Intro. to Network Programming 2020 Spring

**Homework 4 - Bulletin Board System: Pub/Sub system**

**Due on** Sunday, June 14, 2020 by 23:55

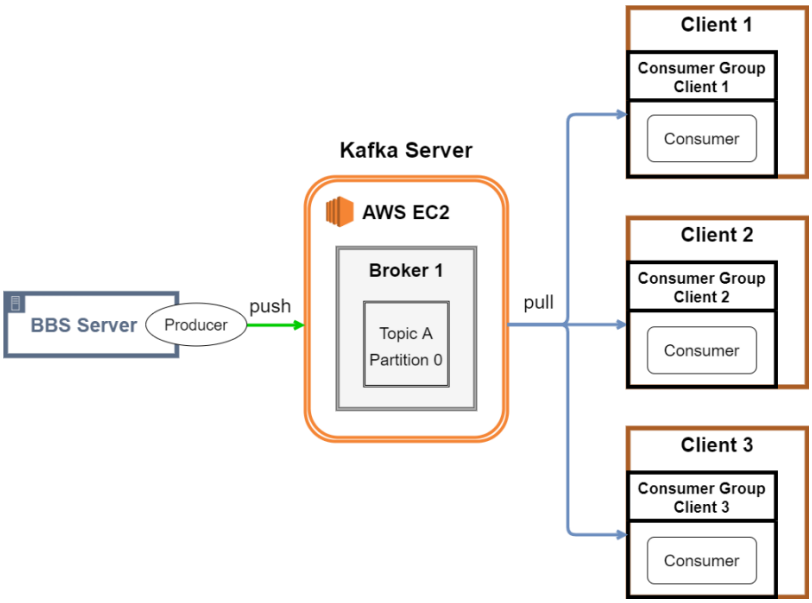
## 1. Introduction

In this part, you are going to implement the **subscription features for the BBS service.** The event will be raised whenever the client creates a new post that title contains the keyword subscribed by any other client.

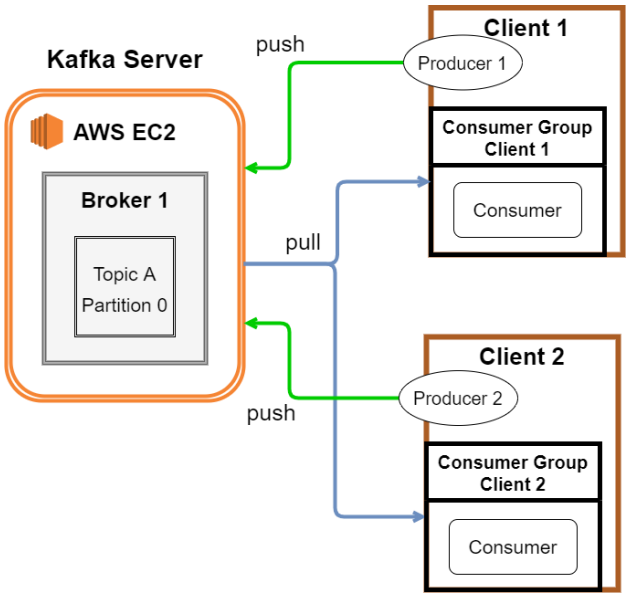
## 2. Example Architecture using Apache Kafka

The middleware server will get a message/record from the producer when there has a new post and also notify the client(s) who have subscribed to the specified topic.

A. BBS server is a producer, and each client is a consumer




B. Each client act as both a producer and a consumer



## 3. Requirements

The service can serve **at least 10 clients.** Your **server** and **client program** must be able to **handle all commands in the previous part (output results must be the same as the previous part)**. For some commands such as **whoami**, **exit**, **logout**, **create-board**, **list-board ##<key>** and **list-post <board-name> ##<key>**, your client program only sends the command to the server and gets the corresponding result from the server. However, there are new commands that your client program will **subscribe to the new post**. These commands are described as follows:

| Command format | Description | Result | |
|---|---|---|---|
| subscribe --board <board-name><br>--keyword <keyword><br>(command and arguments are in the **same line**) | Subscribe the board with a keyword, **notify** the client whenever the event be raised.<br>*Note [1]:* ***Can subscribe the same board multiple times with different keywords.*** *Notify user once someone creates a new post with a specified board and* ***the title contains one of the keywords***<br>*Note [2]: Subscribe objective (board)* ***can be nonexistent*** | Success | Subscribe successfully<br>(The notify message should at least include ***board***, ***title,*** and ***author***) |
| | | Fail (1) | Please login first |
| | | Fail (2) | [Invalid option] usage: subscribe --board <board-name> --keyword <keyword> |
| | | Fail (3) | Already subscribed |
| subscribe --author <author-name><br>--keyword <keyword><br>(command and arguments are in the **same line**) | Subscribe the author with a keyword, **notify** the client whenever the event be raised.<br>*Both notes [1] and [2] are same as the previous one (change objective to author)* ↑ | Success | *same as the previous one* |
| | | Fail (1) | *same as the previous one* |
| | | Fail (2) | [Invalid option] usage: subscribe --author <author-name> --keyword <keyword> |
| | | Fail (3) | *same as the previous one* |
| unsubscribe --board <board-name> | Unsubscribe the board from the server (or middleware server) and remove all the keywords associated with a specified board. | Success | Unsubscribe successfully |
| | | Fail (1) | Please login first |
| | | Fail (2) | You haven't subscribed <board-name> |
| unsubscribe --author <author-name> | Unsubscribe the author from the server (or middleware server) and remove all the keywords associated with a specified author. | Success | Unsubscribe successfully |
| | | Fail (1) | Please login first |
| | | Fail (2) | You haven't subscribed <author-name> |
| list-sub | List the information about the subscribed board(s) and author(s). | Success | *List all the subscribed info* |
| | | Fail (1) | Please login first |

## 4. Scenario

Run your server first, and run your client program to connect to your server. The sample outputs of the client program are listed as follows:

| Client 1 | Client 2 | Client 3 |
|---|---|---|
| bash$ ./client 127.0.0.1 7777<br>**********<br>** Welcome to the BBS server. **<br>**********<br>**% register** Paul paul@cs.nctu.edu.tw 12345<br>Register successfully. | bash$ ./client 127.0.0.1 7777<br>**********<br>** Welcome to the BBS server. **<br>**********<br>**% register** Brad brad@cs.nctu.edu.tw 12345<br>Register successfully. | bash$ ./client 127.0.0.1 7777<br>**********<br>** Welcome to the BBS server. **<br>**********<br>**% register** Gary gary@cs.nctu.edu.tw 12345<br>Register successfully. |
| **% login** Paul 12345<br>Welcome, Paul. | **% login** Brad 12345<br>Welcome, Brad. | **% login** Gary 12345<br>Welcome, Gary. |
| | **% subscribe --board** HW4_Board<br>**--keyword** Project<br>Subscribe successfully | **% subscribe --author** Jason **--keyword** hw4<br>Subscribe successfully |
| **% create-board** HW4_Board<br>Create board successfully. | | |
| | **% subscribe --board** HW4_Board<br>**--keyword** Project<br>Already subscribed<br>**% subscribe --author** Bryant<br>**--keyword** exam<br>Subscribe successfully | **% subscribe --author** Paul **--keyword** HW<br>Subscribe successfully<br>**% subscribe --author** Paul **--keyword** post<br>Subscribe successfully<br>**% unsubscribe --author** Brad<br>You haven't subscribed Brad |
| **% create-post** HW4_Board<br>**--title** About Project **--content** HW4…<br>Create post successfully. | | |
| | **% *[HW4_Board]** About Project – by Paul*<br>**% list-board** ##HW<br>Index Name Moderator<br>1 HW4_Board Paul | |
| **% create-post** HW4_Board<br>**--title** About HW and Exam **--content** blablabla<br>Create post successfully. | **% unsubscribe --author** Bryant<br>Unsubscribe successfully | |
| | **% list-sub**<br>Board: HW4_Board: Project | **% *[HW4_Board]** About HW and Exam – by Paul*<br>**% list-sub**<br>Author: Jason: hw4; Paul: HW, post |

## 5. Notes

1. **About implementation**:
   - There is **no limitation on your implementation.** You can choose whatever which framework, library, or even implement the logic by yourself, as long as it can achieve the goal of spec.
   - The output message should show the information **at least greater or equal to mentioned in the requirements**. In addition, it not restricted in any format.
2. **About Kafka environment setup:**
   - Please refer to the slide in E3 – *Apache Kafka Installation and Configuration*.
3. **About Kafka clients**:
   - C/C++
     - https://github.com/edenhill/librdkafka
   - Python
     - https://github.com/dpkp/kafka-python
     - https://github.com/confluentinc/confluent-kafka-python
   - Node.js
     - https://github.com/Blizzard/node-rdkafka
     - https://github.com/tulios/kafkajs
     - https://github.com/SOHU-Co/kafka-node

## 6. Grading Policy (100%)

- [20%] Commands from previous parts
- [30%] Subscribe part
- [30%] Unsubscribe part
- [20%] List-sub part

## 7. Submission

Please upload a zip file called "hw4_{$student_id}.zip" (e.g., hw4_0516000.zip) that includes your source code. It must include at least your **server source code** and **client source code**. The submission that doesn't follow the rule will **get 20% punishment** on the grade.

You will get **0 points** on this project for **plagiarism**.