Intro. to Network Programming 2022 Fall

Homework 3 – Game 1A2B: Part 3

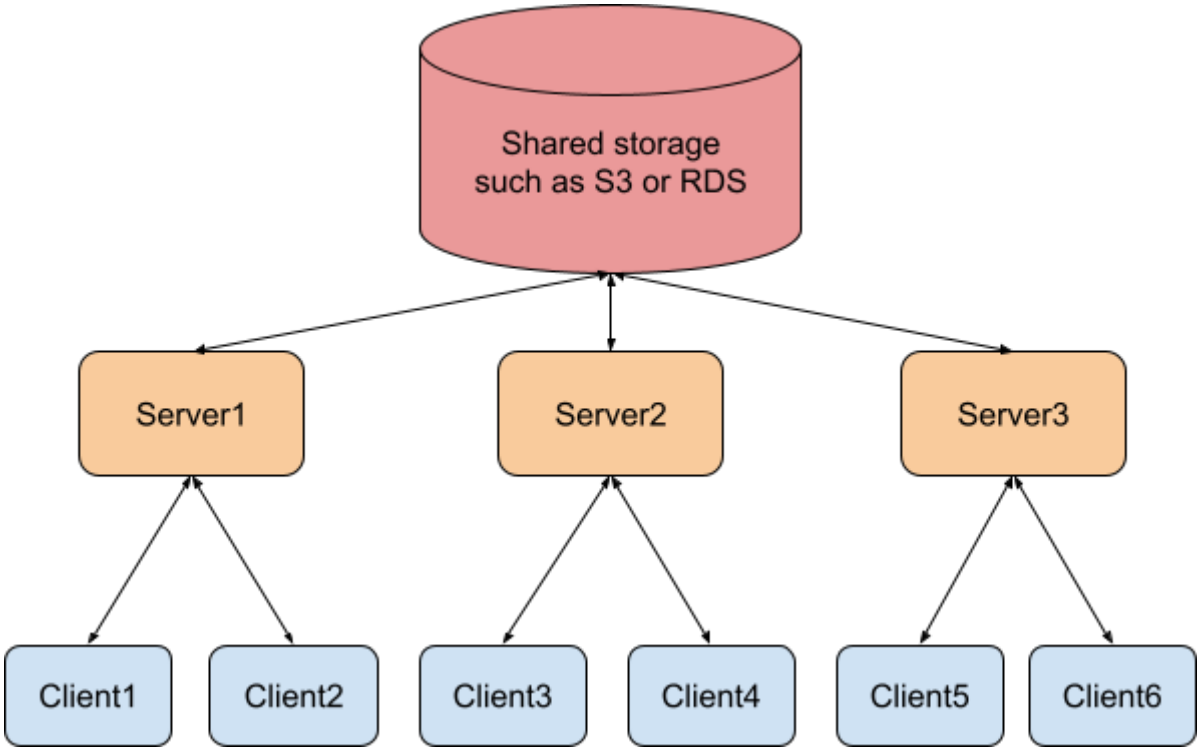
Please change the default email name into <Student ID> <Name>

Please use https://awsacademy.instructure.com/login/canvas **to login AWS Academy.**

Description

In this part, you are asked to use only **C/C++** and **AWS services** to design **multi-server for 1A2B game**. Your program should be able to handle multiple connections and receive user commands from **standard input**. After receiving the command, the server sends the corresponding message back.

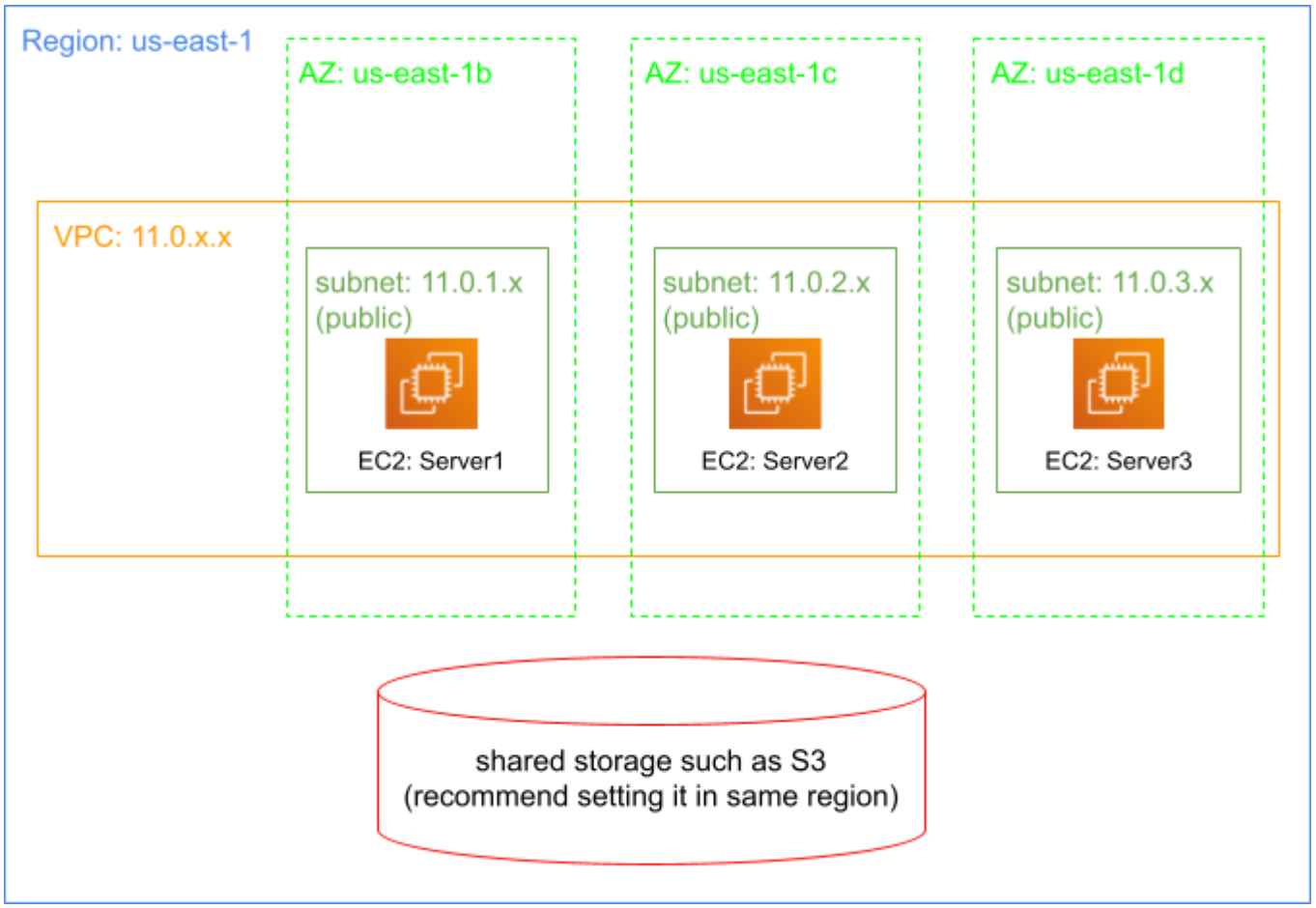
System Architecture



These servers should handle all commands in the previous part and they need to know each other numbers of login users by the shared storage. Note that your server program can get access to Amazon services using their API.

Requirements

- **AWS Environment**



The environment should be set up as above except the shared storage. It is recommended setting the shared storage in the same region. Basically, you can just use default value for other settings that we do not point out. If something that we do not notice should be configured, we will update it below.

Note: All the EC2s use the same Security Group. This Security Group should allow only SSH, TCP and UDP traffic.

- **Command**

  Your server should be able to handle at least **10** clients

  Client commands are listed in the table

Command format	Description	Result	
status	Print out all the servers' numbers of the login users. You can assume all the servers are running. You can use the same code for each server by checking their IP address or just write the different code to different EC2 to distinguish which servers you are in. Please note that you have to send this request and receive the response with **TCP.**	Example description: Server1 has 4 login users. Server2 has 5 login users. Server3 has 6 login users. This command can be used whether clients login or not.	Server1: 4 Server2: 5 Server3: 6

Grade (100%)

- **Server in AWS (80%)**
  - AWS Environment (60%)
  - Successfully Connect to each server(include ssh, tcp and udp) (10%)
  - Handle all the commands in the previous part (10%)
- **Shared storage (20%)**
  - **status** command (Here uses some commands in the previous part.)(20%)

General

- You need to use AWS Academy Learner Lab to set up your AWS cloud.
- We will not provide you any docker images, you should build your environment in the AWS EC2 by yourself and ensure your code can run in it.
- We will access your EC2 to build your code. Please predefine your environment first.
- You need to put your code in the home(~) directory.
- We will run bash build.sh to compile your code, and your server executable code pathname should be ~/build/server
- We provide "demo.py", "client.py", "testcases", "testcases_correct". Run `python3 demo.py --server-ip <server IP>` in the EC2 to ensure your previous part is correct.
- We also provide "demo_part3.py", "client_part3.py", "testcases_part3", "testcases_correct_part3". Run `python3 demo_part3.py --server-ips <server1 IP>

<server2 IP> <server3 IP>` in your computer after starting 3 servers to ensure your **status** command is correct.

Since each testcase is run independently, you should restart 3 servers manually for each testcase. (If each testcase is run dependently, it is hard to score the homework when you have a small mistake on a testcase.)

- Note that each testcase is run independently (No data at the start of each testcase) in this assignment, so you need to reinitialize the data in the shared storage when restarting 3 servers.

  Reinitializing the data in the shared storage should be implemented in your code. It will not be operated manually. Only restarting 3 servers will be operated manually.

- In EC2, your home(~) directory format should be as follows:

  Example:

  ~
  - build.sh
  - <other source code or directory>
  - …

  After `bash build.sh` example:

  ~
  - build.sh
  - build
    - server
  - <other source code or directory>
  - …

- Please choose **End Lab** in AWS Academy Learner Lab when you finish your homework. All the resources that you configured will still be available when we restart it.
- Please use **vockey** as your EC2 key pair. Do not create your own key pairs.
- Grading workflow
  1. Choose **Start Lab** and go to your AWS Management Console by AWS Academy Learner Lab and check your AWS Environment (60%)
  2. ssh 3 servers successfully and the home(~) directory format is correct (5%)
  3. ~~Remove sample testcases, upload hidden testcases,~~ and run demo.py in EC2 to check your previous part **in sample testcases** is correct. (15%)
  4. Run ~/build/server, then run demo_part3.py in our computers and manually restart 3 servers in EC2 for each testcase to check if your part3 is correct. (20%) (“restart” means terminate ~/build/server and run ~/build/server again)

Timeline

- Submission Deadline 2022/12/21 23:59
- Late Submission Deadline 2022/1/4 23:59

Submission

Please upload a zip file called "**HW3_StudentID.zip**" e.g., HW3_310551077.zip to **E3**. And the zip file should include all your source code. The format should be like *EC2 home(~) directory before `bash build.sh` format.*

- Example:

  — HW3_310551077.zip

  |--- build.sh

  |--- <other source code or directory>

  |--- …

Reference

We will list some references for you to build your AWS cloud

- AWS SDK for C++ Developer Guide - AWS SDK for C++ (amazon.com)
- AWS Documentation (amazon.com)

Ask us from Teams if you have any questions!