

COP5615 – Operating System Principles – Fall 2006

Document Version 1.0

Project 3 – Protocol Design Phase

Assigned Thursday, October 05, 2006

Due Date: October 12, 2006 (Thursday) 4:59 pm EST (on campus)

October 14, 2006 (Saturday) 11:59 pm EST (EDGE)

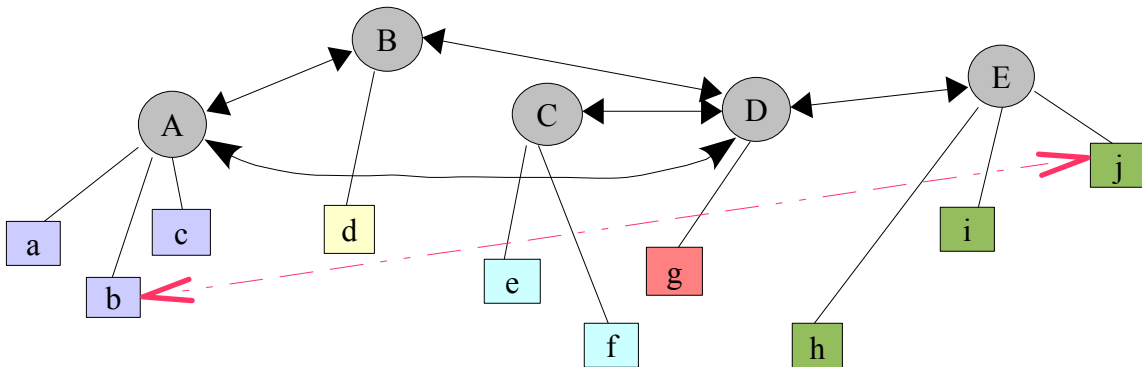
SPECIAL NOTE: THIS IS JUST THE PROTOCOL DESIGN PHASE OF PROJECT 3 AND THE DELIVERABLE IS JUST A CONSISTENT AND COMPLETE PROTOCOL SPECIFICATION. INTER AND INTRA TEAM DISCUSSION AND COLLABORATION IS ALLOWED AND ENCOURAGED FOR THIS PHASE.

General Information:

This is a team project and you are supposed to work in teams of no more than 4 students. Please decide on the team members and email TA with team members' list latest by October 10, 2006. Teams once formed are not allowed to change their members later on for the rest of the semester unless permitted by the professor.

Project Description:

Continuing from project 2, in this project we will extend the scenario to include multiple chat servers. Refer to the representative diagram shown below [this is just an illustrative example]:



The circles in the above diagram represents chat servers implemented in project 2. The square elements represents the user nodes connected to the servers. Each user node consists of user agent as well as end user modules developed in project 2. In project 2, peer-to-peer communication was disabled, which will be re-enabled in project 3. Also the servers will have the capability to communicate with each other.

Teams are required to come up with a server discovery protocol so that servers can discover any available nearby servers. When a nearby server is discovered, the server is required to distribute its list of attached users among its neighbors in order to facilitate fault tolerance in the design. Also when a nearby server is discovered, its details are to be passed to the attached nodes as well.

When a server fail, nodes connected to the server must be able to detect the failure and reattach itself to one of the nearby servers. If they are not able to reconnect to any other server, the nodes must die. Later when the crashed server becomes online, it may query its neighbors (it can dump the neighbor list in a file periodically) to find out what nodes were under its jurisdiction, and then may contact those nodes asking them to reattach them to itself. Also the nodes attached to a server may search for nodes connected to distant servers by querying its own server. That server should then query its neighbor server nodes and so on till either the requested node information is found or no such node exists in the connected server graph. The requesting node then can

establish a direct connection to the far-away node via peer-to-peer connection as implemented in project 1.

List any additional commands that must be implemented along with the command structure in the end-user.

A sample demonstration of the above described feature is given below:

nodes a, b, c are attached to server A. Server A prevents alias name collision under its domain. Additional requirement will be that name collision now must be enforced across multiple chat rooms as well. Hence nodes a, b and c can now be named hierarchically in a globally unique fashion. Global names of a, b, c will now be A.a, A.b and A.c. Of course this necessitates that the server themselves are named uniquely. This can be enforced during the server discovery phase, when a server discovers a new neighbor, it confirms that the server names are unique, in case of a collision one of the server must automatically change its alias name. A.a, A.b and A.c can still engage in chat room communication using multicast as done in project 2.

Now suppose A.b wants to talk to node j, it send a search request to its parent server A, A in turn sends the search request to its neighbors B and D, D propagates the search request to its neighbors and so on. Finally E finds j listed as its user and it responds back to D which responds back to A and it sends the search result to b, now b can send messages directly to j using send command implemented in project 1.

Also needed is a strategy to make a server discoverable by another subset of servers but not all the current live servers. [Hint: random number range set + multicast may be used].

Combination of multicast and UDP unicast may be used in designing the protocol specification. STRICTLY NO TCP COMMUNICATION IS ALLOWED.