Creating and deploying to an EC2 Instance

The directions to create an Amazon EC2 account (aws.amazon.com) and start virtual server instances is well documented on Amazon's AWS user guides. These notes are provided to cover nuances with the configurations for our 275 labs' runtime.

Virtual instance/machine

Gaining experience with Linux or guest operating systems, like EC2, allows you to the opportunity to work with Linux and deployment logistics. However, EC2 is optional for the course; a laptop can be successfully used for 275.

This guide supports setup of an EC2 vCPU for labs and assignments, it does not replace the EC2 documentation. Great, let's get started.

What resources will I need?

The free account provides access to EC2's lower tier configuration (CPU, memory, storage, and bandwidth). This configuration includes one thread and is a great starting point to understand how to start a vCPU, login, and deploy your software. It is advised that you should start with the free configuration (i.e. t2.micro) to learn how to manage a EC2 instance(s).

Many of 275's labs and course work will leverage multiple cores (threads) and can include stressing bandwidth, memory, and/or compute resources. If you choose to use EC2 instances, configurations of 4, 8, 16 cores (threads, vCPUs), and moderate to 5 Gbps network performance is beneficial. However, these configurations incur a fee and are not required for the class. Class labs and work can be conducted on your laptop. EC2 is offered as an experience. You may benefit in working with EC2 in 1) Gaining working knowledge of Amazon services, 2) Have not worked with Linux or managed/used guest OSs before, and 3) Have not provisioned a server.

Bottom line: Your laptop will work. Choices include: Installing Linux as the host OS, dual-boot the laptop, or use a virtual VM to host a Linux installation.

If you choose to explore configurations on EC2, you will need more cores than the single core offered by t2.micro. Please consider T2.xlarge, T3a series, M4, and M5 which represents a range for commercial deployments. These more advanced configurations can be expensive, and to re-iterate: **The use of your laptop is more than acceptable**. What do you gain? Using a laptop has its drawbacks that include it does not incorporate costs of network latency as the processes are running locally, and the contention for memory, CPU, and storage by the processes (client, server(s)).

Lastly, a fourth choice does present itself in creating a LAN by networking multiple laptops with a small (e.g., 5-port) switch. Note: This is contradictory to the recommendations for COVID-19 in not gathering in groups. Please use caution and appropriate gear and practices if you choose this option.

First, setup an account

This section refers to the following EC2 documentation:

- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/get-set-up-for-amazon-ec2.html

Steps:

1. Create your AWS (EC2) account
2. Establish a Key Pair for SSH login to your virtual server
    a. Create for OpenSSH (.pem)
3. Create a security group in AWS to allow connections
    a. Know your IP address. If your ISP uses DHCP, you will need to add a new rule each time your IP changes.
    b. Setup rules for SSH, HTTP, HTTPS, or other well-known services you are planning on allowing external connections.

Creating a virtual server

The goal of this step is to become accustomed to starting an instance, logging on, and general Linux commands for those new at Linux. This section refers to the following EC2 documentation:

- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/get-set-up-for-amazon-ec2.html
- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html#ec2-launch-instance

What you should be familiar with:

1. Create a basic/free Amazon Machine Image (AMI). You can choose from a range configurations.

    a. Login to aws.amazon.com
    b. Under Services/Compute select EC2

c. Choose Launch Instance
d. Select a 'Free tier eligible' image: Red Hat Enterprise Linux 8
e. Select t2.micro and click 'Review and Launch', Launch, and View Instances.

We are interested in the following though others are worth exploring:

a. Linux server: Red Hat Enterprise, x86 (Linux bare bones)
b. Amazon Linux AMI, x86: Provisioned with packages, and languages
c. Later in the course, a deep Learning (CUDA) – Ubuntu or Amazon based

Do you need a LAMP configuration? Most likely not, LAMPs are good for web-ish deployments and as we are not interested in httpd or storage, it is not needed. Furthermore, running these additional services will reduce the memory and cycles available to your server - diminishing performance.

Red Hat vs Amazon Linux: While the Amazon instance has pre-installed packages, these packages may not be what you need. So, using the RH instance may require more initial work, you better understand what dependencies your code requires, and can control what services are running.

2. Once the instance status is running, you can start a Terminal to login using the ssh command. You should login as ec2-user:

```
ssh -I [your .pem file] ec2-user@[public DNS name]
```

3. Explore your Linux instance, practice creating directories and files, searching, locating tools (java, etc), and shutting down the server.

4. Copying files to the vCPU using scp (secure copy) is the similar to logging in. From your laptop (not mac OSX use lowercase 'I', -i):

```
scp -v -I [your .pem file] [a file] ec2-user@[public DNS
name]:/home/ec2-user/
```

A final note on this section: Each time you create a connection to a new host, an entry in the ssh hosts file is added. If you no longer need the connection, you should periodically clean up these old entries (~/.ssh/known-hosts). The known-hosts file is a simple text file and can be edited with vim to delete entries no longer needed.

Deploying to your virtual server

Provisioning your server

Provisioning your server may require multiple secure copies from your laptop to the instance. The instance is backed up allowing you to preserve your runtime configuration. Note when downloading and installing packages, be sure to match the operating system you are running (e.g., Linux/x64).

Testing your code

Before you deploy your code to an vCPU instance, you will want to verify that the two processes (client and server) are running correctly on your laptop. This step provides for 1) faster code development and debugging, and 2) a level of confidence on how the software is working. In many software development environments, the *code→unit_test-integrate-system_test* is conducted on small private server instances to reduce disruption to others, and speed one's development.

When you are ready to deploy to your EC2 instance, use the scp command.