

Jason Loo

Comp 175: SysAdmin and Security

Installation Report

Summary of the system:

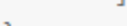
My project is based on the idea of creating an AWS EC2 instance that is used to train a deep learning model in addition to using SLURM to allow one model to be used at a time if requested by multiple users. In other words, it is similar to multiple people wanting to use a car, however it can only be driven by one person at a time, so SLURM helps us queue up these jobs, or schedule them, so that it can work efficiently.

What is nice about SLURM is that it is open source so no license is required. There is one centralized manager which is the main body of this project, and will monitor what work needs to be done and what kind of resources are available, which in this case is just one task.

Installation Process:

- 1) Ensure you have an independent AWS account, this is because we will be generating an AWS Access Key ID and Secret Access Key from under "My Security Credentials" in the menu after clicking on our username on the top right(which is unavailable in the educate account)
- 2) We will be utilizing IAM, which is the Identity and Access Management portal
 - a. Click on Users
 - b. Create a user, for instance mine is jloo
 - c. Allow for all privileges and set a password that you will remember
 - d. For permissions you want to search ECS_FullAccess then provide that under the **Attach existing policies directly**
 - e. Next add an **inline policy** here we will edit a JSON file that will let this user user ecr

```
"Statement": [  
  {  
    "Action": "ecr:*",  
    "Effect": "Allow",  
    "Resource": "*" }  
]
```

- f. 
- 3) Next we will be ready to create our EC2 Instance!
 - a. Click **Launch Instance** after navigating to the EC2 portal
 - b. Select what type of Deep learning base you want to use, in my case I went with **Deep Learning Base AMI (Ubuntu 18.04) Version 30.0**

- c. Next you want to choose a c5.large instance because normally this would be taking in a fair amount of work
- d. Then you want to keep storage the same
- e. For tags, I just have **Name** and the value associated with it is **Comp 175 Project 1 DL**
- f. You can launch this instance
- g. You will be prompted for what you would like to do regarding a keypair and you could use an existing one, however I generated a new one for simplicity with any name you like, I have it as **Comp175DLProject1**
- h. Now you can connect to the instance by right clicking on the instance and clicking **connect**
- i. Then you want to use the option for **SSH**
- j. Here you would copy and paste number 4 into your hostname for MobaXterm
- k. Your username would be **ubuntu**
- l. Then click advance and navigate to the keypair you linked to that instance.
- 4) Now you are in the terminal of the instance, but we still need to set up the user
 - a. You want to make sure you have an **Access Key ID** and **Secret Access Key** handy when you enter this next part
 - i. If you are not sure where to grab them head over to your username on the aws browser and under your username, you will see **My Security Credentials**
 - ii. Here you want to expand **Access Keys** then either create a new access key/ make one active / show key
 - b. Back on the instance terminal, type **aws configure**
 - c. Then copy and paste in the **Access Key ID** and the **Secret Access Key**
 - d. You can just leave region name and output format blank
 - e. Next you want to use the following command to actually log in, and it should succeed
 - i.

```
$(aws ecr get-login --region us-east-1 --no-include-email --registry-ids 763104351884)
```
 - f. The purpose of the last portion is to allow us to use an open source project called Docker on our Linux container
 - i. Specifically, docker is used to help deploy applications, and in our case, tensorflow
 - g. We want to utilize Tensorflow since it is another free and open source application for deeplearning, so utilize the following command
 - i.

```
docker run -it 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.13-cpu-py36-ubuntu16.04
```
- 5) Now actually training the model is easy since we would have to do a **git clone** and run it via **python** however the trickier part is actually getting slurm up and running on the server
- 6) Download <https://download.schedmd.com/slurm/slurm-20.02.5.tar.bz2> onto the server, I would recommend using the command **wget** and ensure it is in a specific directory you will remember
- 7) Since it is a .bz2 you want to use the command **bunzip** on that file
- 8) Then also since it is a .tar file, we want to use **tar -xf** on the file
- 9) Next we want to install SLURM using **./conf*** with the config script viewable on **build_linux**
 - a. After that has installed we will need to run **make -j install >m.p**
 - b. This will take some time
- 10) After that finishes installing, we will need to go into **slurm/install_linux/etc** and **gedit** or **nano** **slurm.conf** to edit the configuration file

- 11) Here you will notice at the top it will mention how to grab the Hostname and Username, in my case, it was ip-172-31-29-167 and ubuntu respectively, so place those into the control machine and anywhere you see **HOSTNAME** and **USER**
- 12) We want to start up the controller, **slurmctld** and **slurmd**
 - a. Before starting, I recommend **sudo yum install tmux** to help with multi tasking between different terminals if you are running on **linux**
 - b. If you are running ubuntu you are ok since tmux is already installed!
 - c. Just type **tmux** to enter a session
 - d. Then to exit you will hit **ctrl +b** then hit **d**
 - e. To reenter you will just type **tmux attach**
- 13) Now lets see if we can run some of these jobs in parallel
- 14) We know we made a large instance which is nice, but you could always use **./slurmd -C** to find out what kind of processors you have and how many threads per core there are
- 15) Now lets make the bash script that will be executing our jobs
 - a. We first need to make sure we already cloned a repository, for example we could do a **git clone <https://github.com/fchollet/keras.git>**
 - b. Then to start training we have a simple command **python keras/examples/mnist_cnn.py**
 - c. So in bash we can make a file with **vi DL**
 - d. Then type that command inside to execute
- 16) Make sure it is executable via **chmod 700 DL**
- 17) Then we can successfully run it via **sbatch -n4 DL**
 - a. Make sure you are in the following directory
 - i. Slurm/install/bin