COMP 416 – Computer Networks

Project #1

Due: October 15, 2018, 11.59pm (Late submissions will not be accepted.)

Submit your project deliverables to folder: **F:\COURSES\UGRADS\COMP416\HOMEWORK\PROJECT1**

Note: This is a group-oriented project, you are encouraged to form groups of 3 students, we suggest you a fair task distribution at the end of this project description.

Working individually is allowed but not recommended.

DriveCloud: Development of a cloud storage protocol

Introduction and Motivation:

This project work is about **application layer** of the network protocol stack. It involves application layer software development, client/server protocol, application layer protocol principles, socket programming, and multithreading. Through this project, you are not only going to develop a simple cloud storage protocol from scratch but also you are going to interact the application layer abstract programming interface (API) of Google Drive which is called Drive Client API. This API provides you linking your developed cloud storage protocol, with the Google Drive protocols of syncing data.

A major disadvantage of benefiting from the cloud storage synchronizing applications e.g., Google Drive Sync, is their inefficiency of bandwidth utilization in local area network (LAN) setups. Assume that you have one PC and one laptop at home both furnished with you Google Drive sync, connected to the same Wi-Fi router, and share a high-speed 100 Mbytes/second Ethernet. However, if you modify your Google Drive Sync folder in the PC, the changes are not directly applied to the Google Drive sync folder in the laptop. Rather, they first are uploaded to the Google Drive server and committed to your account. Then, the laptop computer's Google Drive Sync realizes the changes on its subsequent request for an update on the files state and issues a download request for the same files. Here in this scenario, a simple file modification charges your internet bandwidth twice; one initial upstream bandwidth consumption from PC and one downstream bandwidth consumption from the laptop for synchronizing itself with PC by downloading the updated files directly from the Google Drive. Likewise, your internet speed is normally slower than your Google Drive network setup, and hence synchronizing two hosts directly with your Google Drive takes longer time than synchronizing them in your LAN.

One naive solution is to provide a hybrid synchronization using your both Ethernet and internet's bandwidth. Two hosts inside a LAN can first be synchronized via the high-speed free of charge LAN connection. Then only one of them can take care of synchronization with the Google Drive cloud storage service, while the other one is resting aside. In addition to drastically increased synchronization speed between hosts, this solution saves the internet's bandwidth consumption with the gain of two.

Project Overview:

In this project, you are asked to develop a cloud data storage protocol named **DriveCloud** based on master/follower model. The master/follower model is one sort of the server/client model that was discussed in the lecture/problem session. DriveCloud master provides two types of TCP connections to interact with the DriveCloud follower: One connection for exchanging the protocol commands, and one for data transfers. Fig.1 shows connections for a sample DriveCloud master and follower. As shown in this figure, DriveCloud's master host also takes the responsibility of interacting with a Google Drive storage account using the Google Drive API.

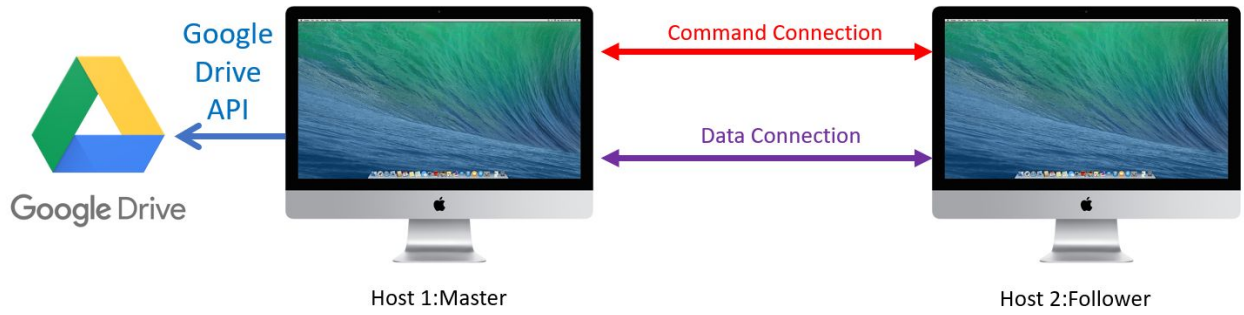

Figure 1. The DriveCloud connections.

Implementation Details:

DriveCloud design has three main components:

- Interactions with the Google Drive
- Master side of the protocol
- Follower side of the protocol

Master side overview:

DriveCloud master should:

- Establish a Google Drive API Client connection to a folder named DriveCloud in your Google Drive account. The DriveCloud folder should be placed in the Apps folder of the Google Drive to isolate the DriveCloud application from the rest of your Google Drive account. Note that the DriveCloud folder is created by the Google Drive itself and should not be created manually.
- Maintain a user-created folder named DriveCloud on the master computer. The folder should be located on the **desktop** of the user's computer. The content of the folder is expected to be synchronized with the Google Drive account. To simplify the

implementation, synchronization between the master and Google Drive account is considered **bidirectional**. Master needs to keep the track of all the changes in both its local DriveCloud folder as well as the DriveCloud folder on the cloud, and make sure that they are synchronized all the times. Once master realizes that there is any changes on its local copy or the Google Drive version of the DriveCloud folder, it should synchronize these two folders with each other.

- The master is expected to interact with Google Drive in a timely fashion (e.g., every 30 seconds), which is called the ***synchronization time window,*** upon the time it is executed, detect any changes in the status of the files in both local copy and Google Drive version of the DriveCloud folder. For this task to be accomplished, you may implement the master in a way that it keeps a list of already download/uploaded files from/to the Google Drive, as well as the list of changed files on the user's DriveCloud folder since the last interaction with the Google Drive. On synchronization time windows, the master should detect changes on DriveCloud folder at the Google Drive side using its API, compares with its local record of interactions and changes, and display a message about the sync status of its DriveCloud folder:

  "Current time: <current exact time>, no update is needed. Already synced!"

  or

  "Current time: the following files are going to be synchronized":

  -<name of the file> <going to be uploaded/download to/from the cloud><size>

  After displaying this message, upon detecting a synchronization requirement, the DriveCloud master should start and commit synchronizing the listed files with the Google Drive. Once the synchronization is done, the master needs to show the "Synchronization done with Google Drive" message. Likewise, during the upload, download time of a file, master should show the following message:

  "Downloading/Uploading <file name>"

  After the download or upload was complete for a file, master should show the following message:

  "Download/Upload completed for <file name>"

Important Note: The suggested synchronization method by keeping the track of files, is just a naive solution. You are welcome to come up with your own idea during the implementation. However, the described behavior of master and messages that should be shown are required and immutable.

DriveCloud master side requirements and protocol commands:

- In DriveCloud, master runs on the TCP server sockets.
- Uses two ports, one for the protocol commands that is exchanged between master and follower, and the other one for data (i.e., file) exchange between the master and follower.
- Upon a follower, connection establishes, as shown in PS, master should cast the server socket into a thread-based socket for that DriveCloud follower, which acts as the client. Thus, **DriveCloud master side protocol should be multithreaded to support multiple followers**. To implement this part, DriveCloud master should have a class as a thread to handle the connection between the client and server. As was described in the PS session, master (server) listens to the line by employing the accept() function on a server socket. The moment a connection has been established between a follower (client) and master, the master generates a TCP socket for that follower and starts the thread by passing the generated socket to it. In this way, multiple followers can query the server concurrently on the same port.
- Receive connection requests from followers, receive command requests, and process them.
- Have its storage folder to put files transferred.
- Provide downloading and uploading of **both ASCII Text and Binary files e.g., music**.
- Sends the hash value of each of its files to the client upon its request.

Follower side requirements:

DriveCloud follower is expected to replicate the exact behavior of master on synchronizing its local DriveCloud folder. However, in contrast to master that synchronizes itself with Google Drive Client API, follower should synchronizes itself with master. *No connection to Google Drive API is allowed for the follower.* You may do the synchronization between master and follower by either keeping the track of uploaded/downloaded files as suggested for the cloud-master interaction, or come up with you own way of implementation. You may exchange hashed value of the files to track the changes. **However, in both synchronization cases, only the updated and added files should be exchanged, and exchanging the intact files are not allowed in any case.**

The data exchanges between master and follower should be done over the data connection as shown in Figure 1. Once all the file is sent, the sender should take the hashed value of the file and send it on the command socket of the DriveCloud. On receiving the file the receiver should

also hear to the command socket and receive the hashed value of the file. Receiver should check the correctness of the file transmission by taking the hash value of each received file and compare it with the corresponding hashed value received from the sender. If there is a mismatch in the hash value of a file, the entire file should be requested again by the receiver. After transmission of the file and hashed values are done, the sender should listen to the *command socket*, until it receives one of the following messages from the receiver. Note that as the distinct feature of DriveCloud protocol, the following messages should be exactly exchanged on the command line in a case sensitive feature. You may be asked during the demo to monitor your command line for the sake of verification of correct implementation.

- "CONSISTENCY_CHECK_PASSED": This means the entire file have been transmitted successfully, and both master and follower can close their data socket.
- "RETRANSMIT": This message alarms the sender that the data received were inconsistent and corrupted, and the sender should retransmit the entire file again.

This procedure is repeated until the receiver sends "CONSISTENCY_CHECK_PASSED" message.

Follower should also print the same messages as the master on the same events described for master case. Additionally, on checking the consistency, the receiver should print the proper message associated with the consistency check similar to the same version that is sent to the sender i.e., "Consistency check for <filename> passed", or "Retransmit request for fine <filename>".

Follower side implementation details:

DriveCloud follower should:

- Have its local DriveCloud folder on the desktop.
- Check the DriveCloud master with the same synchronization window time as master. By inconsistency, we mean any addition, deletion, or change in the state of a file in either master or follower's side.
- Synchronize the changes with the master.
- Upload the added or changed files at the follower side to the master, and download the updated or added files at the master side from the master.
- All uploads and downloads should be solely done via the data connection line.
- Remove the deleted files at the follower side from the master side's DriveCloud folder, and remove the deleted files at the master side from follower side's DriveCloud folder.

- **Simplification:** For the sake of simplicity, you may assume that between two synchronization windows, a file is being changed at ONLY one place i.e., Google drive, master, or follower, but not more than one place.
- **Simplification:** Follower should only synchronize itself with the master (and not Google Drive), and master should only synchronize itself with Google Drive (and not the follower). Hence, the synchronization between master and follower is all the responsibility of the followers code.

DriveCloud system execution scenario requirements:

Requirements of a DriveCloud system execution scenarios are described below. In the project demonstration, you will be asked to show an execution scenario with the given requirements.

1. Follower and Master should both be unified in the same project. Upon the execution, your implementation should ask user to determine either of master or follower mode.
2. All the parameters e.g., address of master, Google Drive sign in credentials, etc, that concerns the user should be asked upon startup from the user. In the other words, there should not be any parameter of your program that needs to be adjusted within the code.
3. There are ***two separate computers*** connected to the same Wi-Fi router (i.e., access point).
4. One computer would act as DriveCloud master and the other would act as DriveCloud follower.
5. The DriveCloud master should also be connected to the Google Drive account of one of the group members.
6. There may exist some sample text and binary files (e.g., .txt, .mp3, .mkv, .pdf) in DriveCloud folders on Google Drive, master, and follower folders, which may not necessarily be the same version.
7. DriveCloud system should support both Text and Binary data transfers.
8. DriveCloud master should continuously (e.g., every 30 seconds) synchronizes its DriveCloud folder with the Google Drive's DriveCloud folder. As the result, the content of DriveCloud folder on the master's Desktop and DriveCloud App folder on Google Drive should be the same. The same should be happen between the master and follower DriveCloud folders at the same length synchronization windows, and as the result, the content of DriveCloud folder on the master's Desktop and DriveCloud folder on follower's Desktop should be the same.

9. You may be asked to run several concurrent followers that querying the server. Note that if you are executing more than one DriveCloud follower on a single computer, your code should automatically allocate different name DriveCloud folders on the desktop e.g., DriveCloud1 and DriveCloud2.
10. DriveCloud should support both text and binary data storage and transfers. We suggest you start with the implementation of text data operations. After that, you can implement binary data operations functionality.
11. For the sake of simplicity, you may assume that files' name are immutable, as well as there exists no sub-folder in the DriveSync folder.

Project deliverables:

You should submit your source code and project report (in a single .rar or .zip file).

- Your entire project should be implemented in Java. Implementation in network-based languages (e.g., Ruby) is not allowed. For implementation in other languages you should first consult with TA and get confirmation.
- Source Code: A .zip or .rar file that contains your implementation in a single Eclipse or Intellij IDEA. If you aim to implement your project in any IDE rather than the mentioned one, you should first consult with TA and get confirmation.

- The **report** is an ***important part of your project*** presentation, which should be submitted as both a .pdf and Word file. Your report should show the step by step Google Drive API configuration and connection with your code. As well as your master-follower synchronization. **Report acts as a proof of work for you to assert your contributions to this project.** Everyone who reads your report should be able to reproduce the parts we asked to document without hard effort and any other external resource. If you need to put the code in your report, segment it as small as possible (i.e., just the parts you need to explain) and clarify each segment before heading to the next segment. For codes, you should be taking ***screenshot*** instead of copy/pasting the direct code. Strictly avoid ***replicating the code*** as whole in the report, or leaving code ***unexplained***. You are expected to **provide detailed explanations** of the following clearly in your report:

    - How does your DriveCloud master connect to Google Drive, checks for updated files, downloads a specific file, and uploads a file? Your answer should be generalized as a step by step guide for anyone who aims to connect a Java application to Google Drive, check for updates, and do uploads and downloads.
    - How does your DriveCloud follower synchronize itself with the master? You should be explaining your proposed synchronization protocol in a clear and smooth language first, and then the code correspondence of it.

Demonstration:

You are required to demonstrate the execution of your DriveCloud protocol with the defined requirements. Your demo sessions will be announced by the TAs. Attending the demo session is required for your project to be graded. ***All group members*** are supposed to be available during the demo session. **The on time attendance of all group members to the demo session is considered as a grading criteria.**

Important Notes:

- **Please read this project document carefully BEFORE starting your design and implementation.**
- In case you use some code parts from the PS codes or the Internet, you must provide the references in your report (such as web links) and explicitly define the parts that you used.
- You should **not** share your code or ideas with other project groups. Please be aware of the KU Statement on Academic Honesty.
- For demonstration, you are supposed to bring two laptops and run your program on your own laptops (DriveCloud client laptop and DriveCloud server laptop).
- In your report, you need to dedicate a section describing the task division among the group members. There you are supposed to clearly state the contributions of each group member to the project.
- Your entire code should be commented as well as JavaDoc provided properly. You may use the following reference to get ideas about writing a clear and neat JavaDoc. Quality of your provided comments and JavaDoc is considered as part of your overall grade.
  https://www.oracle.com/technetwork/java/javase/documentation/index-137868.html

- Your report will present your designs aspects and acts as a reference for your implementations before and after the demo session. **Please take the report part seriously and write your report as accurate and complete as possible.**
- In case you are using any types of Google Drive Sync client (e.g., Google Backup and Sync), you should make sure that you made your DriveCloud folder out of the sync in such clients. In other words, there should be only one copy of your DriveCloud folder on

every computer, which should only reside on your desktop as it is, and without being enclosed by any other Google Drive related folder.

Suggested task distribution:

We recommend you working in a group of 3 students, and suggest the following task distribution accordingly:

- Student 1: Google Drive API connection and operations.
- Student 2: Master code specially synchronizing with Google Drive.
- Student 3: Follower code specially synchronizing with Master.

Good Luck!