CS 455-555: Distributed Systems

Homework 4 (50 points)

Due Date –On Class Website–

Objectives

- Gain some experience with Amazon Web Services (AWS) for distributed applications.
- Use the time server and client code (`examples/sockets/tcp/multithreaded`) to demonstrate the use of AWS.
- Experience a few more morsels of DevOps! Or is this Brain Candy for DevOps?
- Learn the temptations of the blue pill....

Recipe

Step 0 Create and setup your Amazon Web Services (AWS) account.

Step 1 Complete the tutorials for AWS and AWS EC2.

Step 2 Create an EC2 instance using the basic Amazon Machine Image (AMI) for Linux. The image is based on CentOS/Fedora so the basic sysadmin commands will be familiar to you (they use `yum` like we have been using). Then connect to this instance using `ssh` and your security keys.

Step 3 **Public IP**. You can now run commands, like `ifconfig` to see what network address it was assigned. Note that the network address that shows inside the EC2 instance will be a private one. To see the public address, you will have to go the *AWS Management Console*. If you don't see a public address there, then you may have to create a new instance and enable the public IP. Just walk through the *Launch Instance* wizard and pay attention to the various menus.

Step 4 **Configure ports for incoming traffic**. Add a rule to the Security Groups for *Custom TCP Rule* that allows traffic to port 5005. You will have to specify allowed IP addresses as the AWS security may block traffic if you leave port 5005 open to everyone. I would recommend adding a rule to add your home IP (it can auto detect that) and also add Boise State so you can use it on campus. For Boise State, use the network address `132.178.0.0/16`, which will allow you to connect to this EC2 instance from anywhere on campus.

Note: Everyone can use port 5005 as all servers (and even ports!) are virtual so there are no conflicts. Blue pill rules!

Step 3 The Linux AMI comes with Java 1.7 pre-installed, which should be fine for this homework. However, you will need to install `java-devel` package as well as `git` as shown below.

```
sudo yum install java-devel git
```

Next clone the `CS455-resources` repo and compile the TimeServer. Now run it and you are all set and waiting for some clients!

Step 4a Run the TimeClient on your localhost and connect to the TimeServer running in AWS using its public IP address and port number 5005.

Step 4b Run a copy of the EC2 instance and run the TimeServer on it as well. This can be done simply with the menu option *Action* → *Launch More Like This*. However, you will notice that the new instance is bare and you have to reinstall `java-devel` and `git` packages, clone the repo, compile and run the time server etc.

Extra Credit Create your own custom AMI from the base EC2 instance and call it *TimeServer* (or some easily identifiable name). Launch this instance and verify that it has all your custom changes in it.

Step 5 Modify the TimeClient code so that if it fails to connect to the first server, it automatically tries the second server. You would modify the code so that it accepts a list of IP addresses for multiple servers and keeps a list internally to try. Test this functionality. For example, you can stop one of the EC2 instances to test the client.

Required Files

- Please use the separate folder named `hw4` to store your homework files.
- Please use the class name TimeServer for the server class for your program and TimeClient for the client class. Please submit the source code for client and server classes.
- Include a screenshot of your AWS EC2 management console that shows the two EC2 instances running.
- Include screenshot(s) showing your two instances of TimeServer running and the client connecting to one and then to another server.
- Include a brief `README.md` that contains your name, class number and section, homework #, and your observations on the homework.

Submitting the Homework

Homework is individual so we will use your individual github classroom repository. To obtain credit, submit your homework through classroom github as described below.

Change to the directory called hw4 in your classroom individual repo and place all required files in that folder. Then do the following steps (on your master branch):

- git add [The appropriate files]
- git commit -m "Homework hw4 complete"
- git push origin master