Name: LI Wei

Login ID: liw4

Student ID: 654719

Client-server asynchronous implementation

First, client sends message to server each time. Client is sender and server is receiver. When client sends each message, server will be able to receiver. Client records the number of sends. Server also has record the number of receives. Sender has few steps to process asynchronous, each cycle time, sender have including “data”, “start send data”, “stop send this data”. When sender has a lot of data that needs to be sent, it is only processes the send, it is does not response from receiver. So, this one of specifies cycle time, number of data counter will be increase until next cycle time counter will be setting to initial counter one and start increase again. Similarly, receiver has record number of receive counter in each of cycle time. Receiver counter will be increased in each cycle time. When each specifies cycle time to finish receiver counter will be reset. So, receiver also has no communication with sender for asynchronous. However, if receiver has missing some data from sender or sender has missing some data to send. Sender will continue to send the rest of data in each cycle time. Then if server has not received specific data, server will be holding on this position for sender to send the data again. When send finishes by each of cycle time, it will restart send data. So, server will be receiving correct data by sender in next cycle time. The diagrams follow by Figure1 in below:

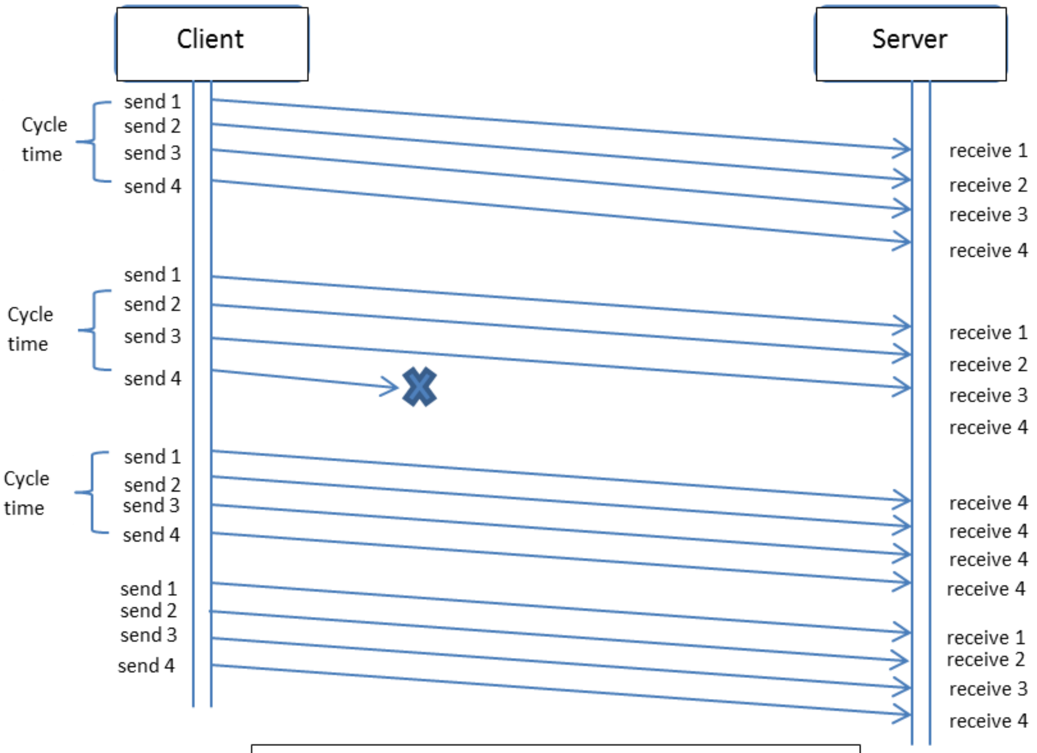

Figure 1: show Client-server asynchronous

Arbitrary block sizes use UDP

Block size only can be allowed in UDP to be up to 65507 bytes. If block size is more than 65507 bytes UDP will not work. So, the program needs to separate huge data into small parts and store each part. Then, system can separate process small part; there have one simple way to do it. It is fragmentation, which is to fragment data to many sub groups. The process methods are system receives specific data, and this data is over the allowed range in UDP. Fragmentation will be used to fragment this specifies data to different part. The program will be set thread for define the subpart size. For this case, program will be set less than 65507 bytes in each of parts. Then, those sub parts of data will be used in UDP to send to the receiver to process. System will mark each sub-part data when system use fragmentation to implementation this case. The program will use Boolean to check this specific data whether or not fragment part. Then, when server has received all of fragment parts from client, then it will combine all of parts together. Some variable value will let the server know which part of fragment is last part then server can combine all of data. The diagrams follow by Figure2 in below:

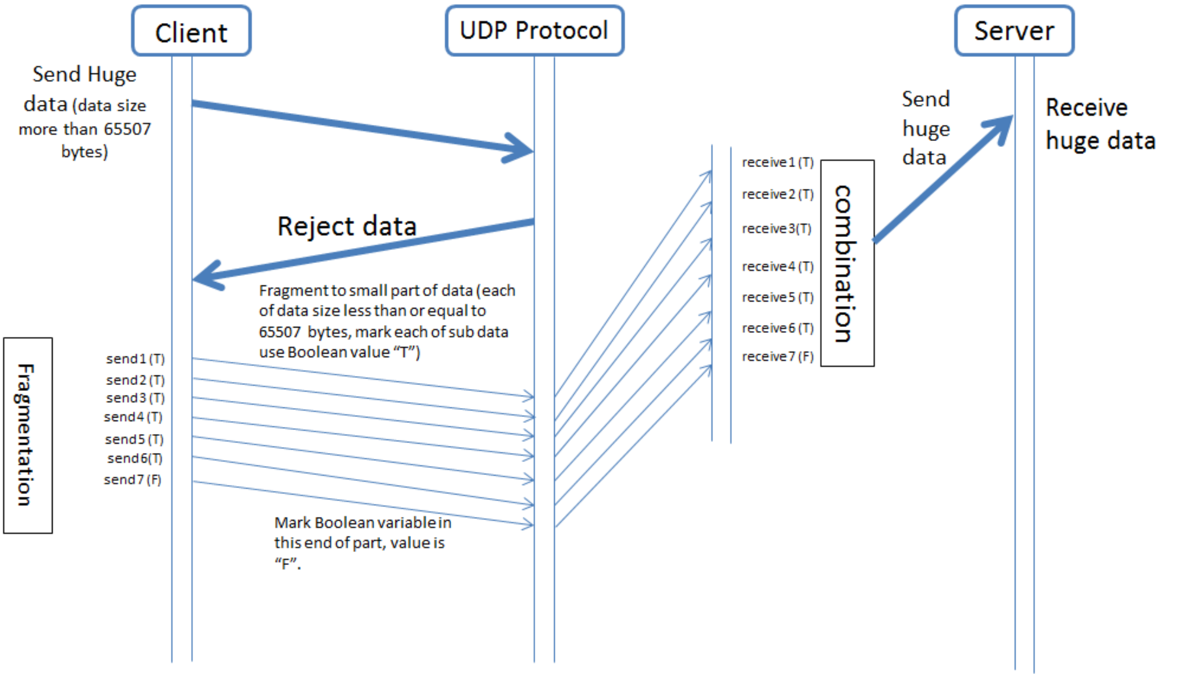

Figure 2: Show the arbitrary block size using UDP

Research

Dropbox

Dropbox is a file hosting service system operated by Dropbox Company. This system mostly provides two functional parts which is cloud storage and file synchronization. Cloud storage is a server that can process many client requests. For example, when client synchronize some data in server, he/she can use another device to download and continue to process this data. Another condition is when user synchronizes data in server. Not only will the user assess this data, but also someone else can assess this data if the user can give that person permission to assess this data. The synchronization in Dropbox which uses the client software to manage local storage and synchronize the server. For example, when a user changes, adds, or deletes some data in storage. If it is done, software will be send message to server command server to modify the data. After that, user can assess data in other devices. According to this, Dropbox has some problem for using. For example, first problem is security. Another problem is when comparing with another synchronization system; it will be have less than synchronize speed. Moreover, the space will be able to the problem for Dropbox. The cloud storage diagrams follow by in below:

Comparing

In above, it is describe about cloud storage which is use specifies software "Dropbox". However, cloud storage still has some problems compare with peer-to-peer encrypted file synchronization. So, I will introduce a different software, which is "AeroFS". The first problem about Dropbox is security. Dropbox is cloud storage that means server can service any of clients. For this situation, client will be able to assess any of data in server. For example, when user wants to share the data to another person, he/she only need to share the link to person. But, AeroFS will be use different way to protect user, when user to want to share specifies data to another person, user need to pass specifies security key to person, when person have permitted to assess data. The second problem is speed about synchronization. In Dropbox, user to synchronize data in server, user can share the data to anyone. This means the shared data will be many people. The speed of synchronization will be distributed to those people. So, speed will be slower. But, when a user shares data in AeroFS, each person will be receiving one specific key. That means is one by one to share the data. So, the speed will be fast. Last problem is storage space, when user gives permission to another person. This person can operate this data, even add some information. This will decrease the free space, but AeroFS needs to have different keys for each person that wants to process the data. If this key is "only read" the data, the person only be able to read the data. Moreover, AeroFS is peer-to-peer system. Means is one person can only have one space.