Setting Up NGINX Web Server and Reverse Proxy in AWS

Since its introduction more than a decade ago, NGINX has become the most popular web server for busy websites. More than 160 million websites use NGINX, including [more than half of the top 100,000 websites](https://w3techs.com/technologies/cross/web_server/ranking).

With NGINX having reached such popularity, it makes sense to use NGINX for all your [web‑serving](https://www.nginx.com/resources/glossary/web-server/) needs, and additionally to take advantage of its capabilities as a [reverse proxy server](https://www.nginx.com/resources/glossary/reverse-proxy/), [caching server](https://www.nginx.com/resources/glossary/caching/), [load balancer](https://www.nginx.com/resources/glossary/load-balancing/), and more.

This blog post guides you in getting NGINX and NGINX Plus up and running on Ubuntu 16.04 on Amazon Web Services (AWS), an easy‑to‑use setup that doesn’t affect the settings on your personal computer. This blog post covers both NGINX and NGINX Plus.

Once you complete the steps given here, you may want to continue with the [NGINX Core](https://university.nginx.com/instructor-led-training/nginx-core2) training, a full‑day course where you’ll learn how to configure, administer, and manage NGINX. The course also readies you for our other instructor‑led and self‑paced [courses on advanced topics](https://university.nginx.com/).

Accessing a Terminal

For this tutorial, you need a terminal to connect to your Amazon Elastic Compute Cloud (EC2) instance. Macs have a built‑in terminal, but Windows users need to download PuTTY or an equivalent tool. If you prefer to use your own local terminal to set up NGINX or NGINX Plus and serve web content, feel free to skip to [Setting up AWS](https://www.nginx.com/blog/setting-up-nginx/#aws-setup).

Accessing a Terminal on MacOS

On Macs, you can use the built‑in terminal tool to do everything in this tutorial.

1. Click the magnifying glass icon at the right end of the top bar on the screen.
2. Type **terminal** in the pop‑up box and press Enter. A separate terminal window opens.

Accessing a Terminal on Windows

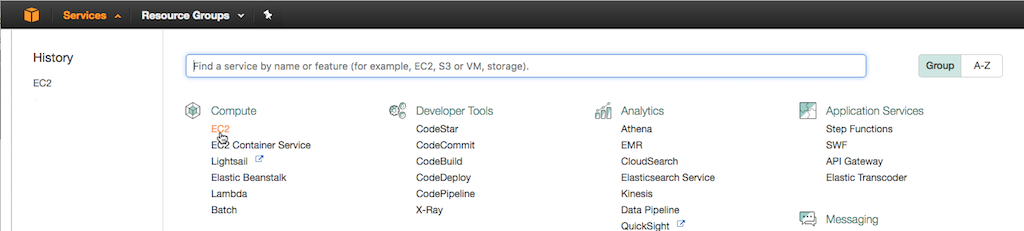
Windows doesn’t come with a built‑in terminal, so you will need to download one from the internet. PuTTY is a common ssh client used ubiquitously throughout the world.

1. Access <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>.
2. Download and install the file in a location of your choice, then open it. You can’t access the actual terminal until you finish the steps in the [next section](https://www.nginx.com/blog/setting-up-nginx/#aws-setup), however.

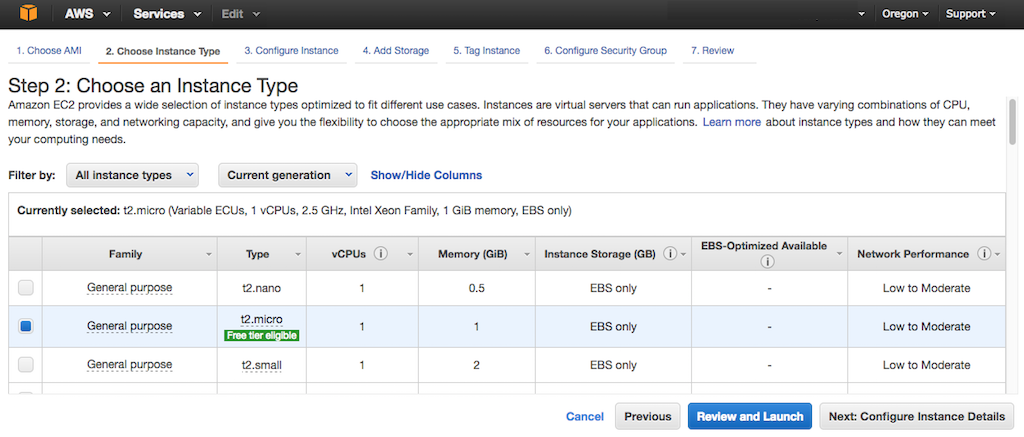
Setting Up AWS

Set up your AWS deployment by creating an EC2 instance and connecting to it.

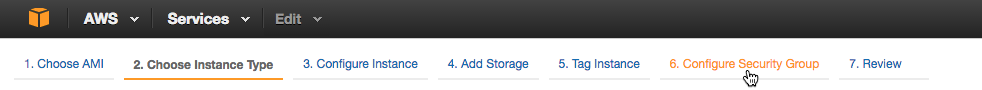
1. On the [AWS home page](https://aws.amazon.com/), sign in to the (AWS Management) Console or create a new account.
2. Click **Services** in the top Console navigation bar, then **EC2** in the **Compute** section.



1. Click the  Launch Instance  button on the page that opens.
2. On the **Step 1** page, click the  Select  button In the **Ubuntu Server 16.04 LTS (HVM), SSD Volume Type** row.
3. On the **Step 2** page, select the **t2.micro** instance type, which as of this writing is marked  Free tier eligible  and selected by default.

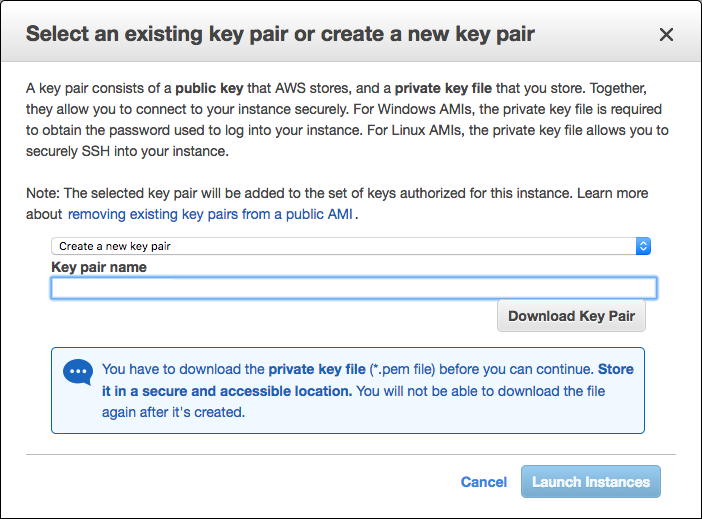


1. By default EC2 instances accept only SSH traffic. To allow incoming HTTP and HTTPS traffic:
   1. At the top of the window, click 6. Configure Security Group.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-configure-security-group.png)

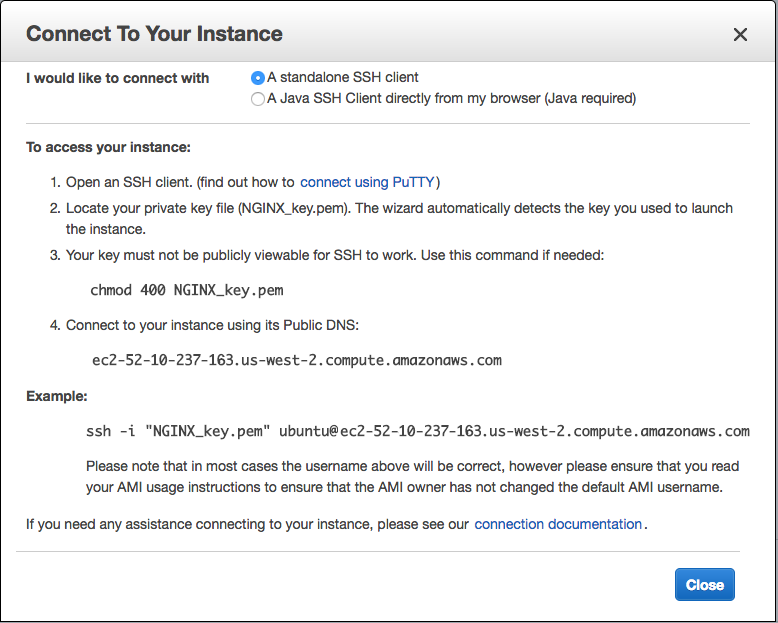
* 1. On the **Step 6** page, click the **Add Rule** button below the table, then select **HTTP** in the **Type** column. We are allowing access to our instances from any IP address (the value in the **Source** column is Anywhere).
  2. Repeat Step b for **HTTPS**.

1. Click the  Review and Launch  button at the bottom of the page.
2. On the **Step 7** page, click the  Launch  button at the bottom.
3. In the box that pops up, create a new key pair:
   1. Select **Create a new key pair** from the upper drop‑down menu.
   2. Type a name in the **Key pair name** field, such as **NGINX\_key**.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-key-pair-name.png)

* 1. Click the **Download Key Pair** button.
  2. In your file manager utility, move the downloaded **.pem** file to a secure location. For the tutorial we’re placing it in the **/Desktop/NGINX** directory.
  3. Click the  Launch Instances  button.

1. On the **Launch Status** page that opens, click the  View Instances  button at the bottom.
2. On the page that opens, the new instance appears in the table. Give it a name:
   1. Click the pencil icon in the **Name** column.
   2. Type the name in the field and click the check‑mark icon.
3. Click the **Connect** button at the top of the screen. A window like the following pops up.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-connect-instance-popup.png)

1. Follow the directions on the pop‑up to finish connecting to your instance (including the ones accessible by clicking **connect using PuTTY**, if you’re using Windows). This includes pasting the sample ssh command into your terminal once you’ve navigated to your key.

Installing NGINX Software

1. Update the NGINX software:

**sudo apt-get update -y**

**sudo apt-get upgrade -y**

**sudo apt-get disk-upgrade -y**

**sudo apt-get install awscli**

**sudo apt-get install nginx**

To open the AWS UI, go to Route 53 and add \* for ‘name’ after clicking create record then save.

In the server, go to /etc/nginx/site-available and create yourdomainname.tld add the code below. Change path and subdomain as need.

Setup the link:

LINK TO SITE ENABLED

sudo ln -s /etc/nginx/sites-available/demo.epicmath.com /etc/nginx/sites-enabled/demo.epicmath.com

// server\_name ~^(?<sub>[a-zA-Z0-9-]+)\.domain\.com$; # will cause the sub-domain to

// yourdomainname.tld

server {

listen 80 default\_server;

listen [::]:80 default\_server ipv6only=on;

root /var/www/html;

index index.html index.htm;

server\_name bwatest.com;

location / {

try\_files $uri $uri/ =404;

}

}

server {

listen 80;

listen [::]:80;

root /var/www/html/video;

index index.html index.htm index.php;

server\_name video.bwatest.com;

location / {

try\_files $uri $uri/ /index.html;

}

}

server {

listen 80;

listen [::]:80;

root /var/www/html/display;

index index.html index.htm index.php;

server\_name display.bwatest.com;

location / {

try\_files $uri $uri/ /index.html;

}

FOR NODE APPS

server {  
listen 80;  
server\_name q-apps.io www.q-apps.io;

location / {  
 proxy\_pass [http://localhost:8080;](http://localhost:8080;/)  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection ‘upgrade’;  
 proxy\_set\_header Host $host;  
 proxy\_cache\_bypass $http\_upgrade;  
 }

}

#for subdomain 1

server {  
listen 80;  
server\_name chat.q-apps.io;

location / {  
 proxy\_pass [http://localhost:9000;](http://localhost:9000;/)  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection ‘upgrade’;  
 proxy\_set\_header Host $host;  
 proxy\_cache\_bypass $http\_upgrade;  
 }  
}