Linux Server Configuration Project

About the project

A baseline installation of a Linux distribution on a virtual machine and prepare it to host web applications, to include installing updates, securing it from a number of attack vectors and installing/configuring web and database servers

IP Address: <http://3.122.205.129/>

SSH Port: 2200

**Start a new Ubuntu Linux Server instance on Amazon Lightsail**

1. Create an AWS account
2. Click **Create instance** button on the home page
3. Select **Linux/Unix** platform
4. Select **OS Only** and **Ubuntu** as blueprint
5. Select an instance plan
6. Name your instance
7. Click **Create** button

**SSH into your Server**

1. Download private key from the **SSH keys** section in the **Account** section on Amazon Lightsail. The file name should be like *LightsailDefaultPrivateKey-us-east-2.pem*
2. Create a new file named **lightsail\_key.rsa** under ~/.ssh folder on your local machine
3. Copy and paste content from downloaded private key file to **lightsail\_key.rsa**
4. Set file permission as owner only : $ chmod 600 ~/.ssh/lightsail\_key.rsa
5. SSH into the instance: $ ssh -i ~/.ssh/lightsail\_key.rsa ubuntu@18.218.99.181

**Update all currently installed packages**

1. Run sudo apt-get update to update packages
2. Run sudo apt-get upgrade to install newest versions of packages
3. Set for future updates: sudo apt-get dist-upgrade

**Change the SSH port from 22 to 2200**

1. Run $ sudo nano /etc/ssh/sshd\_config to open up the configuration file
2. Change the port number from **22** to **2200** in this file
3. Save and exit the file
4. Restart SSH: $ sudo service ssh restart

**Configure the firewall**

1. Check firewall status: $ sudo ufw status
2. Set default firewall to deny all incomings: $ sudo ufw default deny incoming
3. Set default firewall to allow all outgoings: $ sudo ufw default allow outgoing
4. Allow incoming TCP packets on port 2200 to allow SSH: $ sudo ufw allow 2200/tcp
5. Allow incoming TCP packets on port 80 to allow www: $ sudo ufw allow www
6. Allow incoming UDP packets on port 123 to allow NTP: $ sudo ufw allow 123/udp
7. Close port 22: $ sudo ufw deny 22
8. Enable firewall: $ sudo ufw enable
9. Check out current firewall status: $ sudo ufw status
10. Update the firewall configuration on Amazon Lightsail website under **Networking**. Delete default SSH port 22 and add **port 80, 123, 2200**
11. Open up a new terminal and you can now ssh in via the new port 2200: $ ssh -i ~/.ssh/lightsail\_key.rsa ubuntu@18.218.99.181 -p 2200

## Change timezone to UTC and Fix language issues

sudo timedatectl set-timezone UTC

sudo update-locale LANG=en\_US.utf8 LANGUAGE=en\_US.utf8 LC\_ALL=en\_US.utf8

3. Create a new user grader and Give him sudo access

sudo adduser grader

sudo nano /etc/sudoers.d/grader

Then add the following text grader ALL=(ALL) ALL

Setup SSH keys for grader

On local machine ssh-keygen Then choose the path for storing public and private keys

On remote machine home as user grader

sudo su - grader

mkdir .ssh

touch .ssh/authorized\_keys

sudo chmod 700 .ssh

sudo chmod 600 .ssh/authorized\_keys

nano .ssh/authorized\_keys

Then paste the contents of the public key created on the local machine

## Change the SSH port from 22 to 2200 | Enforce key-based authentication | Disable login for root user

sudo nano /etc/ssh/sshd\_config

Then change the following:

Find the Port line and edit it to 2200.

Find the PasswordAuthentication line and edit it to no.

Find the PermitRootLogin line and edit it to no.

Save the file and run sudo service ssh restart

## Install Apache2 and mod-wsgi for python3 and Git

sudo apt-get install apache2 libapache2-mod-wsgi-py3 git

Note: For Python2 replace libapache2-mod-wsgi-py3 with libapache2-mod-wsgi

## Install and configure PostgreSQL

sudo apt-get install libpq-dev python3-dev

sudo apt-get install postgresql postgresql-contrib

sudo su - postgres

psql

Then

CREATE USER catalog WITH PASSWORD 'password';

CREATE DATABASE catalog WITH OWNER catalog;

\c catalog

REVOKE ALL ON SCHEMA public FROM public;

GRANT ALL ON SCHEMA public TO catalog;

Alter role catalog with login;

\q

exit

Note: In your catalog project you should change database engine to

engine = create\_engine('postgresql://catalog:password@localhost/catalog')

## Clone the Catalog app from GitHub and Configure it

cd /var/www/

sudo mkdir catalog

sudo chown grader:grader catalog

git clone <your\_repo\_url> catalog

cd catalog

git checkout production # If you have a diffrent branch!

nano catalog.wsgi

Then add the following in catalog.wsgi file

#!/usr/bin/python3

import sys

sys.stdout = sys.stderr

# Add this if you'll create a virtual environment, So you need to activate it

# -------

activate\_this = '/var/www/catalog/env/bin/activate\_this.py'

with open(activate\_this) as file\_:

exec(file\_.read(), dict(\_\_file\_\_=activate\_this))

# -------

sys.path.insert(0,"/var/www/catalog")

from app import app as application

application.secret\_key = 'super\_secret\_key'

Optional but recommended: Setup virtual environment and Install app dependencies

sudo apt-get install python3-pip

sudo -H pip3 install virtualenv

virtualenv env

source env/bin/activate

pip3 install -r requirements.txt

If you don't have requirements.txt file, you can use

pip3 install flask packaging oauth2client redis passlib flask-httpauth

pip3 install sqlalchemy flask-sqlalchemy psycopg2 bleach requests

## Configure apache server

sudo nano /etc/apache2/sites-enabled/000-default.conf

Then add the following content:

# serve catalog app

<VirtualHost \*:80>

ServerName <IP\_Address or Domain>

ServerAlias <DNS>

ServerAdmin <Email>

DocumentRoot /var/www/catalog

WSGIDaemonProcess catalog user=grader group=grader

WSGIScriptAlias / /var/www/catalog/catalog.wsgi

<Directory /var/www/catalog>

WSGIProcessGroup catalog

WSGIApplicationGroup %{GLOBAL}

Require all granted

</Directory>

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

LogLevel warn

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

</VirtualHost>

## Reload & Restart Apache Server

sudo service apache2 reload

sudo service apache2 restart

## Resources

Amazon EC2 Linux Instances

Flask mod\_wsgi (Apache)

Apache Server Configuration Files

Deploy a Flask Application on an Ubuntu VPS

Set Up Apache Virtual Hosts on Ubuntu

mod\_wsgi documentation

Automatic Security Updates

Protect SSH with Fail2Ban

UFW with Fail2ban

Fix locale issue

Ask Ubuntu

Stack Overflow