**Owen Galvin, CSCI E-88 Homework01**

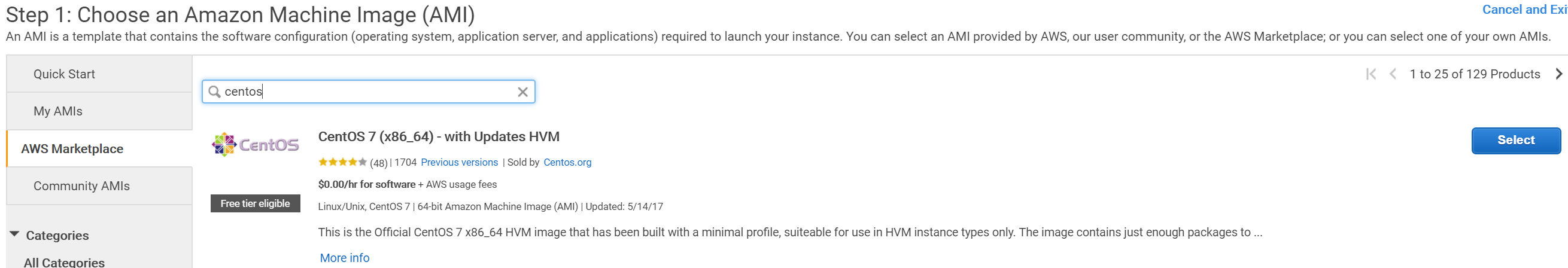
**Problem 1:**

**-- setup your AWS account if you have not done so already, create an EC2 instance , CentOS-based one (latest version is fine)**

**Solution:**

Few notes on environment, I am on Windows 10 machine, primary IDE will likely be PyCharm, with its Terminal set to use cygwin as the shell.

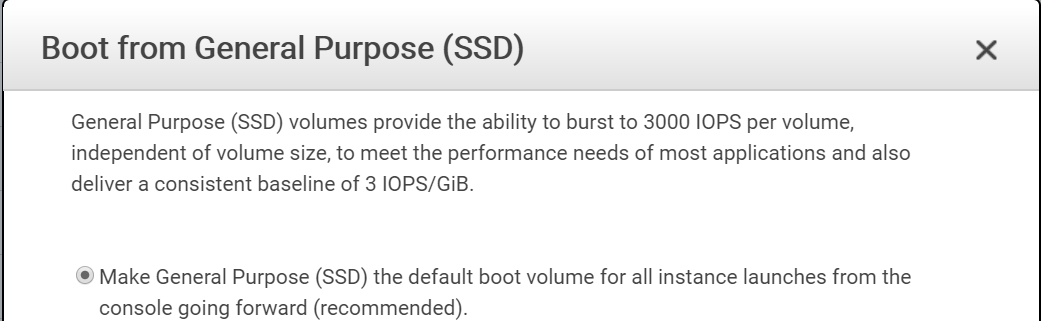
I already have an AWS account, so begin by creating the EC2 instance by going to AWS marketplace and selecting appropriate image type = CentOS 7 (x86\_64) - with Updates HVM.



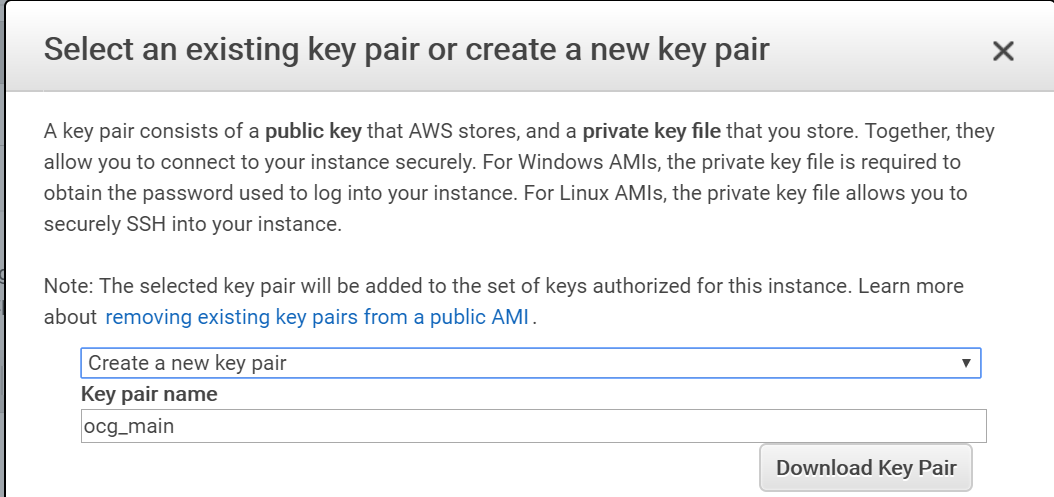
t2 micro will be good enough for this problem, which was also noted in the lab.



SSD defaults good enough also.

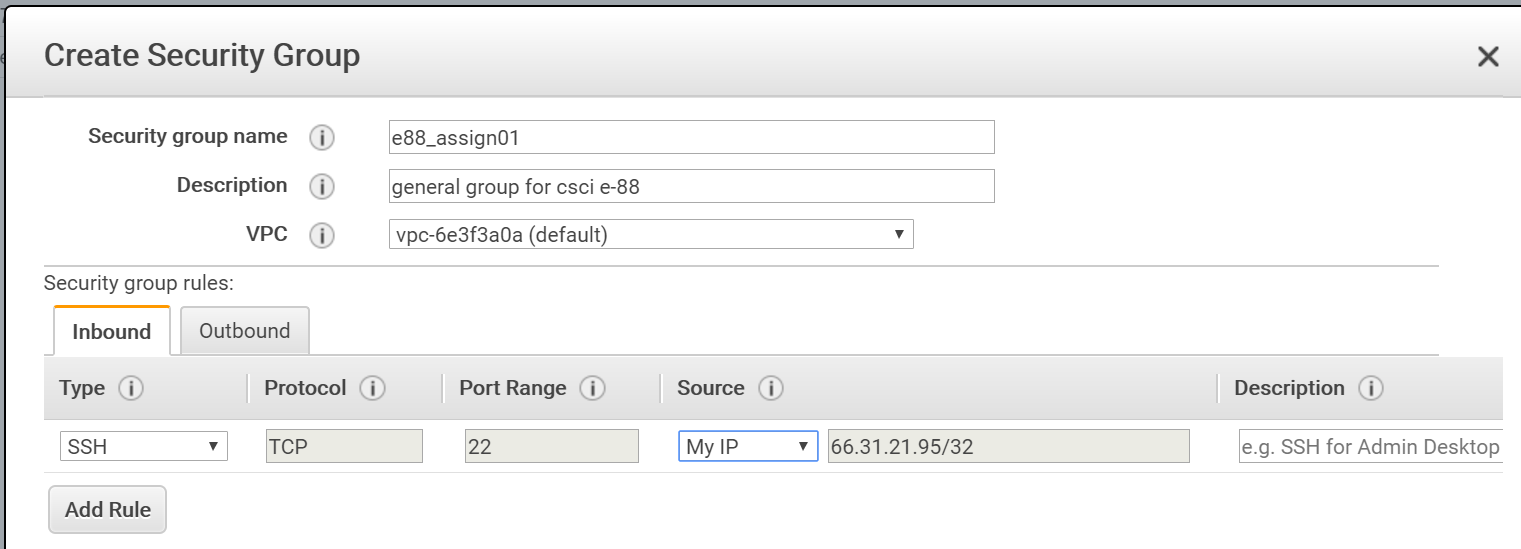


Leave all the defaults in place and Launch the instance, at prompt for key pair choose to create a new one. I have some key pairs but for personal use I’ve decided to create one default key pair for most future work.

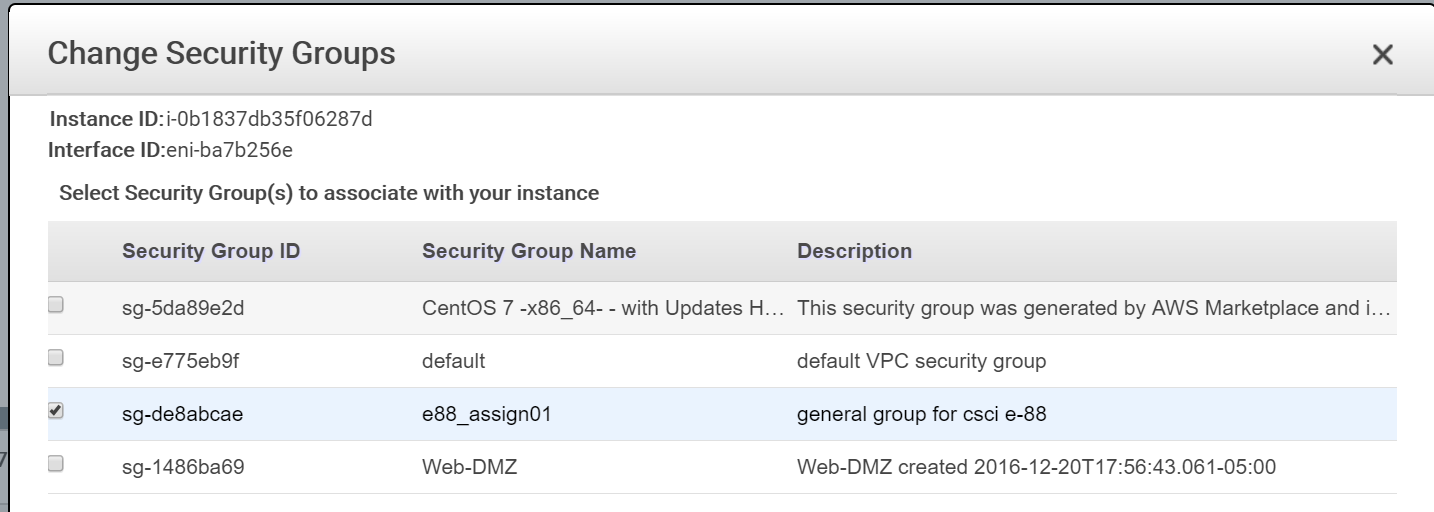


Download the key pair, move to the local directory in which I’ve stored previous .pem and launch the instance.

The above steps created a new security group with default name = “CentOS 7 -x86\_64- - with Updates HVM-1704-AutogenByAWSMP-“, think I’ll wind up deleting that one. Makes sense to create a new generic security group oriented toward this class, taking this opportunity to also add a new inbound rule to allow ssh, though only directly from my current IP address.

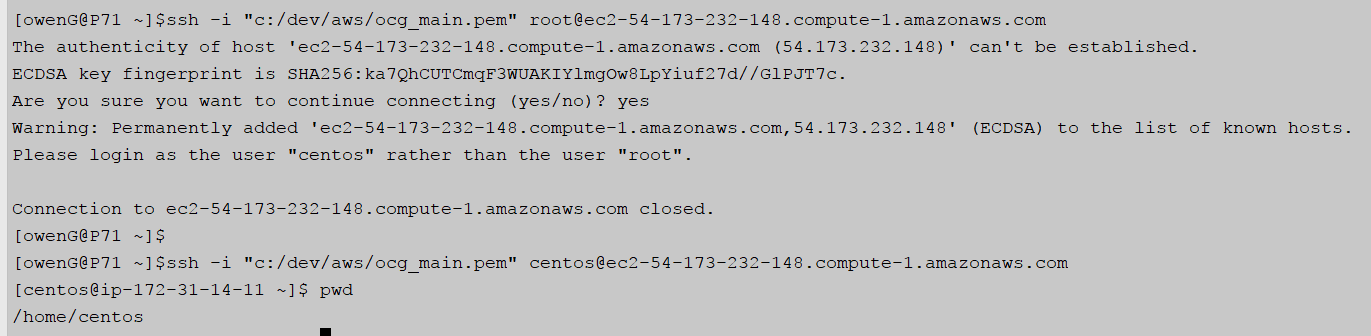


From EC2 section, choose to change security group for my new ec2, from auto-generated default to new e88\_assign01.



Confirm I can ssh into the instance, modify the example from the “Connect” dialog to path to my new .pem file (and to use **centos** as user after first failing):

ssh -i "c:/dev/aws/ocg\_main.pem" centos@ec2-54-173-232-148.compute-1.amazonaws.com



**-- implement Problem 1 from HW0 if you have not already done that**

Here is the entirety of my (python) solution from HW0, so normally I would transfer up to the ec2 via a copy and paste of the code. Should run fine in default Python 2.7 present on ec2.

|  |
| --- |
| **from** random **import** randint  nums = [] **for** i **in** range(100):  line = **' '**.join(str(randint(0, 10)) **for** j **in** range(3))  nums.append(**'{0}\n'**.format(line))  *#orig windows path: with open('c:/temp/assign00\_output.txt', 'w') as f:* **with** open(**'/tmp/assign00\_output.txt'**, **'w'**) **as** f:  f.writelines(nums) |

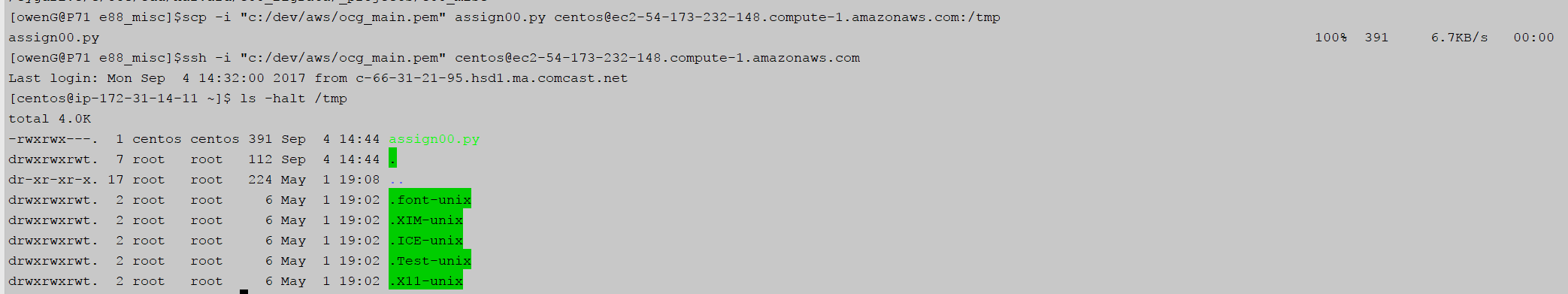
**-- deploy your Problem 1 solution on an EC2 instance or VMWare VM**

But I’ll use scp instead, below are the commands to copy, login, and confirm scp was successful (followed by a screenshot).

scp -i "c:/dev/aws/ocg\_main.pem" assign00.py [centos@ec2-54-173-232-148.compute-1.amazonaws.com:/tmp](mailto:centos@ec2-54-173-232-148.compute-1.amazonaws.com:/tmp)

ssh -i "c:/dev/aws/ocg\_main.pem" [centos@ec2-54-173-232-148.compute-1.amazonaws.com](mailto:centos@ec2-54-173-232-148.compute-1.amazonaws.com)

ls -halt /tmp



**\*\* assign00.py and the generated assign00\_output.txt will be included in homework .zip – output is > 50 lines but it remains so small I think including whole file is sensible.**

**-- run your Problem 1 and demonstrate the results**

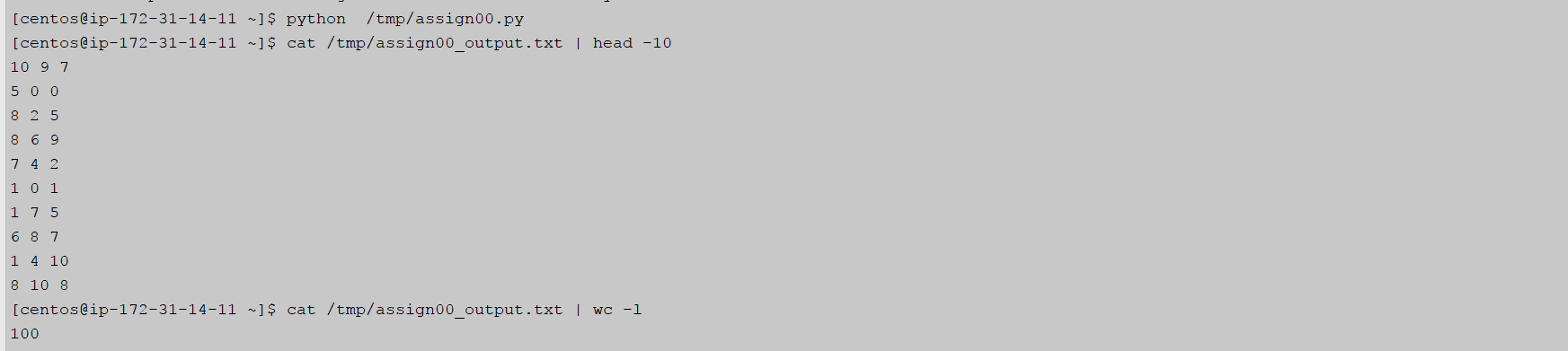
Run below commands, screenshot follows.

1. Run the program via python,
2. demonstrate it succeeded by first outputting first 10 lines of random numbers
3. followed by a line count to show 100 lines were generated.

python /tmp/assign00.py

cat /tmp/assign00\_output.txt | head -10

cat /tmp/assign00\_output.txt | wc -l



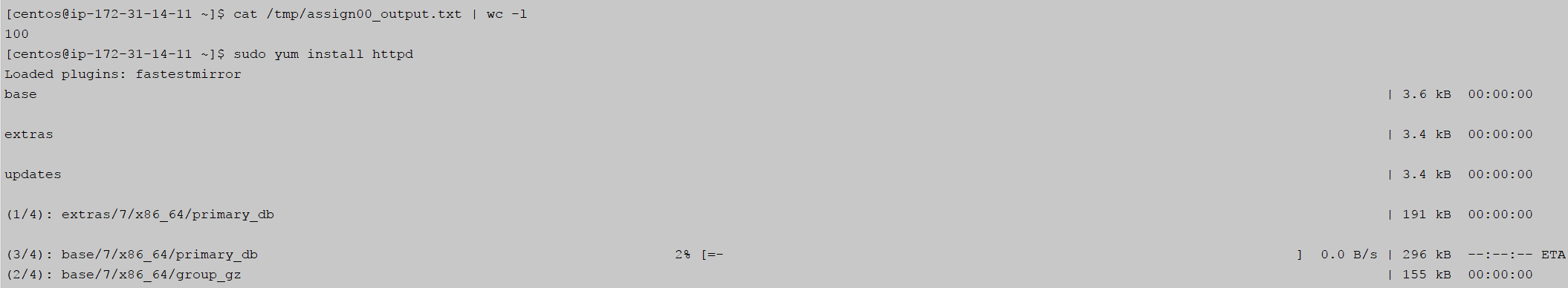
**Problem 2:**

**-- create an EC2 instance with LAMP stack**

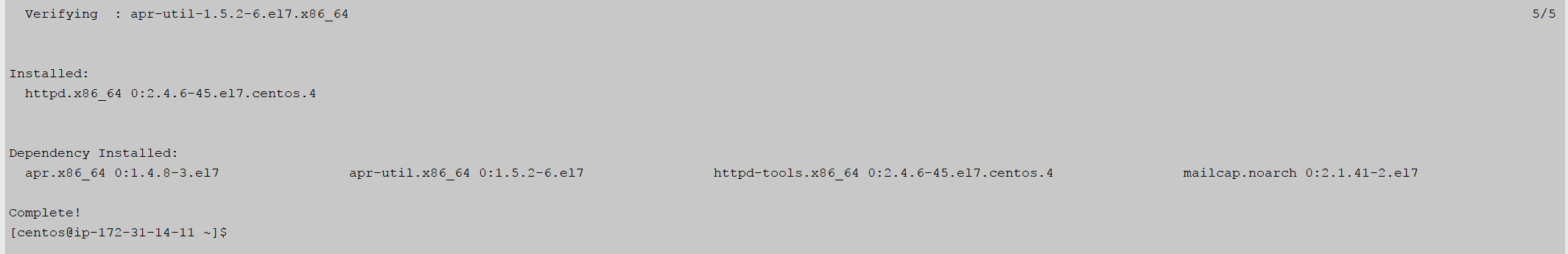
**-- demonstrate that you can run Apache web server and hit a welcome page**

Install Apache web server following instructions from lab

sudo yum install httpd



Answer “y” to a couple of prompts and

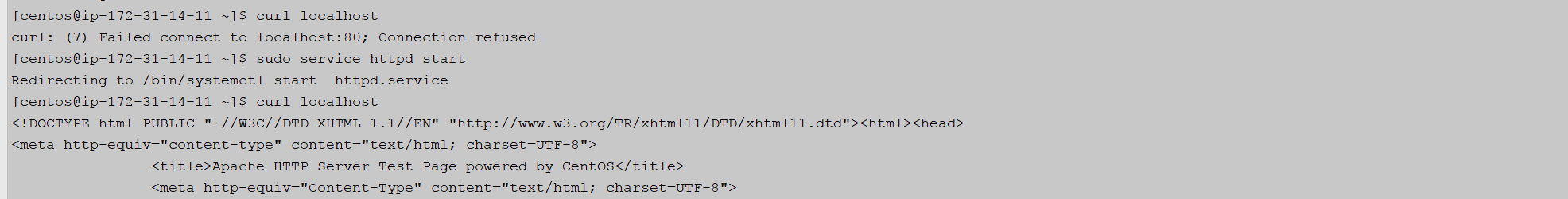


Try to curl localhost, confirm it fails and then sudo start the service and try again to demonstrate web server is now running:

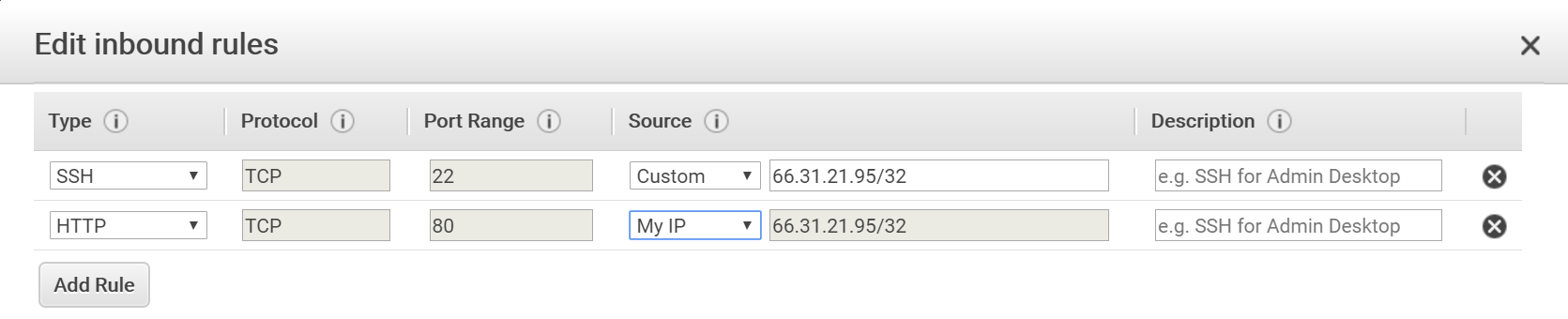
curl localhost

sudo service httpd start

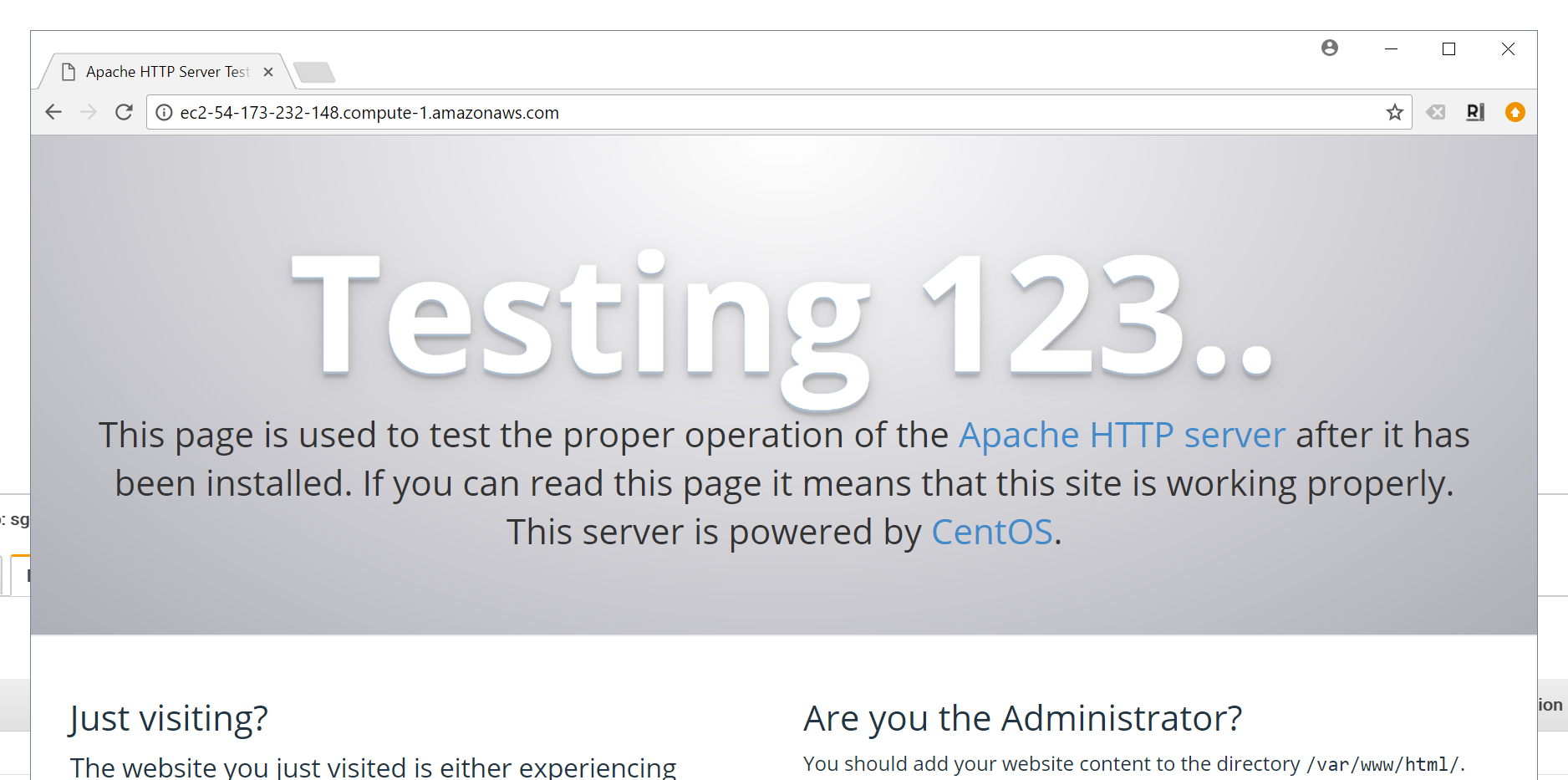
curl localhost



Go back to the security group I had created earlier and add Rule to allow inbound HTTP on port 80 so I can load the page from my laptop.



Try to access the ec2 url using the dns name used for ssh and it works:

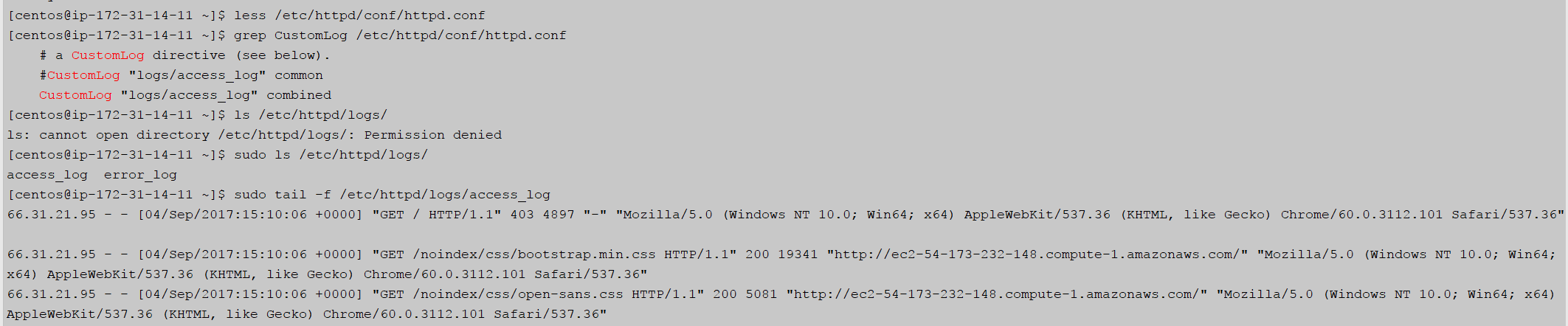


**-- demonstrate that you can see the corresponding requests logged in the log file**

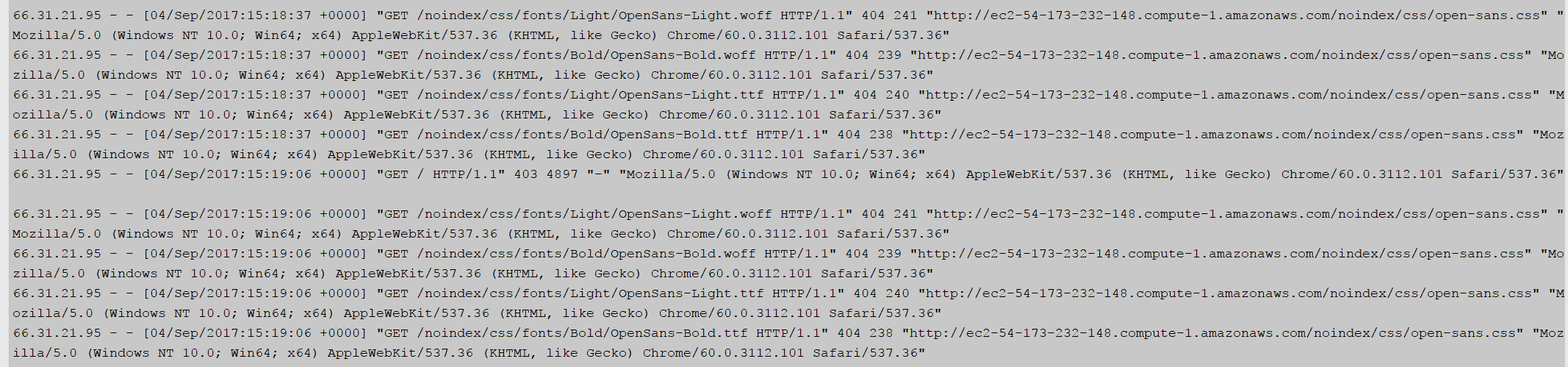
Follow instructions in lab to locate the log file and tail it, confirm nothing is being constantly written:

grep CustomLog /etc/httpd/conf/httpd.conf

sudo tail -f /etc/httpd/logs/access\_log



Hit the url a couple of times while still tailing and confirm lines are logged right after a GET is placed on the url = http://ec2-54-173-232-148.compute-1.amazonaws.com/



**-- demonstrate that you can start your DB**

First steps here are presumably to install mysql on the CentOS ec2, will attempt to follow instructions that were gone over in the session/lab. Begin by installing wget and then locating the download url via my web browser with some help from google, for “Red Hat Enterprise Linux 7 / Oracle Linux 7 (Architecture Independent), RPM Package”, do a wget on that (after installing wget).

sudo yum install wget

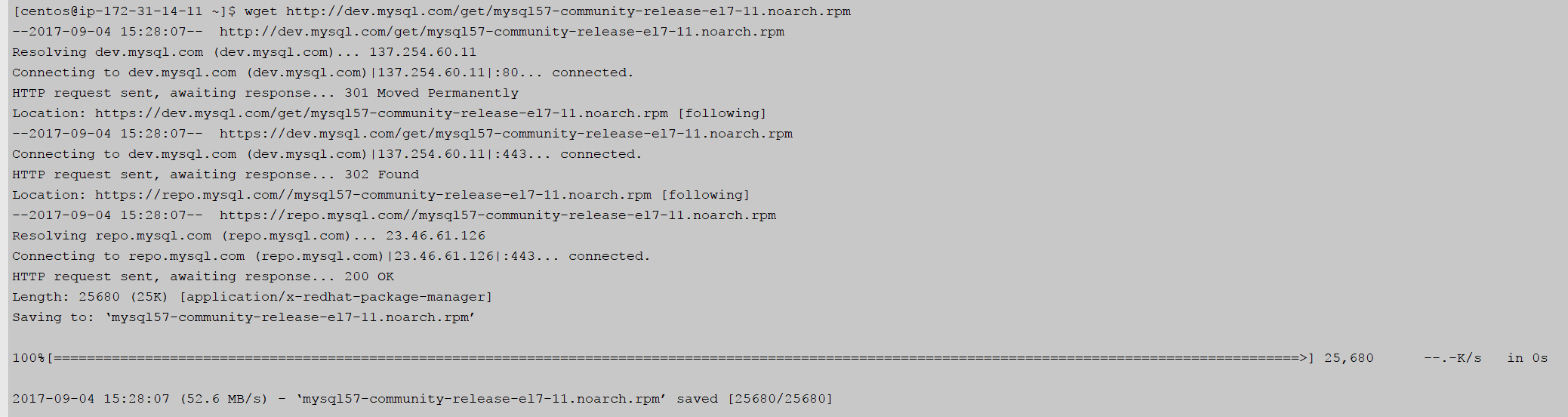
y

wget http://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm



…





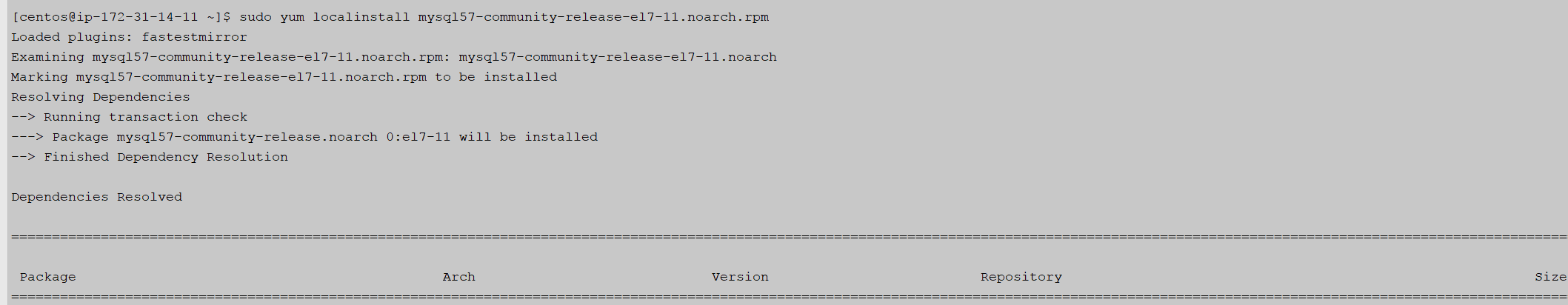
Start diverging from the lab, simply because I came across alternate instructions to install the mysql package. Final line below starts MySQL.

sudo yum localinstall mysql57-community-release-el7-11.noarch.rpm

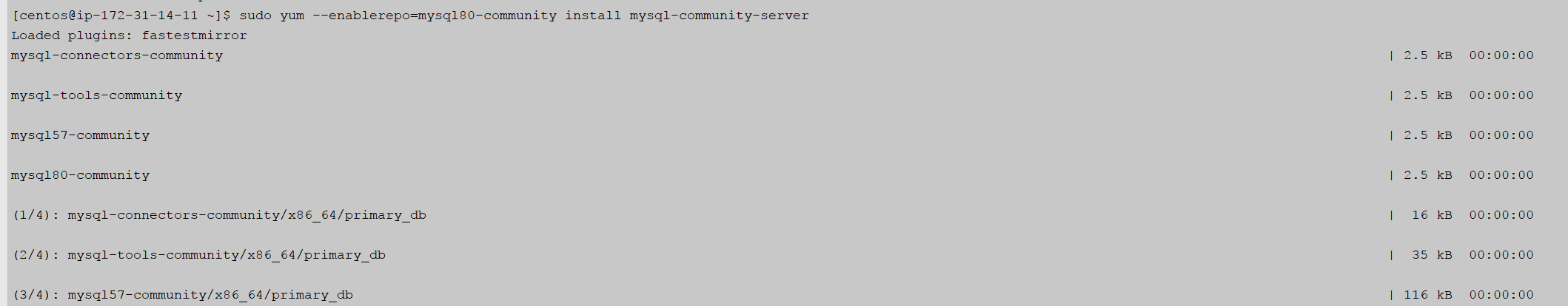
sudo yum --enablerepo=mysql80-community install mysql-community-server

y # on a couple of prompts

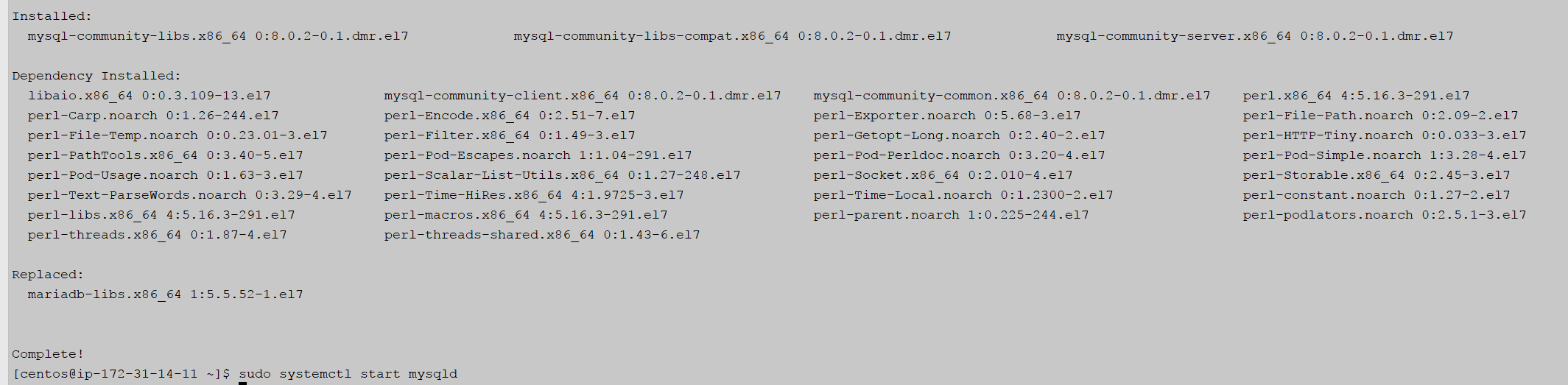
sudo systemctl start mysqld



…



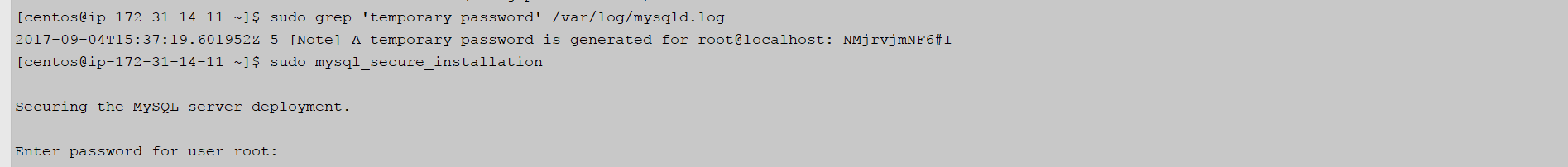
….



Follow the lab instructions to locate the temp password and run the secure installation script. Keep the test database and choose to allow remote root login, so I can easily test:

sudo grep 'temporary password' /var/log/mysqld.log

sudo mysql\_secure\_installation



Change the temp password when prompted.

**-- demonstrate that you can create a table, insert some data, query the data - using either command-line shell or GUI of your choice (PHPMyAdmin is a nice tool)**

I first tried setting up remote access so I could do stuff from the DBeaver app I have installed on my laptop and made progress but ran into roadblocks. So instead I’ll do some command line execution to fulfill assignment:

Log into mysql as root

mysql -u root -p

Then within the mysql shell, create a new test database and switch context to the new db, followed by createing a very simple table with the unfortunate name of test\_db (oops, but hey, it worked fine).

CREATE DATABASE test\_db;

USE test\_db;

CREATE TABLE test\_db(text\_value varchar(20), int\_value int);

Add three rows of data to the table and then SELECT all of them out again:

INSERT INTO test\_db VALUES('value one', 1);

INSERT INTO test\_db VALUES('value two', 2);

INSERT INTO test\_db VALUES('value three', 3);

