How To Install and Configure Nginx on Amazon ec2 & RHEL (Redhat linux).

Nginx is a web server like apache. Not only as a web server, but it can also act as a load balancer, reverse proxy, etc. Performance-wise, Nginx is considered to be better than apache. Processing requests in Nginx is even based as opposed to the spanning new thread model in apache.

In this tutorial, I will explain how to install and configure Nginx on ec2 RHEL and ubuntu instances.

The process for installing & configuring Nginx on RHEL, Centos and Amazon Linux is the same.

Instance Setup

1. Launch an RHEL/Centos/Ubuntu instance using the management console. While launching the instance , configure the security group to **allow traffic from HTTP 80 port & HTTPS 443**.

Install Nginx on RHEL(Redhat Linux) OR (Amazon Linux)

Step 1: In RHEL you cannot download and install Nginx directly. You have to setup the epel (extra packages for enterprise Linux) repo to install Nginx. Install EPEL package repository.

sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

Step 2: Update the package list.

sudo yum update -y

Step 3: Install Nginx

sudo yum install nginx -y

Step 4: Check the version to make sure Nginx is installed.

sudo nginx -v

Step 5: Start and enable Nginx.

sudo systemctl start nginx

sudo systemctl enable nginx

Step 6: Check the status of Nginx to make sure it is running as expected.

sudo systemctl status nginx

YOU HAVE SUCCESSFULLY INSTALLED NGINX ON RHEL(Redhat Linux) OR (Amazon Linux).

NOW Lets Configure Nginx and create vhost on the server's

Setting up Multiple Domains Using Nginx Server Blocks

When you have one or more websites to be hosted on the same Nginx server, you need to use the Virtual Hosts configuration.

In this section I will show you how to create a virtual host configuration for a website.

Step 1: Create the website folder and a public folder to put the static assets inside /var/www

Here am going to give the name as example-one.com. Name the folder with your website name. It will be easy to identify if you have many websites running on the same server.

sudo mkdir /var/www/example-one.com/public_html

Step 2: Create a test index.html file if you dont have your own index file.

sudo vim /var/www/example-one.com/public_html/index.html

Copy the following contents in the index.html file and save the file

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Welcome to example-one.com</title>
  </head>
  <body>
    <h1>Welcome To example-one.com home page!</h1>
  </body>
</html>
```

Step 3: Change the ownership of the root document to the user which is managing the worker process.

For RHEL/Centos the user is nginx,

sudo chown -R nginx: /var/www/example-one.com/public_html/index.html

In most of the RHEL/Centos distributions, SElinux will be set to in enforcing mode. Execute the following commands for SELinux to allow accessing the nginx website files.

Do not use the following command directly.

sudo setsebool -P httpd_can_network_connect on
chcon -Rt httpd_sys_content_t /var/www/

Meanwhile below is the information for SELinux:

Question : How to Check whether SELinux is Enabled or Disabled

Answer :

SELinux gives that extra layer of security to the resources in the system. It provides the MAC (mandatory access control) as contrary to the DAC (Discretionary access control). Before we dive into setting the SELinux modes, let us see what are the different SELinux modes of operation and how do they work. SELinux can operate in any of the 3 modes :

1. Enforced : Actions contrary to the policy are blocked and a corresponding event is logged in the audit log.

2. Permissive : Permissive mode loads the SELinux software, but doesn't enforce the rules, only logging is performed.

3. Disabled : The SELinux is disabled entirely.

Check the SELinux status

use this command to check current status:

```
# getenforce
Permissive
```

The output will be either of the 3 options described above. For more verbose (in the case of permissive), use:

```
# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          permissive
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      28
```

To programmatically check the status as a true/false, one way could be:

```
# selinuxenabled
if [ $? -ne 0 ]
then
    echo "DISABLED"
else
    echo "ENABLED"
fi
```

This will return ENABLED or DISABLED.

Step 4: Create a Nginx configuration file with the websites name

For RHEL/Centos,

Create the config file under /etc/nginx/conf.d/

Copy the following contents in the vhost conf file and save the file

Note: Replace example-one with your domain name.

****

```
server {
    listen 80;
    listen [::]:80;

    root /var/www/example-one.com/public_html;

    index index.html;

    server_name example-one.com www.example-one.com;

    access_log /var/log/nginx/example-one.com.access.log;
    error_log /var/log/nginx/example-one.com.error.log;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

****

NOTE

In the root section kindly keep the path till the last directory only. i.e; Do not include the index.html file in the path.

Do not change index value i.e; the index.html name in 5th line.
Access logs and error logs are generated automatically in this conf settings. Incase you have created multiple vhost's then change the path to:

/var/log/nginx/vhost1/vhost1.com.access.log;
/var/log/nginx/vhost1/vhost1.com.error.log; or for second vhost

/var/log/nginx/vhost2/vhost2.com.access.log;
/var/log/nginx/vhost2/vhost2.com.access.log;

Step 5: Validate the server block configuration using the following command (Check nginx syntax).

sudo nginx -t

Step 6: Restart the nginx server.

sudo systemctl restart nginx

Congratulations!!! You have successfully created a website, now kindly do local pointing and check if the website is working properly

****

NOTE**

If you use the public DNS of the ec2 instance , the server will throw the following error when you restart the Nginx server.

nginx: [emerg] could not build the server_names_hash, you should increase server_names_hash_bucket_size: 64

Since the DNS name is pretty long , you have to add the server name bucket size to 128 to the example-one.conf file.

server_names_hash_bucket_size 128;

Now you will be able to access the website using your domain name. If you have used the test domain names, add it to your systems /etc/hosts file and try accessing the website.

To add more domains to the nginx server, follow the same steps with a different domain name and config names as explained.