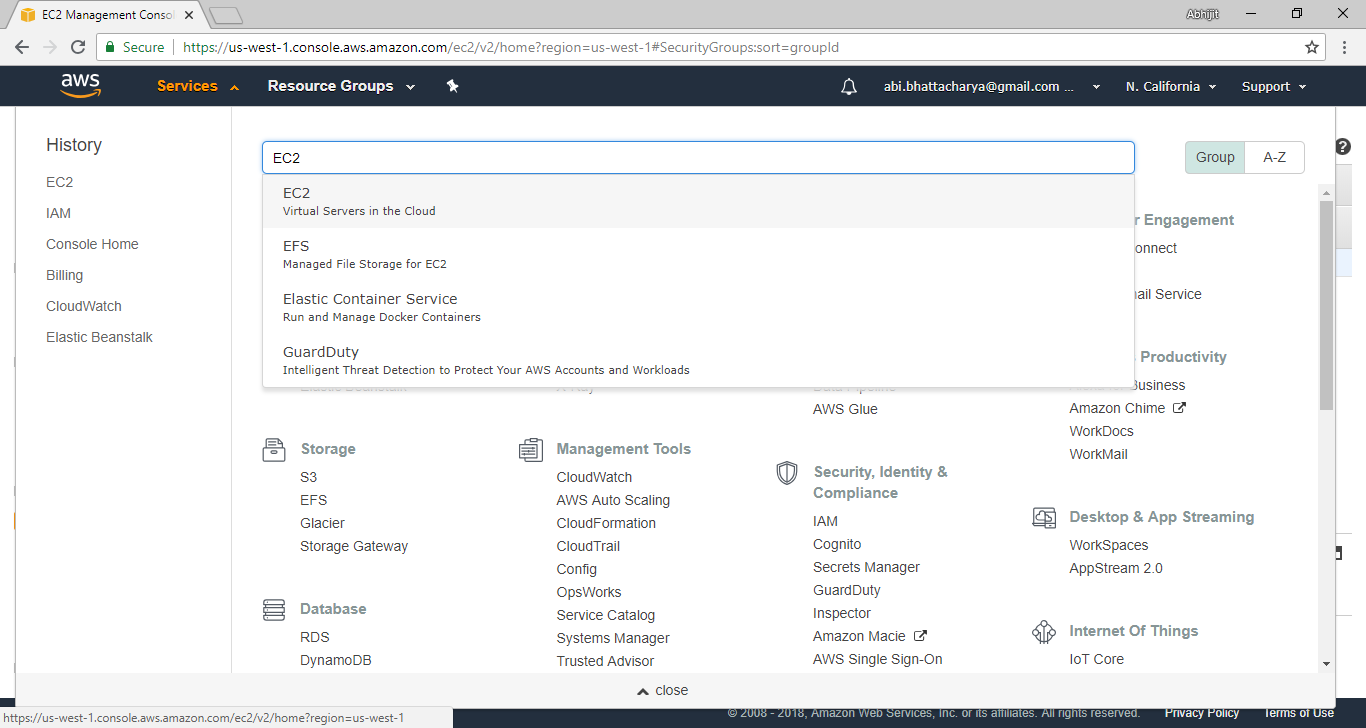
# **SETTING UP THE SERVER ON AWS**

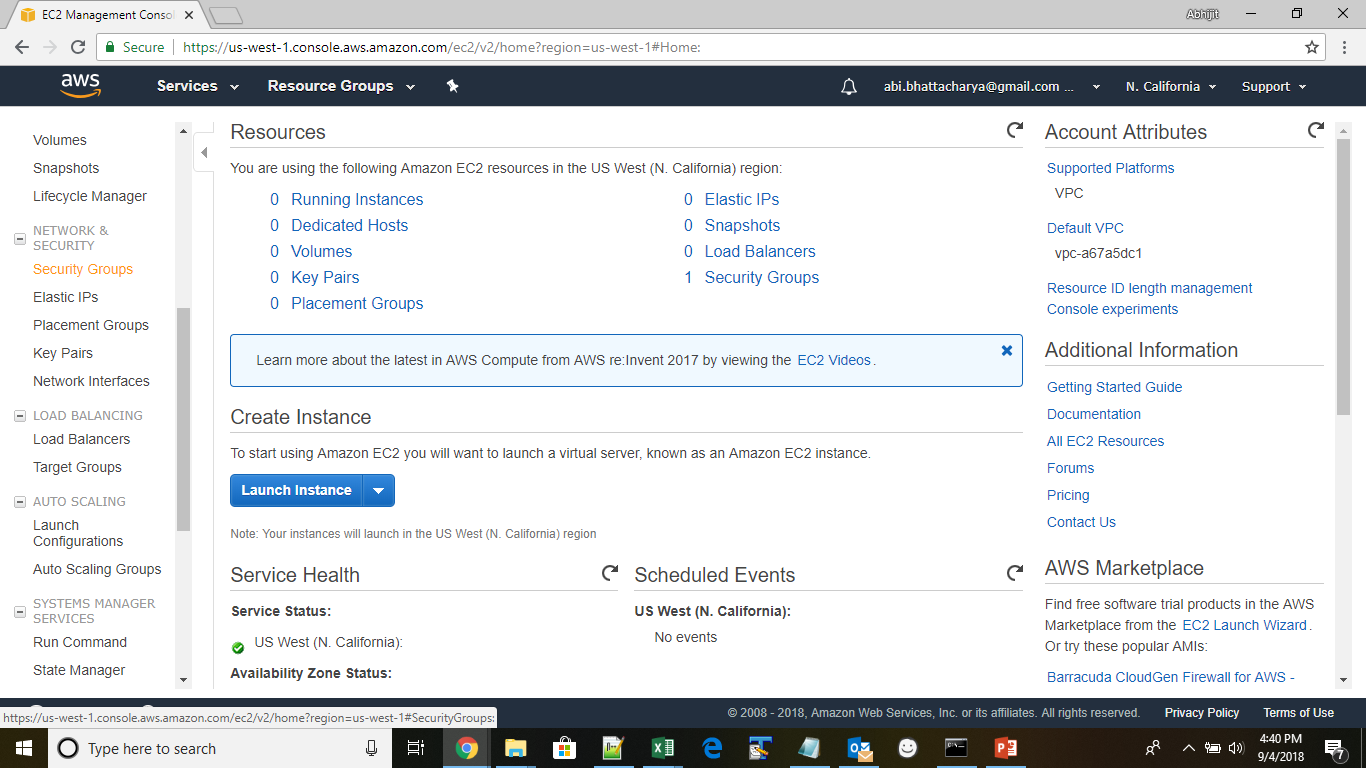
An Ubuntu 22.04 server was setup on AWS using the steps below.

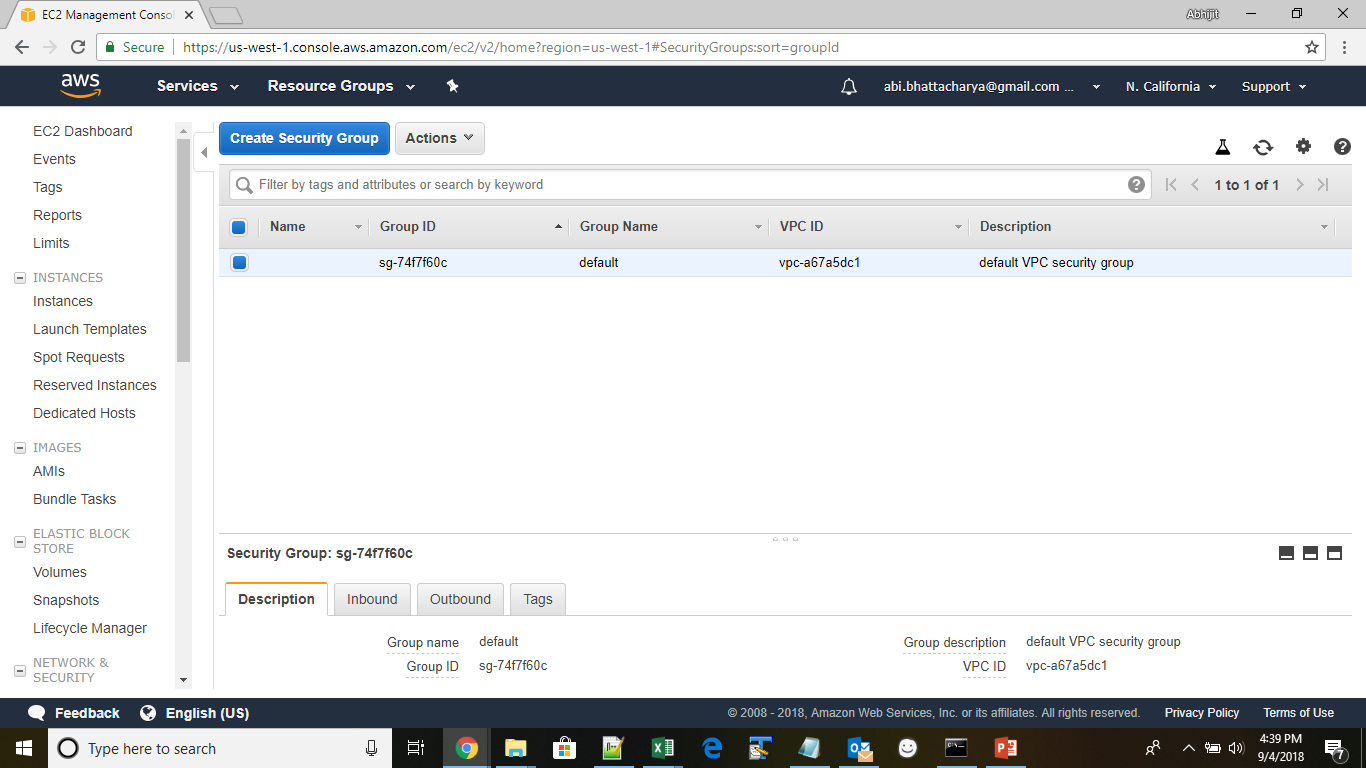
* (“lsb\_release -a” to show Ubuntu version)

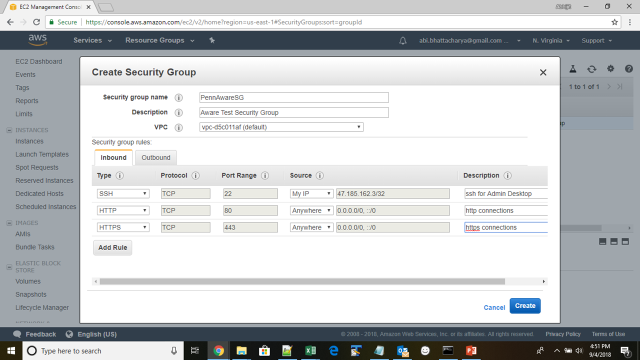
## CREATE SECURITY GROUP

A security group was created by going to the EC2 dashboard in the N. Virginia Region of AWS





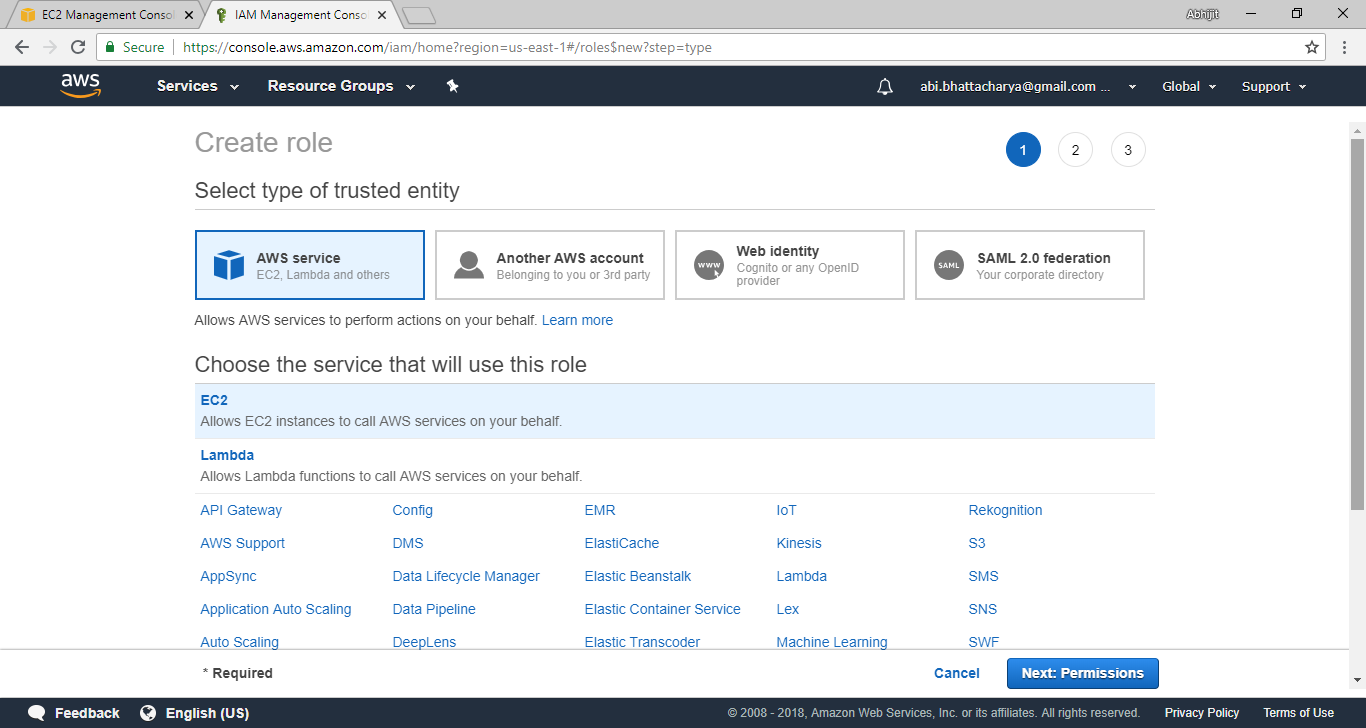


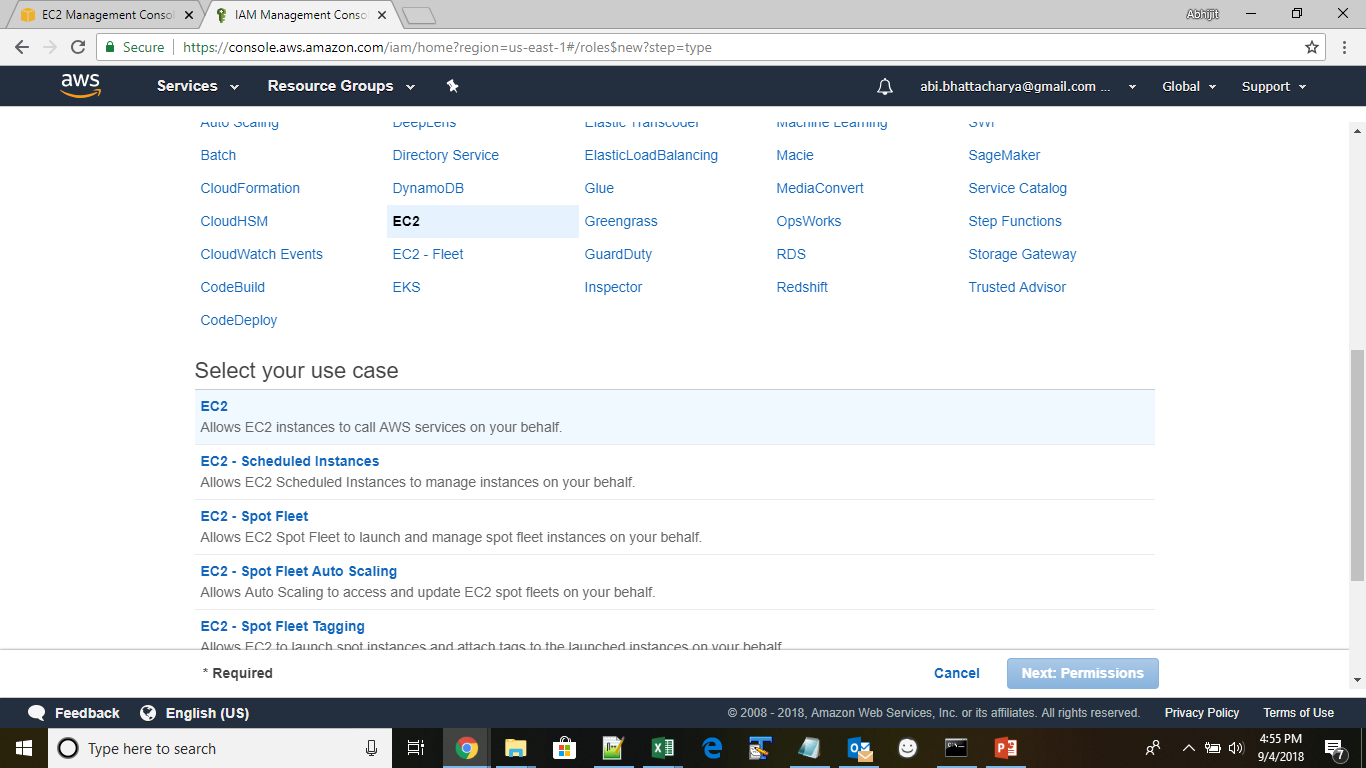


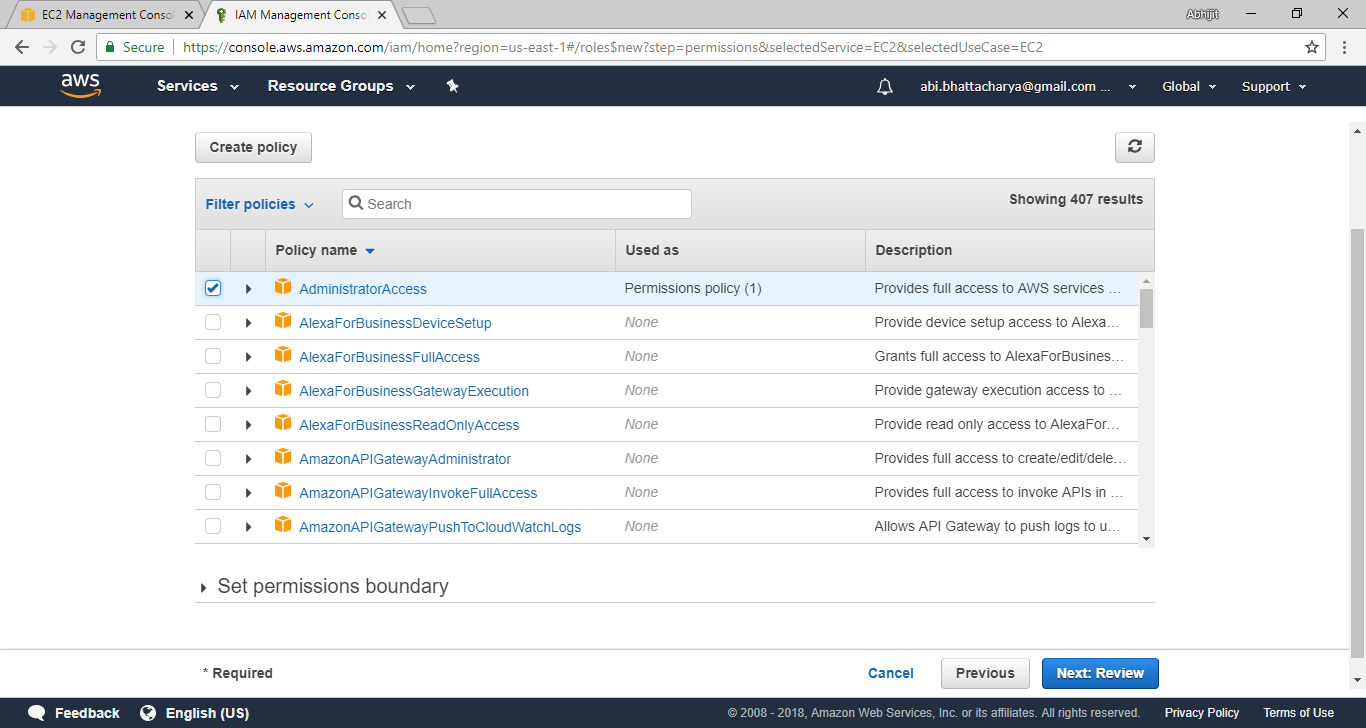
Select “Anywhere” in the source for SSH to allow anyone to SSH into the server and hit create to create the security group.

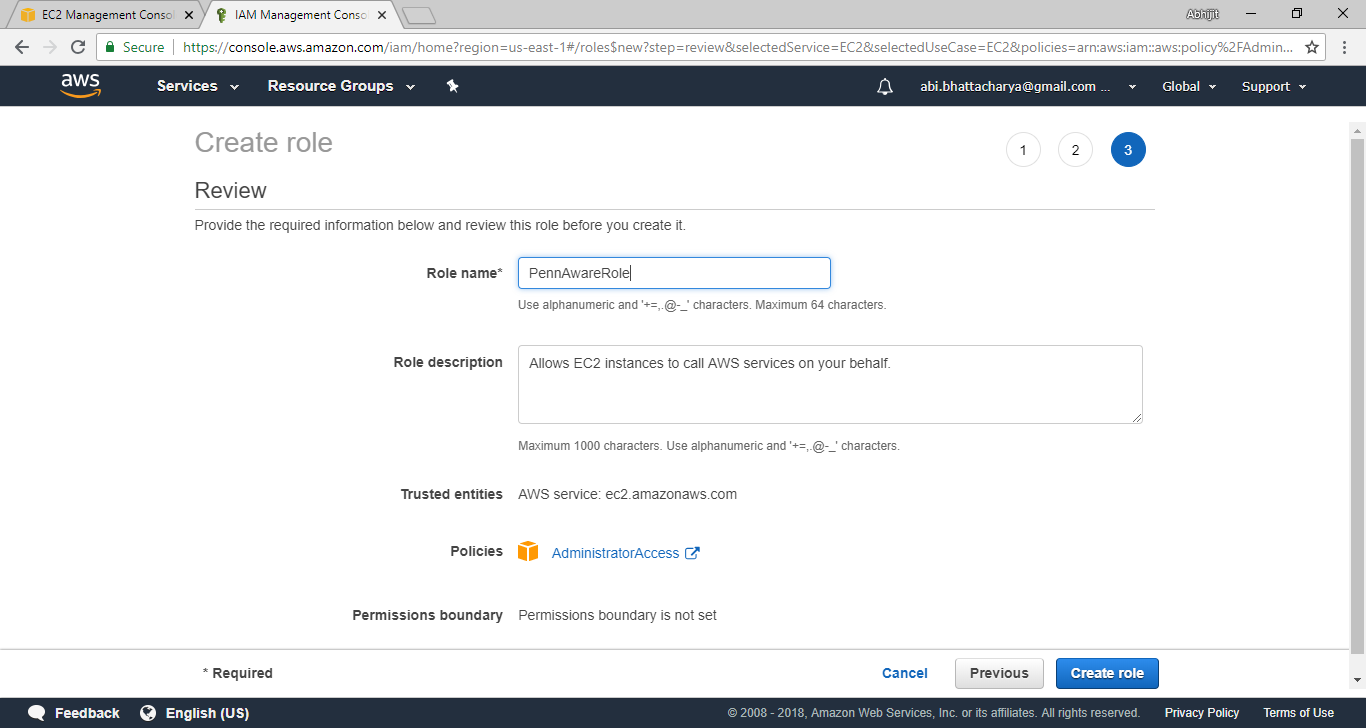
## CREATE AN AWS ROLE

Create a new role with admin access policy to be able to launch a new instance. Roles can be created under IAM service in AWS



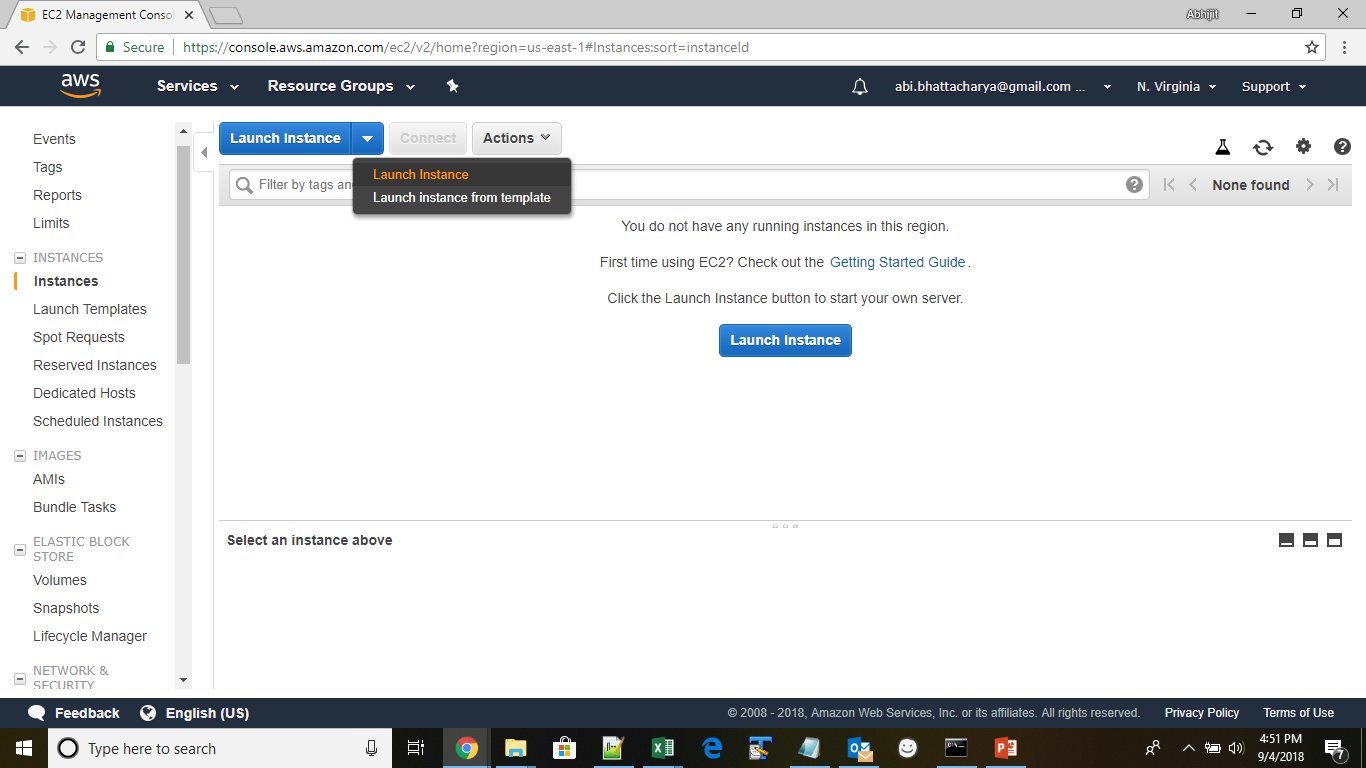


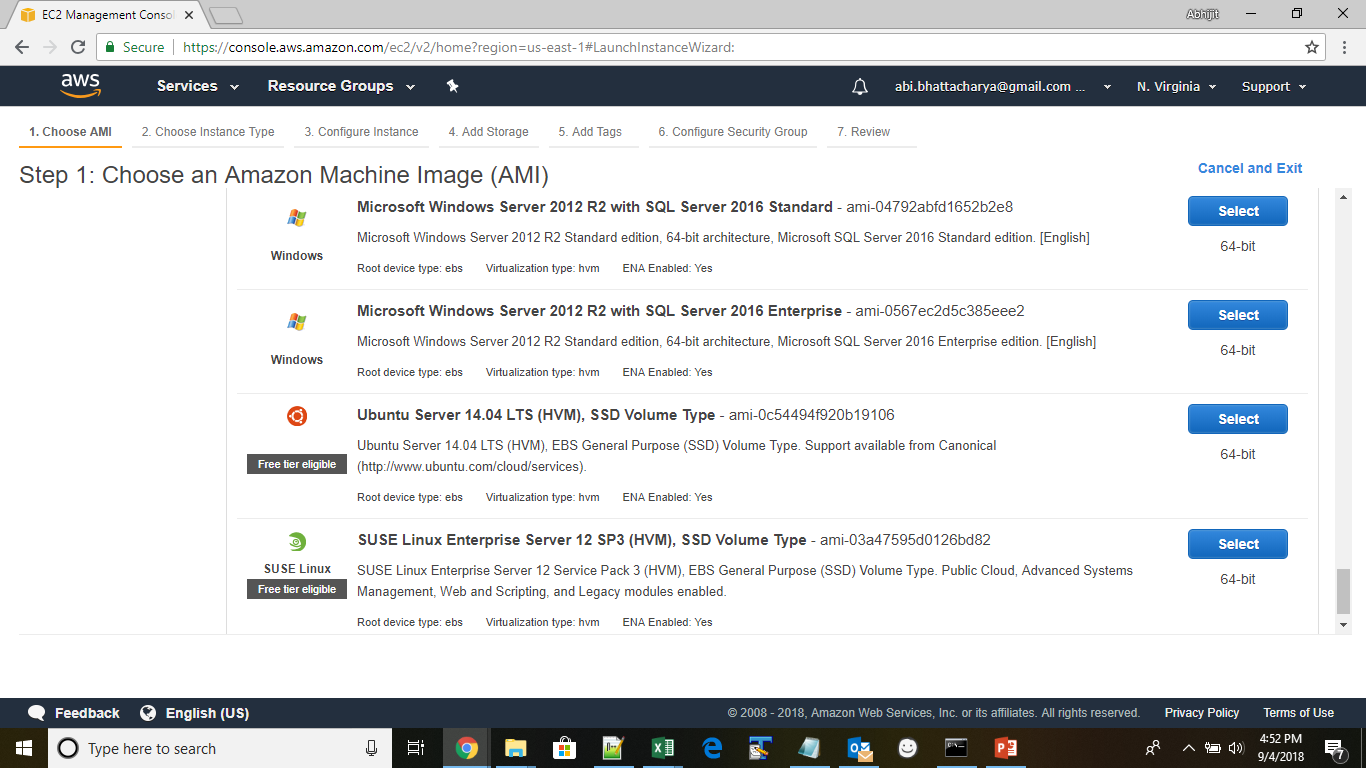




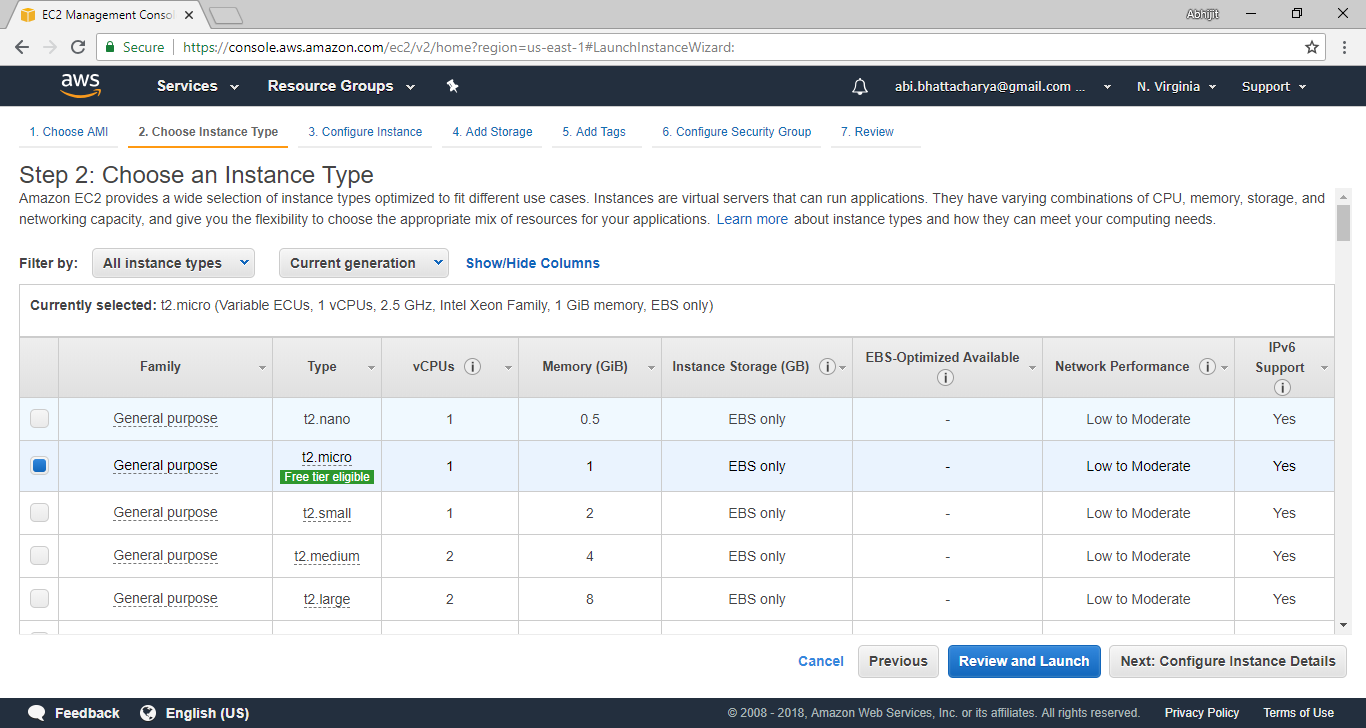
## LAUNCH INSTANCE

Launch a new Ubuntu 14.04 instance using the role and security group created

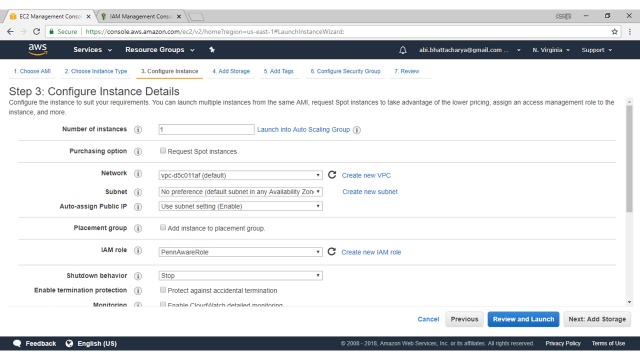




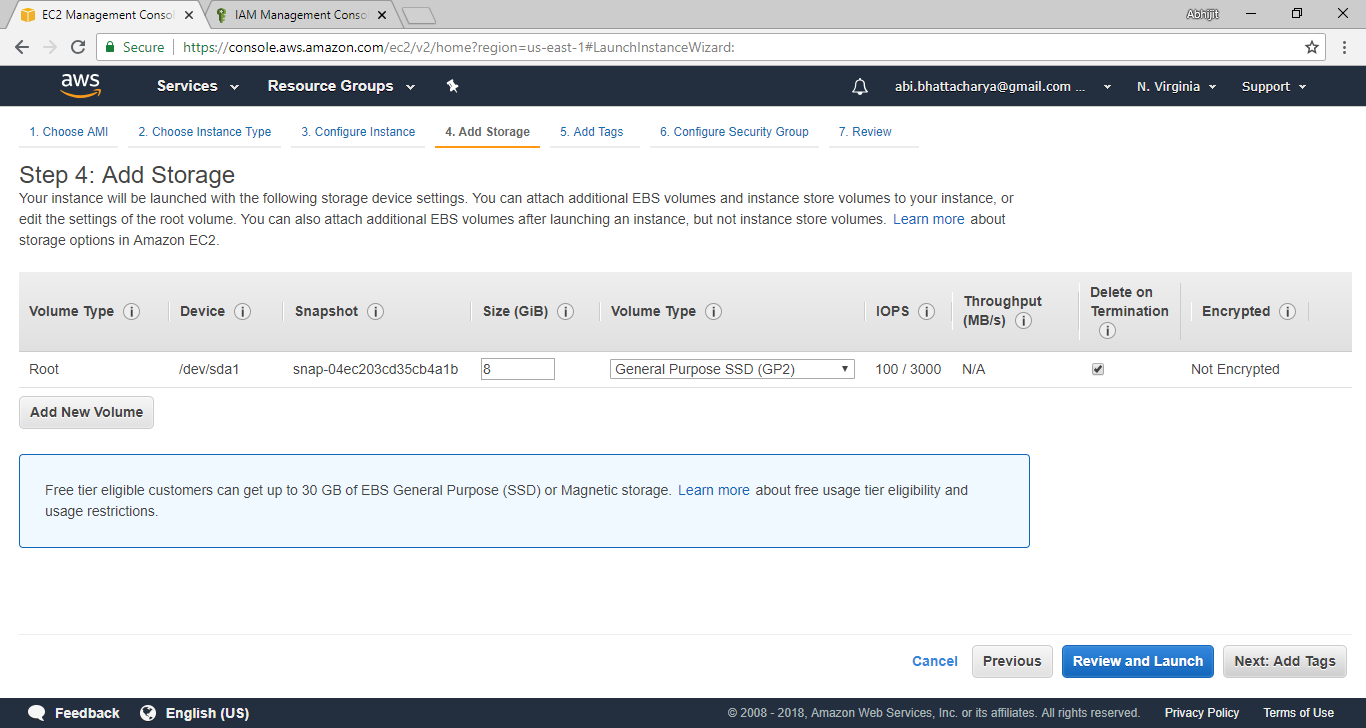
Choose a t2 micro instance



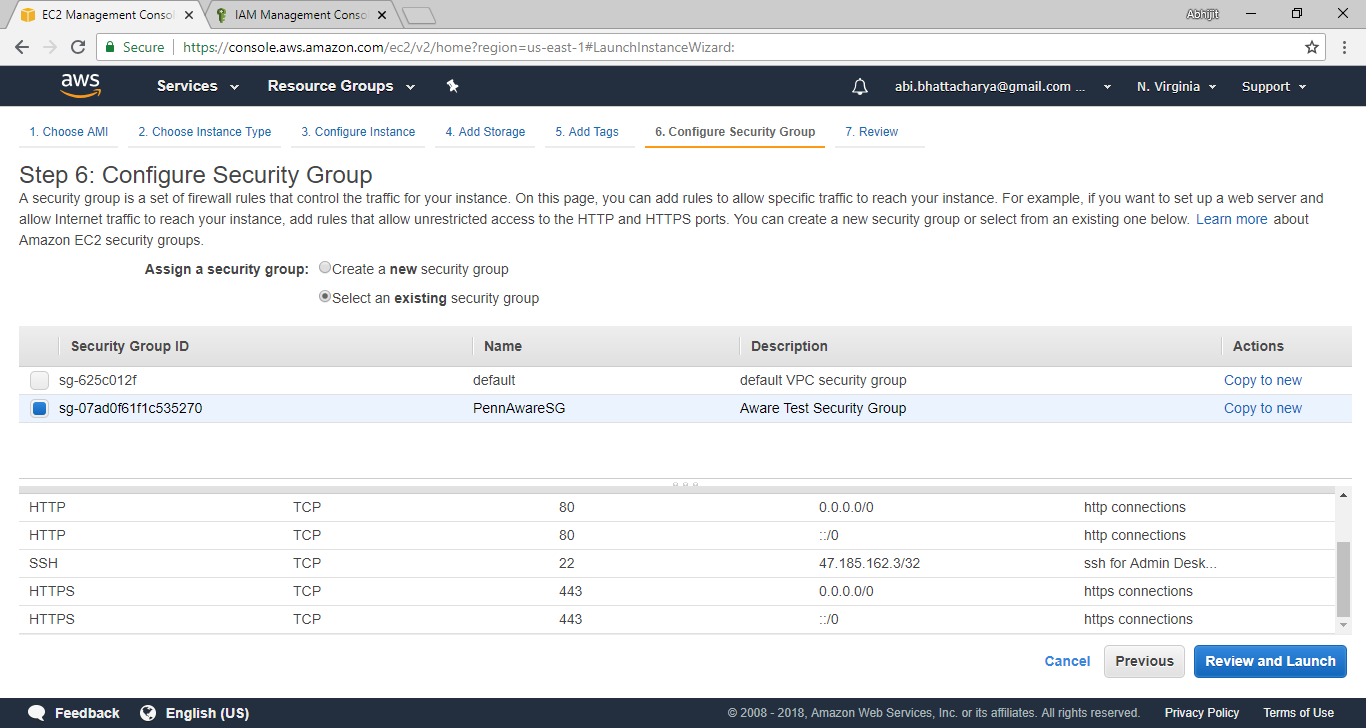
Use role created in instance details



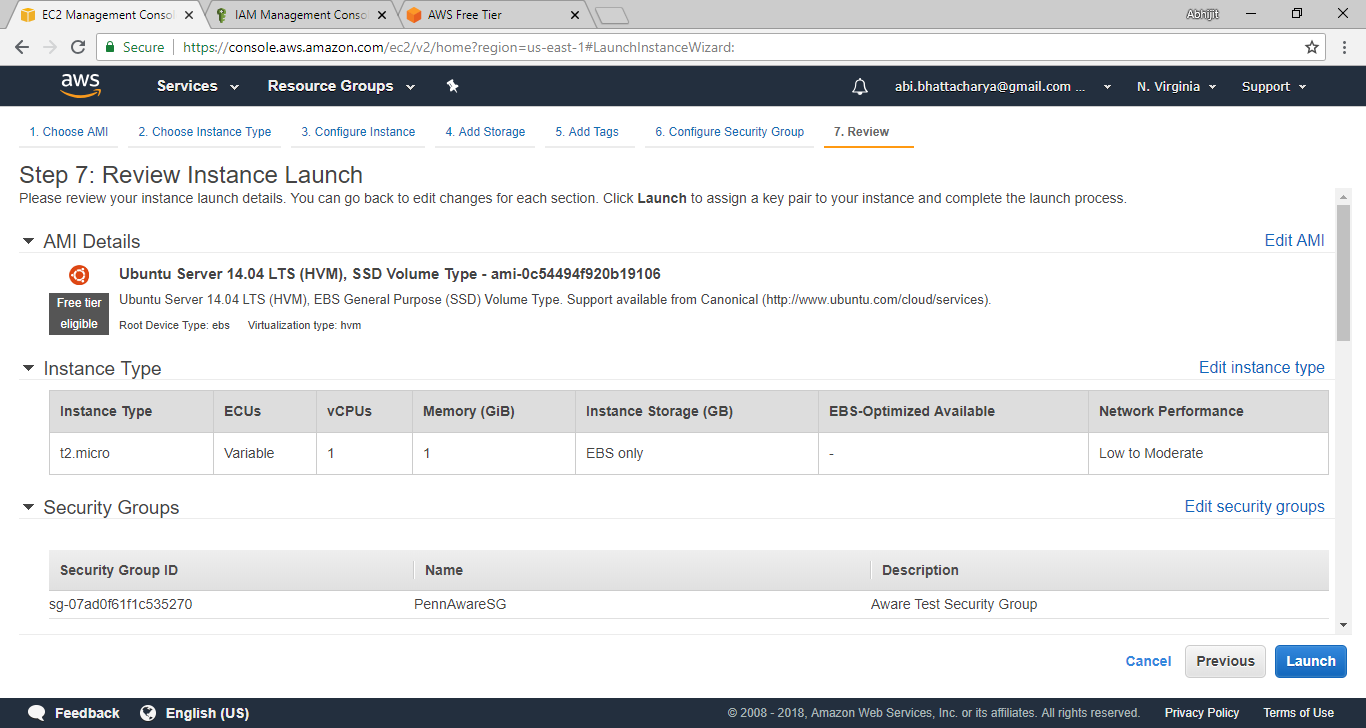
Change disk size to 16 GB

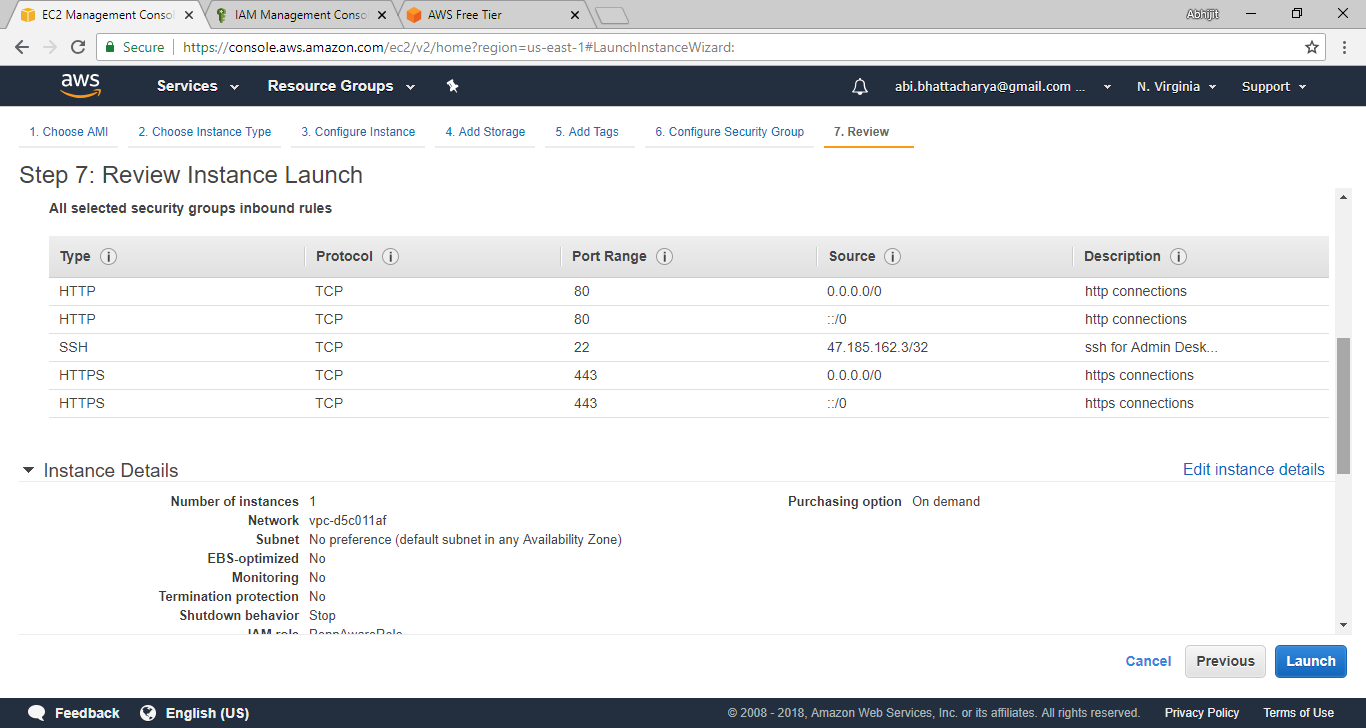


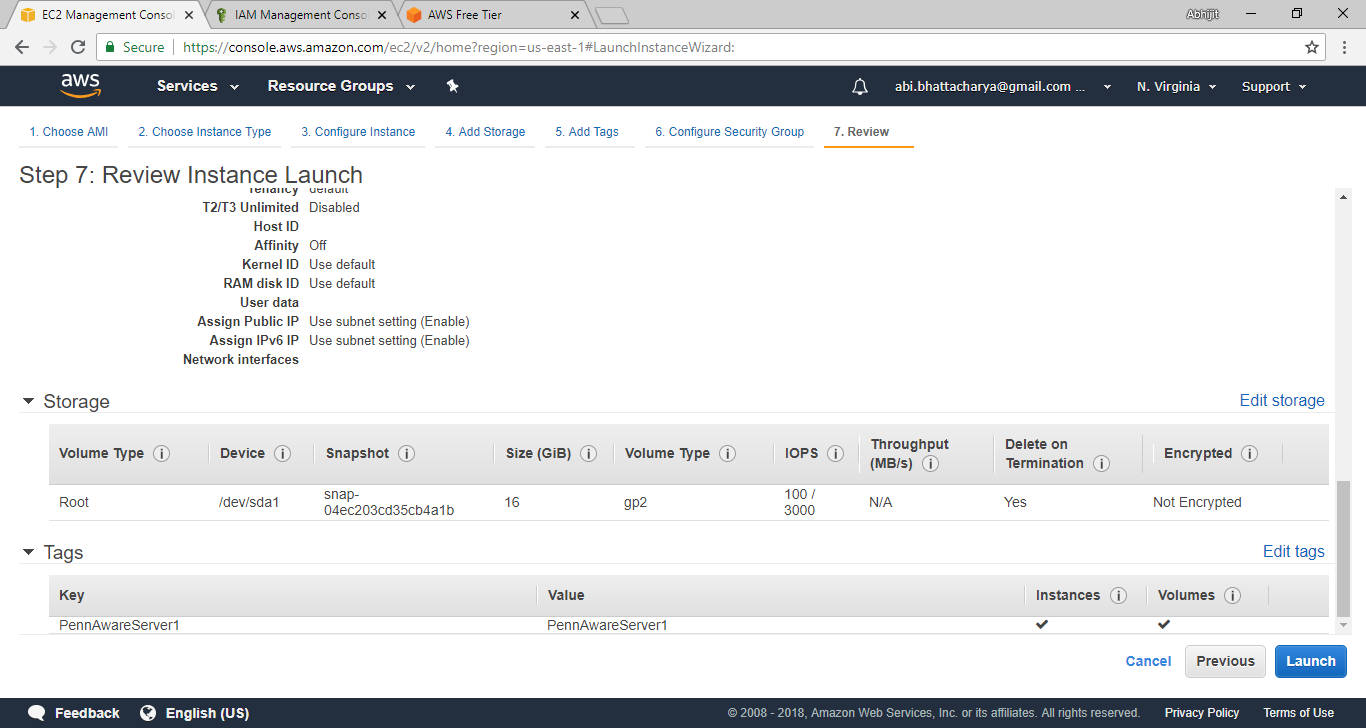
Pick the security group created previously



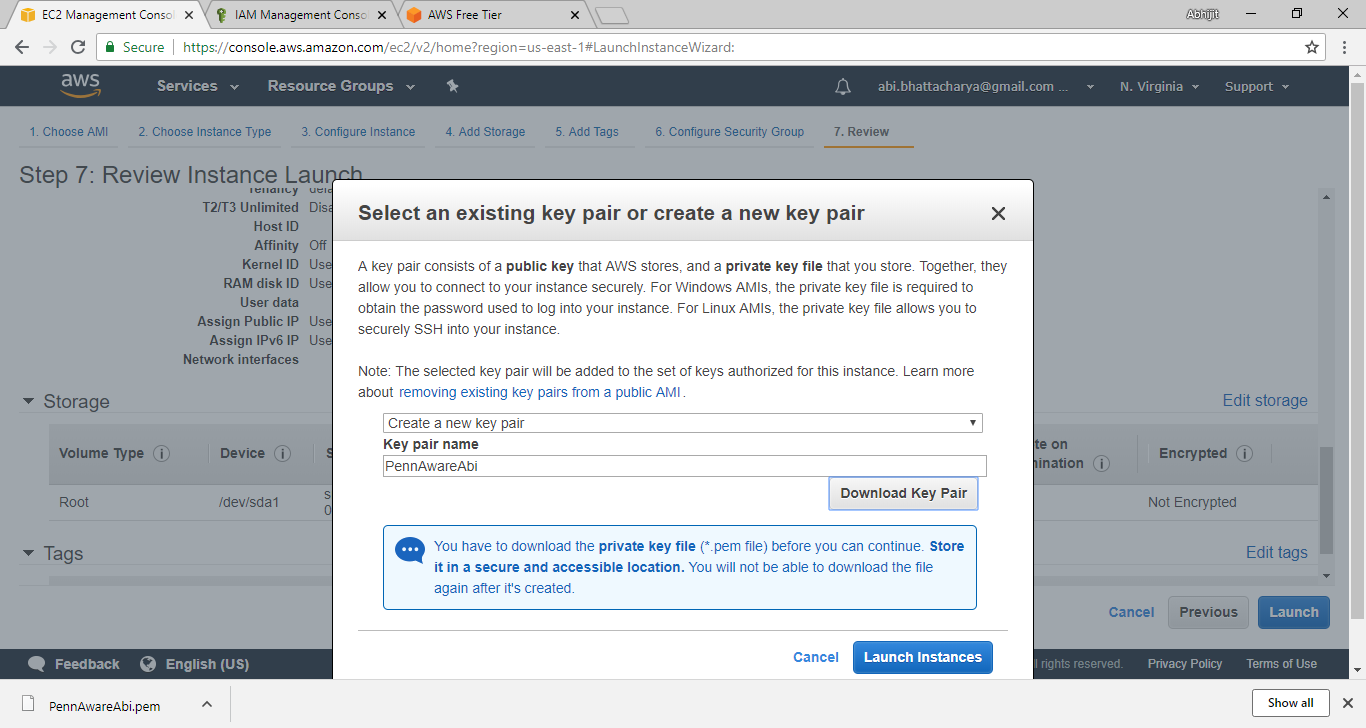
Launch the instance after review of settings

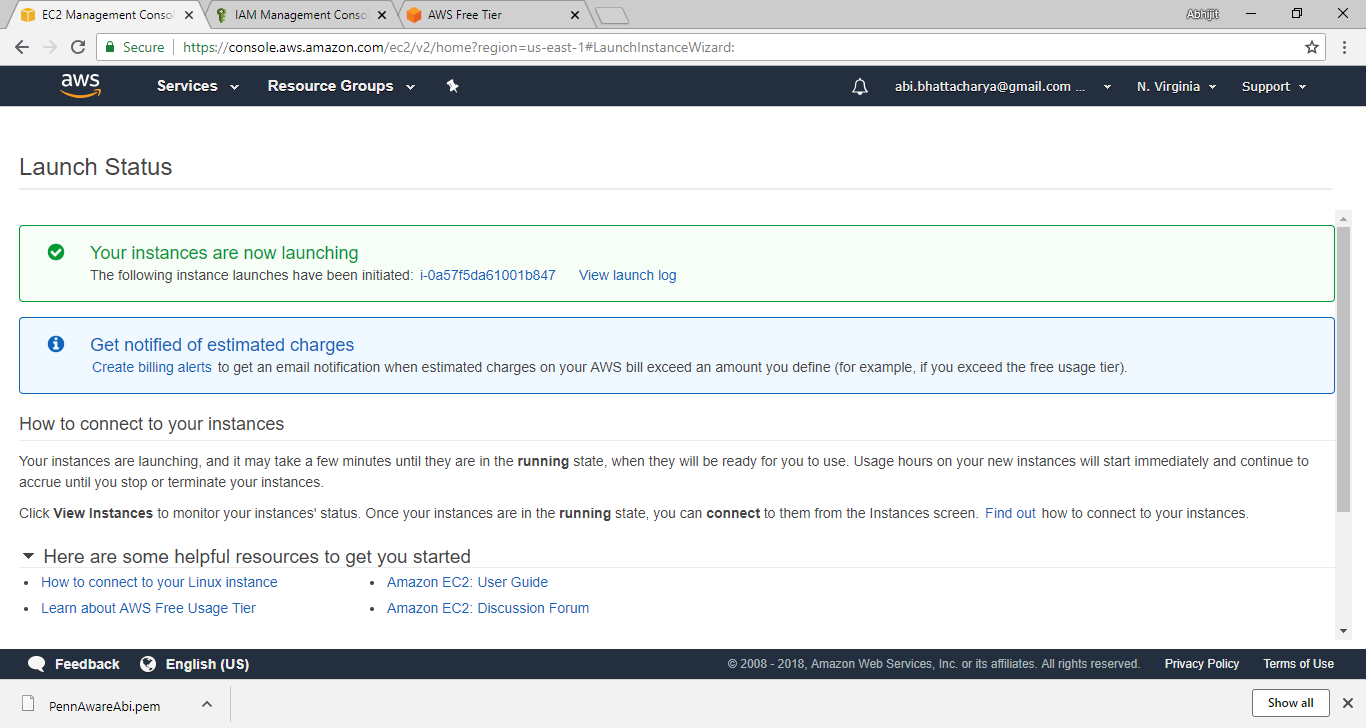






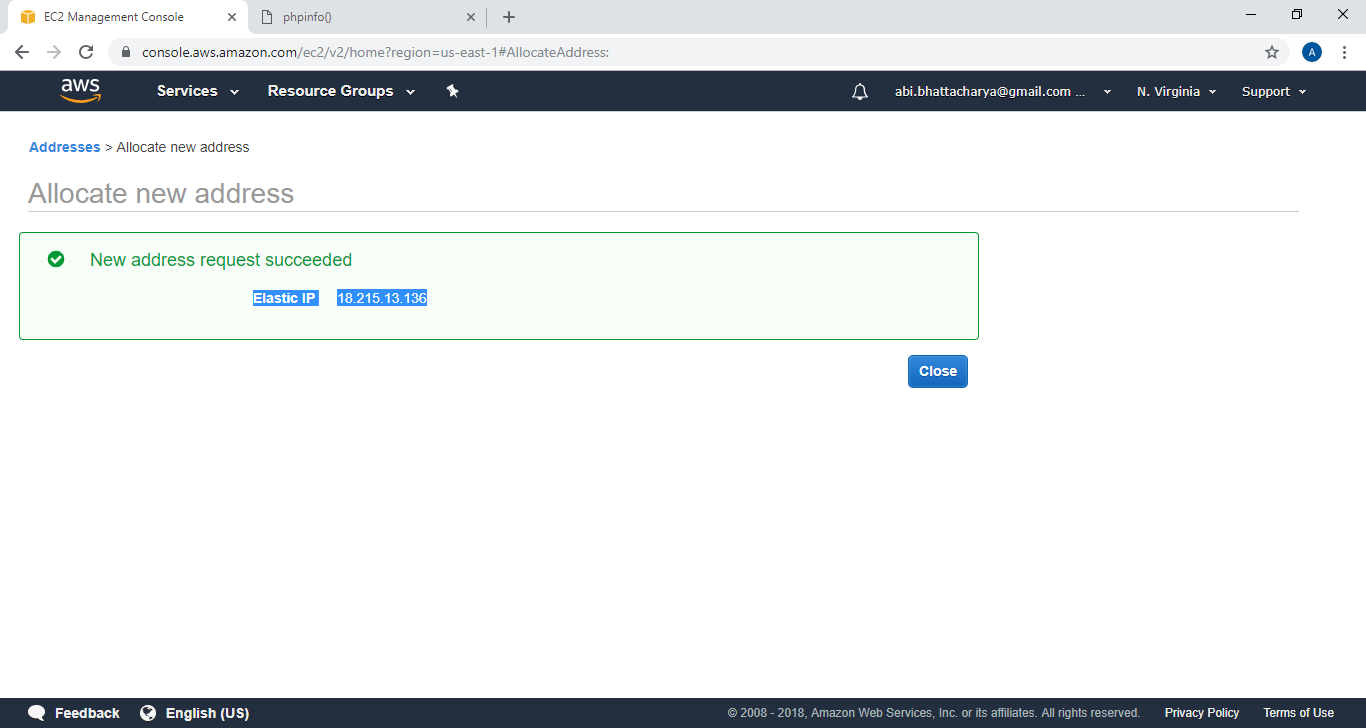
Name and download the key-pair



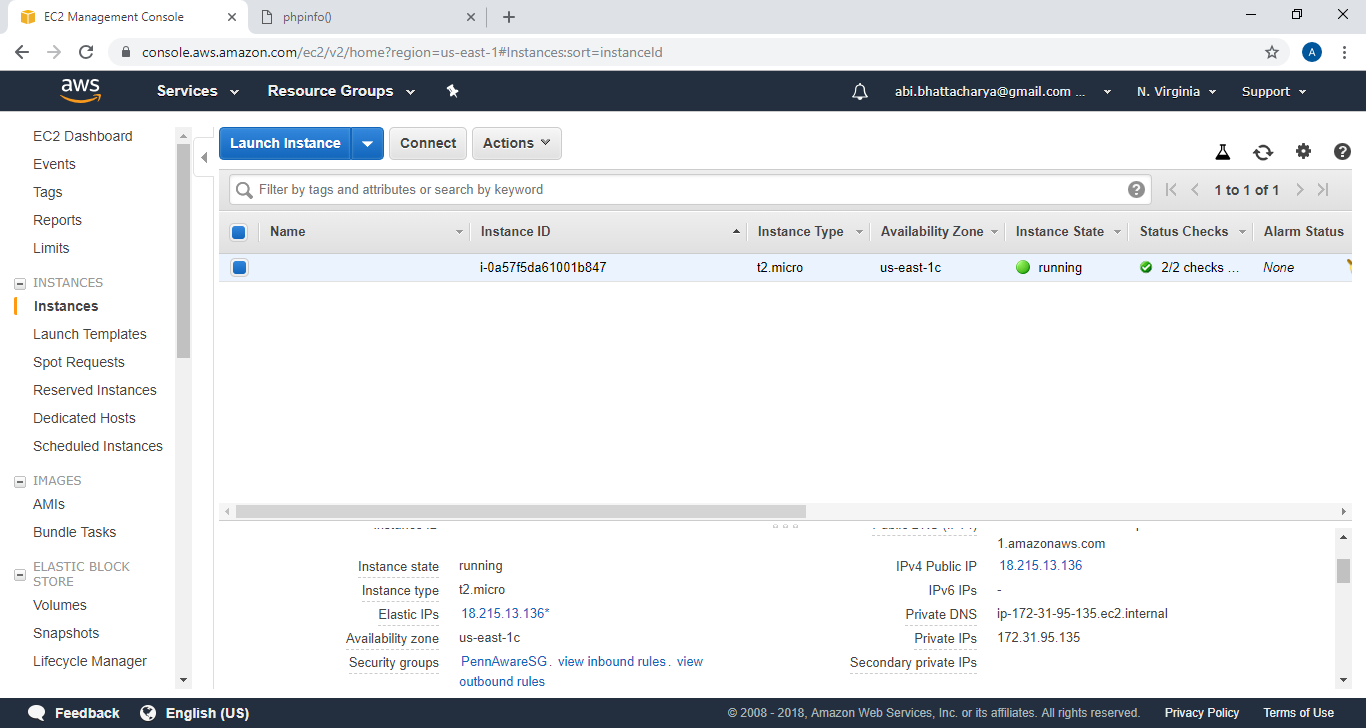


## CREATE AND ASSIGN AN ELASTIC IP ADDRESS

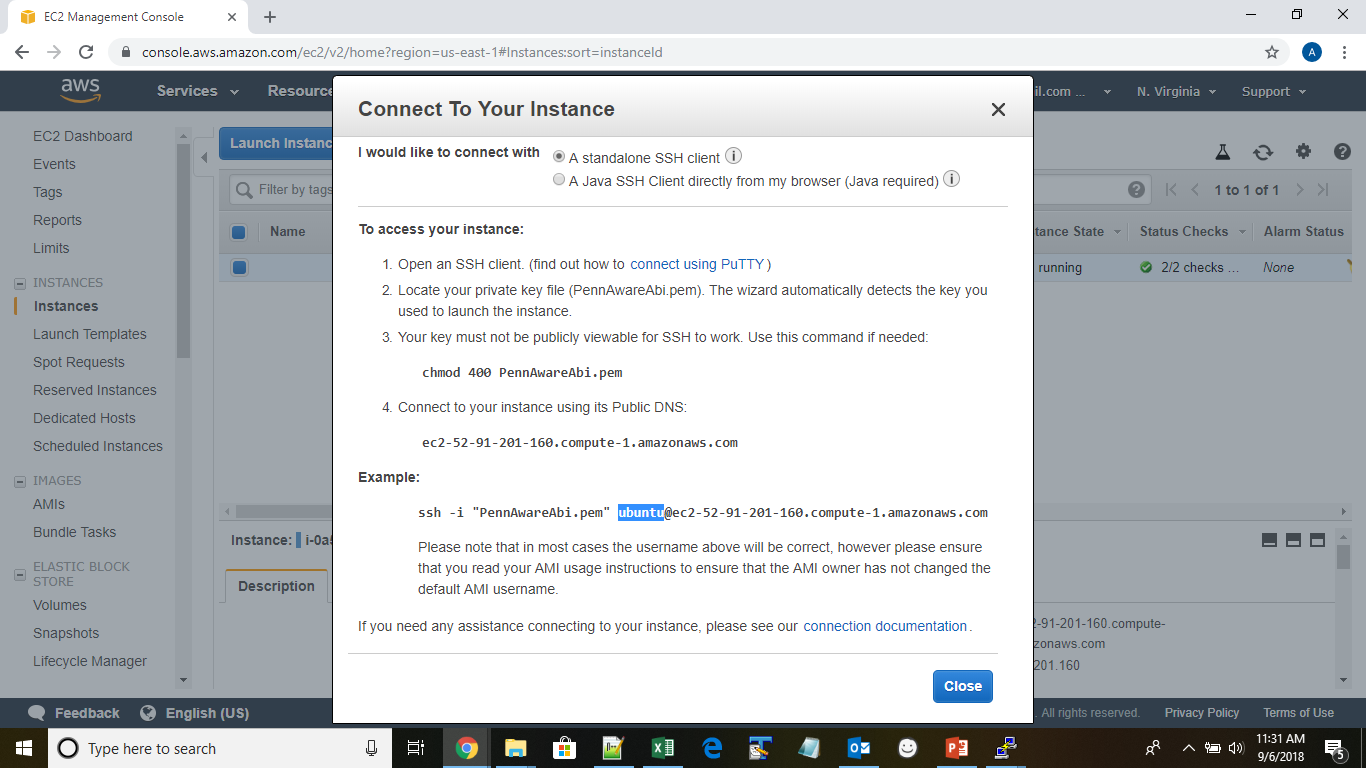
Create elastic IP address from the EC2 console and assign the IP address to the running instance

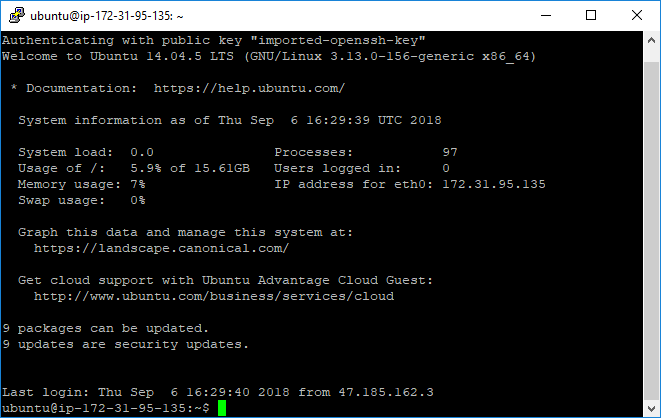


After assigning the IP address, get the public IP address from the instance details in the EC2 dashboard



Putty/connect via SSH to the server at the public IP address using the key pair to access the server via ssh. Hit the connect button next to the launch instance button to get the connection details to the instance.





The public IP address set up in this case was 18.215.13.136.

## GET A NAME ASSIGNED TO THE IP ON THE DNS SERVER

An A name of aware-cloud was added to the wwbp.org domain and was pointed to the static elastic IP address of the server - 18.215.13.136

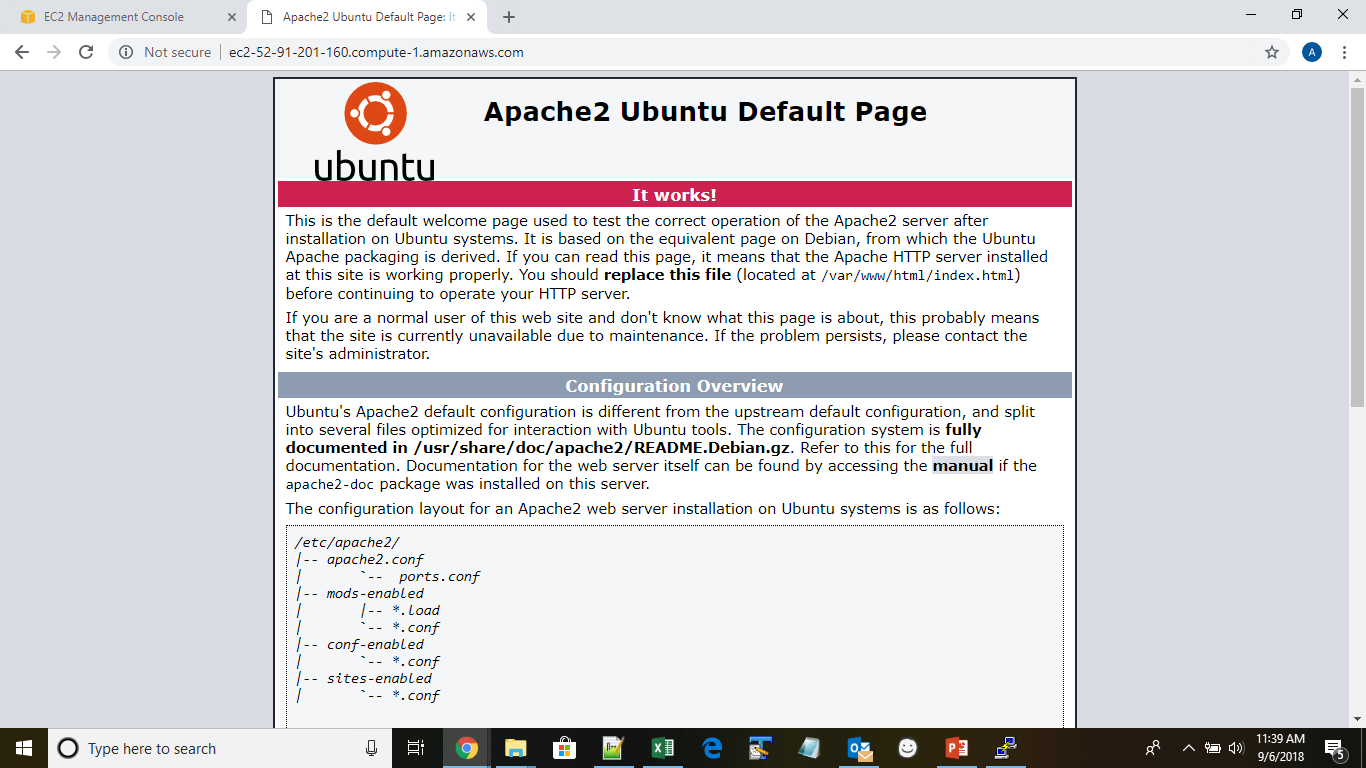
# **INSTALL REMAINING ELEMENT OF THE LAMP STACK**

In addition to the Linux server, MySQL, apache and PHP modules were also installed to prepare for the aware dashboard server install. The following site was referenced for this section: <https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-14-04>

## INSTALL APACHE

Once, connected to the server via ssh, execute the following commands on the shell prompt to install apache:

* sudo apt-get update
* sudo apt-get install apache2
* Check for proper local installation by:
  + “curl <http://localhost>” (you should get back html text)
* If you don’t know the external DNS ip (from connect screen of the EC2 dashboard), you can use:
  + “curl ipecho.net/plain ; echo”
* Check external access from a different machine browser and use
  + “http://<external DNS ip address>” (you should get back the below screen)



## INSTALL MARIADB AND SETUP THE DATABASE

Run the following commands on the ssh terminal to install MySQL.

* “sudo apt-get update”
* “sudo apt-get dist-upgrade”
* “sudo add-apt-repository ppa:ondrej/php”
* “sudo apt-get update”
* “sudo apt-get install mariadb-server php5.6-mysql”
* “sudo mysql\_secure\_installation -u root”

The following answers were given to questions on the secure Installation script:

Enter current password for root (enter for none): <hit return>

Switch to unix\_socket authentication [Y/n] n

Change the root password? [Y/n] n

Remove anonymous users? [Y/n] Y

Disallow root login remotely? [Y/n] Y

Remove test database and access to it? [Y/n] Y

Reload privilege tables now? [Y/n] Y

Thanks for using MariaDB!

All done!

Now we need to set up a privilege account using password. You don’t want to change the root password due to maintenance issues.

* sudo mariadb
* MariaDB [(none)]> GRANT ALL ON \*.\* TO 'admin'@'localhost' IDENTIFIED BY 'password' WITH GRANT OPTION;
* MariaDB [(none)]> FLUSH PRIVILEGES;
* MariaDB [(none)]> SET PASSWORD for ‘admin’@’localhost’ = PASSWORD(‘<new password>’);
* MariaDB [(none)]> exit

Bye

After installation test the installation using the following command at the ssh terminal

* sudo systemctl status mariadb

You should see something like:

● mariadb.service - MariaDB 10.6.18 database server

Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)

Active: active (running) since Mon 2024-10-07 15:52:25 UTC; 26min ago

Docs: man:mariadbd(8)

https://mariadb.com/kb/en/library/systemd/

<data continues>…

If mariaDB isn’t running, you can try:

* sudo systemctl start mariadb

Check the db connection with

* sudo mysqladmin version

You should see output like:

mysqladmin Ver 10.0 Distrib 10.6.18-MariaDB, for debian-linux-gnu on x86\_64

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Server version 10.6.18-MariaDB-0ubuntu0.22.04.1

Enter Password: <root password from above>

## INSTALL PHP

Install PHP by running the following commands at the ssh terminal

* “sudo apt-get install php5.6 libapache2-mod-php5.6 php5.6-mcrypt”

## CREATE APACHE MOD LINKS

The default apache mod list fails to include some of the links we need for https to work

* “cd /etc/apache2/mods-enabled”
* “sudo ln -s ../mods-available/proxy.conf proxy.conf”
* “sudo ln -s ../mods-available/proxy.load proxy.load”
* “sudo ln -s ../mods-available/proxy\_http.load proxy\_http.load”
* “sudo ln -s ../mods-available/proxy\_msrpc.load proxy\_msrpc.load”
* “sudo ln -s ../mods-available/socache\_shmcb.load socache\_shmcb.load”
* “sudo ln -s ../mods-available/ssl.conf ssl.conf”
* “sudo ln -s ../mods-available/ssl.load ssl.load”

# **SETTING UP THE AWARE DASHBOARD**

## PULLING DOWN THE LATEST VERSION OF THE AWARE SERVER

The following commands were executed at the ssh terminal to pull down the latest code of the aware server

* “cd /var/www/html”
* “sudo git clone https://github.com/denzilferreira/aware-server.git”
* “cd /var/www/html/aware-server”
* “sudo git pull”
* “cd ..”
* “sudo mkdir tmp”
* “cd tmp”
* “sudo git clone <https://github.com/TTRUCurtis/aware-server.git>”
* “cd ..”
* “sudo rcp -rp tmp/aware-server/aware-server-overlay/application/\* aware-server/application”
* “cd ~”

## CERTBOT AND SSL CERT INSTALL

The following commands were run at the ssh terminal:

* “sudo apt-get update”
* “sudo apt-get install software-properties-common”
* “sudo add-apt-repository ppa:certbot/certbot”
  + The PPA had been DEPRECIATED.
  + Press [ENTER] to continue [...] <ENTER>
* “sudo apt-get update”
* “sudo apt-get install python3-certbot-apache”
* “sudo certbot --apache”

Please enter the domain name(s) you would like on your certificate (comma and/or space separated) (Enter ‘c’ to cancel): <Enter machine name>

The last command automatically configures Apache and assigns the certificates to your host.

The script says that it did the following:

Successfully received certificate.

Certificate is saved at: /etc/letsencrypt/live/ttru-aware.wwbp.org/fullchain.pem

Key is saved at: /etc/letsencrypt/live/ttru-aware.wwbp.org/privkey.pem

This certificate expires on 2025-01-06.

These files will be updated when the certificate renews.

Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate

Successfully deployed certificate for ttru-aware.wwbp.org to /etc/apache2/sites-available/000-default-le-ssl.conf

Congratulations! You have successfully enabled HTTPS on <https://ttru-aware.wwbp.org>

## SECURING THE AWARE DASHBOARD

The apache configuration file for the dashboard virtual host must be edited to enable the aware dashboard. Per the lets encrypt setup the conf file updated for ssl is /etc/apache2/sites-available/000-default-le-ssl.conf

Edit that file and make the following changes:

* Change the document root from

DocumentRoot /var/www/html

To

DocumentRoot /var/www/html/aware-server

* Change log files from

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

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

To

ErrorLog /var/log/apache2/aware\_error.log

CustomLog /var/log/apache2/aware\_access.log combined

* locate the line

ServerName <machine\_name>

And add the following below it:

<Directory "/var/www/html/aware-server">

Allow from all

Options +Indexes

</Directory>

* After the line:

SSLCertificateKeyFile /etc/letsencrypt/live/<machine\_name>/privkey.pem

Add the following:

SSLCertificateChainFile /etc/letsencrypt/live/<machine\_name>/chain.pem

ErrorLog /var/log/apache2/ssl\_error\_log

LogLevel debug

TransferLog /var/log/apache2/ssl\_access\_log

Please note that these instructions deviate from the aware server install instructions. The cypher suite provided in the aware server instructions is significantly shorter than the one included by Letsencrypt in its include file that actually has a lot of the settings

NOTE: These instructions are in case you are creating a new set of .pem files. If you are being given a set of ssl key files, replace the following lines:

SSLCertificateFile /etc/letsencrypt/live/aware.wwbp.org/fullchain.pem

SSLCertificateKeyFile /etc/letsencrypt/live/aware.wwbp.org/privkey.pem

SSLCertificateChainFile /etc/letsencrypt/live/aware.wwbp.org/chain.pem

With whatever 3 files you have been given. (ex.)

SSLCertificateFile /etc/apache2/sslkey/wwbp\_org\_cert\_2020.cer

SSLCertificateKeyFile /etc/apache2/sslkey/wwbp\_org\_cert\_2020.key

SSLCACertificateFile /etc/apache2/sslkey/\_.wwbp.org\_ca.crt

Please also note that these changes are done for the \*:443 virtual host (https)

Edit file /etc/apache2/sites-available/000-default.conf and make the following changes for the \*:80 virtual host that hosts the http server needed to lookup public certs

* Change the document root from

DocumentRoot /var/www/html

To

DocumentRoot /var/www/html/public

* Add the following lines before </VirtualHost> tag

ServerPath "/public/"

ServerName <machine\_name>

## SET UP PUBLIC CERTIFICATES IN THE PUBLIC FOLDER FOR THE HTTP VIRTUAL HOST

* “sudo mkdir /var/www/html/public”
* “sudo chmod 777 /var/www/html/public”
* “sudo openssl x509 -outform der -in /etc/letsencrypt/live/<machine\_name>/cert.pem -out /var/www/html/public/server.crt”
* “cp /var/www/html/public/server.crt /var/www/html/public/ca.crt”
* “sudo chmod -R 755 /var/www/html/public”

## MAKE ADJUSTMENTS TO PHP.INI

In the ssh terminal run the following commands:

* “sudo nano /etc/php/5.6/apache2/php.ini”

Make the following changes

* Find the line

upload\_max\_filesize = 2M

and replace it with

upload\_max\_filesize = 200M

* look for the word extension and in the sections where extensions are described add the following

extension=mcrypt.so

# **MYSQL CONFIGURATION**

This section works very similar to the documentation provided by aware.

## COPY LETSENCRYPT CERTIFICATES AND ALLOW ACCESS TO THEM BY MYSQL

Run the following commands on the ssh terminal:

* “cd /etc/mysql”

If you haven’t been given ssl files to use:

* “sudo cp /etc/letsencrypt/live/ttru-aware.wwbp.org/cert.pem /etc/mysql”
* “sudo cp /etc/letsencrypt/live/ttru-aware.wwbp.org/chain.pem /etc/mysql”
* “sudo cp /etc/letsencrypt/live/ttru-aware.wwbp.org/fullchain.pem /etc/mysql”
* “sudo cp /etc/letsencrypt/live/ttru-aware.wwbp.org/privkey.pem /etc/mysql”
* “sudo chown mysql:mysql /etc/mysql/\*.pem”

If you have been given files to use: (modfiy for your file locations:

* “sudo cp /etc/apache/sslkey/\_.wwbp.org\_ca.crt /etc/mysql”
* “sudo cp /etc/apache/sslkey/wwbp\_org\_cert\_2020.cer /etc/mysql”
* “sudo cp /etc/apache/sslkey/wwpb\_org\_cert\_2020.key /etc/mysql”

Then do the following:

* “sudo chmod 777 /etc/mysql/mariadb.conf.d/\*”
* “sudo chmod 777 /etc/mysql/”

Uncomment and edit the following lines in the [mysqld] section and then add them to the [mariadb] section (probably end of file):

* “sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf”

ssl-ca=/etc/mysql/chain.pem (or whatever your ca file is)

ssl-cert=/etc/mysql/cert.pem (or whatever your chain file is)

ssl-key=/etc/mysql/privkey.pem (or whatever your key file is)

We need to update some of the config files for utf8mb4 character sets. (Can get files from [douglasvbellew@aware-dev.wwbp.org](mailto:douglasvbellew@aware-dev.wwbp.org):/home/douglasvbellew/mysql)

* Update the following files in /etc/mysql:
  + startup.sql

SET GLOBAL block\_encryption\_mode = 'aes-256-cbc';

SET GLOBAL character\_set\_client = utf8mb4;

SET GLOBAL character\_set\_connection = utf8mb4;

SET GLOBAL character\_set\_database = utf8mb4;

SET GLOBAL character\_set\_results = utf8mb4;

SET GLOBAL character\_set\_server = utf8mb4;

SET GLOBAL character\_set\_system = utf8mb4;

SET GLOBAL collation\_connection = utf8mb4\_unicode\_ci;

SET GLOBAL collation\_database = utf8mb4\_unicode\_ci;

SET GLOBAL collation\_server = utf8mb4\_unicode\_ci;

* + mariadb.conf.d/50-client.cnf

[client]

default-character-set = utf8mb4

* + mariadb.conf.d/50-server.cnf

[mysqld]

\*Under the “Basic Settings” add:

init\_file = '/etc/mysql/startup.sql'

\*Under the “Character sets” section add line 1 and edit line 2

character-set-client-handshake = FALSE

collation-server = utf8mb4\_unicode\_ci

* “sudo chmod 755 /etc/mysql/mariadb.conf.d/\*”
* “sudo chmod 755 /etc/mysql/”

Restart the MySQL server by running:

* “sudo service mysql restart”

Check the status by running

* “sudo service mysql status”

## CREATE A MYSQL DATABASE AND DB USER FOR THE AWARE DASHBOARD

Connect to the database as root and create the database by running the following commands at the ssh terminal:

* “sudo mariadb”

At the sql prompt that comes up execute the following commands

* “CREATE DATABASE aware\_dashboard;”
* “CREATE USER 'dbuser'@'localhost' IDENTIFIED BY 'dbuser';”
* “GRANT ALL PRIVILEGES ON \*.\* TO 'dbuser'@'localhost' WITH GRANT OPTION;”
* “FLUSH PRIVILEGES;”
* “exit;”

## LOAD AWARE DASHBOARD CORE DATABASE

Connect to the database as the new "dbuser" and load the core database by running the following commands at the ssh terminal:

* “mariadb -u dbuser --password=dbuser aware\_dashboard < /var/www/html/aware-server/aware\_dashboard.sql”

## LOAD AWARE DASHBOARD FIXES AND PLUGIN DATA

We want to add additional content above the default aware\_dashboard (so that we can access plugins, etc. Also, we want to change the database so that our tables are utf8mb4 so that we can collect emojis and such things. (Location and availability of files may differ)

“mysql -u dbuser -p”

Enter password: <input dbuser password>

* “use aware\_dashboard;”
* “source /var/www/html/tmp/aware-server/SQL/set\_global\_character\_sets.sql;”
* “source /var/www/html/tmp/aware-server/SQL/sensor\_plugin\_data\_table\_inserts.sql;”
* “source /var/www/html/tmp/aware-server/SQL/SMS\_Plugin\_Setup\_Inserts.sql;”
* “exit;”

## SET MYSQL CONFIGURATION ON YOUR AWARE DASHBOARD

Edit the database.php file in /var/www/html/aware-server/application/config/database.php ensure the values of the variables shown below are set as they are below (NOTE: IF YOU RAN THE OVERLAY PROCEDURE ABOVE, THESE SHOULD ALREADY BE SET):

$db['aware\_dashboard']['hostname'] = 'localhost';

$db['aware\_dashboard']['port'] = '3306';

$db['aware\_dashboard']['username'] = 'dbuser';

$db['aware\_dashboard']['password'] = 'dbuser';

$db['aware\_dashboard']['database'] = 'aware\_dashboard';

We also need to make a change to this file to allow for utf characters above 1 byte (for emojis and such)

$db['aware\_dashboard']['char\_set'] = 'utf8mb4';

$db['aware\_dashboard']['dbcollat'] = 'utf8mb4\_unicode\_ci';

# **SET UP MOSQUITTO MQTT SERVER**

This section removed as unneeded.

# **INSTALL THE ANDROID SDK**

Install the android command line tools to parse uploaded plugin information by running the following in the ssh terminal:

* “cd /usr/local/src”
* “sudo wget http://dl.google.com/android/android-sdk\_r24.4.1-linux.tgz”
* “sudo tar -zxvf android-sdk\_r24.4.1-linux.tgz”
* “sudo apt-get install openjdk-8-jdk openjdk-11-jdk”

Add Android SDK to your user's bash profile by running the following:

* “sudo nano ~/.bashrc”

Add the following content to .bashrc

export ANDROID\_HOME=~/android-sdk-linux

export PATH=$PATH:$ANDROID\_HOME/tools:$ANDROID\_HOME/platform-tools

Save the file and run the following to complete install:

* “source ~/.bashrc”
* “cd android-sdk-linux/tools”
* “sudo ./android update sdk --no-ui -t platform-tools”
  + Do you accept the license ‘android-sdk-license-##########’ [y/n]: y

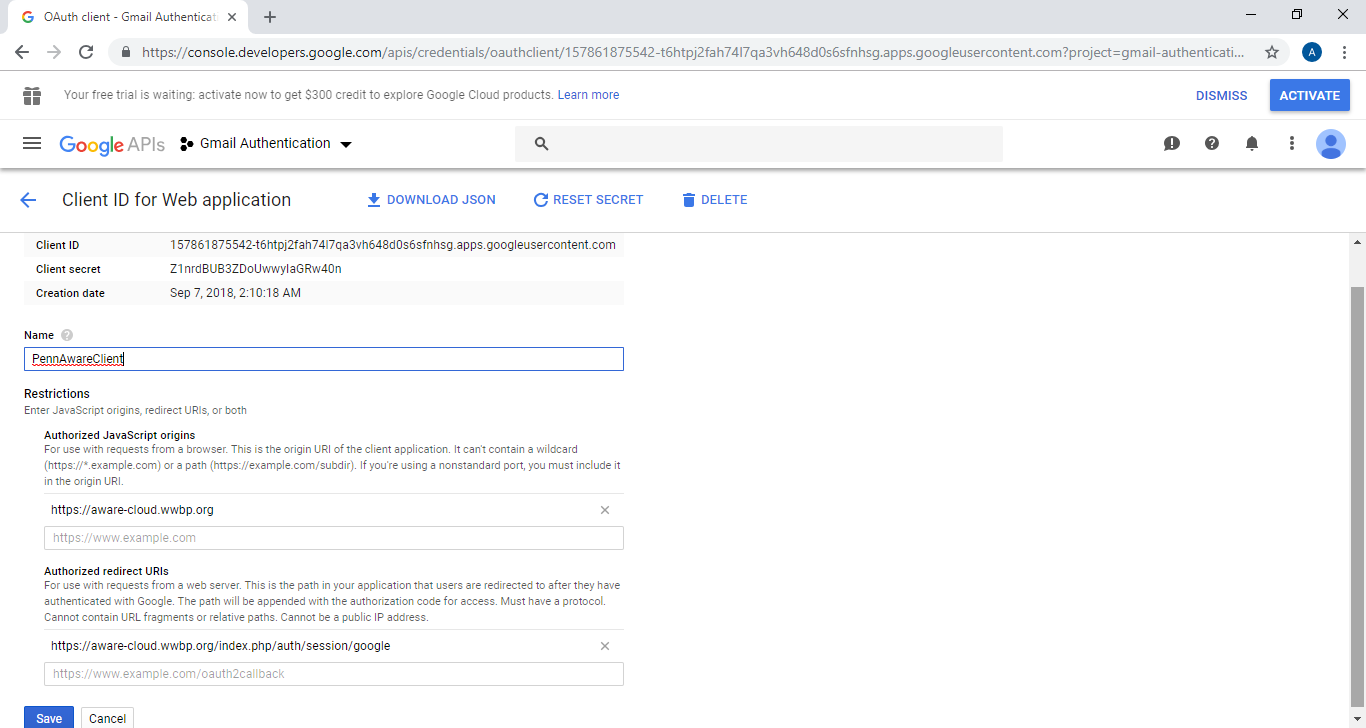
# **CONFIGURING AWARE DASHBOARD**

This section follows instructions laid out in the aware dashboard setup guide.

## ADD YOUR SERVER TO GOOGLE OAUTH CREDENTIALS

Go to the Google’s Developer Console, at https://console.developers.google.com. Create a new project and then create a new Google OAUTH credentials client with settings shown below.

-DB Notes: Sal Created the keys due to needing to know about privacy policies and other information. He associated the project with the address: [sal.giorgi@gmail.com](mailto:sal.giorgi@gmail.com)



The Client ID (e.g., 157861875542-t6htpj2fah74l7qa3vh648d0s6sfnhsg.apps.googleusercontent.com) and Client secret (e.g., Z1nrdBUB3ZDoUwwyIaGRw40n) for the final step

## FINAL AWARE DASHBOARD CONFIGURATION

Edit /var/www/html/aware-server/application/config/config.php (sudo vi or nano) and make sure the file has the following settings set (note that for a different environment some of these settings may need to change e.g. each machine must have unique domain name)

* “sudo chmod 777 /var/www/html/aware-server/application/config/config.php”

$config['encryption\_key'] = 'GX$#th@)?FGHty';

$config['cookie\_secure'] = TRUE;

$config['android\_sdk'] = '/usr/local/src/android-sdk-linux/';

$config['public\_keys'] = '/var/www/html/public/';

$config['mqtt\_hostname'] = 'aware.wwbp.org';

$config['mqtt\_port'] = '8883';

$config['oauth\_id'] = '157861875542-t6htpj2fah74l7qa3vh648d0s6sfnhsg.apps.googleusercontent.com';

$config['oauth\_secret'] = 'Z1nrdBUB3ZDoUwwyIaGRw40n';

* “sudo chmod 755 /var/www/html/aware-server/application/config/config.php”

Save the file and restart apache by running

* “sudo service apache2 restart”

Remove directory /var/www/html/tmp:

* “sudo rm -rf /var/www/html/tmp”

At this point, you will need to log into the dashboard with an account that you want to be the primary user. Once you’ve logged in, you will need to give that (newly created) user manager privileges:

“mariadb -u dbuser -p”

* “use aware\_dashboard;”
* “select \* from users;”
  + Find the “id” of the user you want to give manager privileges
* “update user\_levels SET manager=1 where user\_id="<user\_id\_number>";”
* “select \* from users;”
  + You should see a “1” under the manager column