Name: ______________________________

CS 6/73201 # Exam #1 **Advanced OS**

Monday 3 October 2005

1. **Consider the "workstation" distributed system model, where there are many workstations on a network, but one workstation per user, and where the system automatically migrates processes to idle workstations. In what ways does this fit the definition of distributed system? In what ways is it not a distributed system? (20 points)**

   It fits the distributed system from an architectural standpoint — multiple loosely-coupled processors, each with its own clock and memory. From a software / OS standpoint, it fits the definition in some ways (resource sharing, scalability, process migration, etc.) yet is not transparent as a true distributed system since the user does have a distinct tie to one particular workstation.

2. **Ethernet uses an access control protocol called CSMA/CD (Carrier Sense Multiple Access with Collision Detection). Briefly describe how this protocol works. (15 points)**

   A host listens before broadcasting, and defers until the channel is clear. Then it broadcasts, but listens to the network while broadcasting. If what it receives is not what it sent, it has detected a collision. In that case, it jams the network, waits a random (but increasing) amount of time, and tries again, continuing until it successfully sends the message.

3. **In Christian's algorithm for synchronizing physical clocks, a computer sends a request to a time server, and measures the time delay *D* taken to receive a time reply *T*, and then sets its local time to $T + D/2$. What assumptions are this algorithm making? (10 points)**

   That the network delay is consistent — specifically, that the time delay for the request to reach the server is the same as the time delay for the request to be returned from the server. It also ignores the computation time on the server.

4. **In the Berkeley algorithm for synchronizing physical clocks, a master computer periodically polls the slave computers, observes the time delay taken to receive each reply, estimates the slave computers' local clock times, takes a (fault-tolerant) average, and then sends each slave computer the amount (positive or negative) by which it should adjust its local clock time. Why does it send an adjustment amount, instead of just sending it a new local clock time? (10 points)**

By sending only the adjustment amount it does not have to account for the network time delay required for its response to reach to the slave computer.

5. Lamport's logical clocks implement the "happened before" relationship, but have two problems: (1) they define a partial order rather than a total order, and (2) although it is true that if *a*→*b* the clock value of *a* is less than the clock value of *b*, it is not true that if the clock value of *a* is less than the clock value of *b* then *a*→*b*. Explain why each of these problems is considered to be a problem, and how it can be overcome. (15 points)

Partial order rather than total order is a problem if the algorithm requires a total order (e.g., sorting requests into a queue as in Lamport's distributed mutual exclusion algorithm). This problem can be overcome by using some other relation to break ties (e.g., process ID).

Not being able to compare clock values to determine "happened before" is a problem because determining which event happened before the other is what we need for many algorithms (see above), yet with logical clocks, if one event has a lower time stamp than another it may not necessarily have "happened before" — it may be concurrent. This problem can be overcome using vector clocks.

(Be sure to read the question carefully. Some students explained why only one of these two problems was a problem and how it could be overcome, and thus got only half credit.)

6. Consider Chang and Roberts' Ring algorithm for election.

a. There are two types of messages sent in this algorithm — "election" messages and "elected" messages. What is the purpose of each type of message? (10 points)

The "election" messages pass around the ring the highest process id seen so far, and thus serve both to notify processes of an election as well as to handle the mechanics of electing the process with the largest id.

The "elected" messages propagate the id of the elected process to the entire ring.

b. At various points in the algorithm a node is marked as a "participant" or a "non-participant". Would the algorithm still work without marking nodes in this manner? Explain. (5 points)

The algorithm would still work, but not as efficiently — the main purpose of marking nodes as "participant" is to reduce the number of messages sent when multiple elections are under way simultaneously.

7. Compare the Byzantine agreement problem to the Consensus problem. In what ways are these two problems similar? In what ways are they different? (15 points)

Similar —in both Byzantine agreement and Consensus, all processors agree on a single value (unlike in Interactive Consistency, where they agree on a vector of values).

Different — in Byzantine agreement, one processor broadcasts a value, which is the value all nonfaulty processors should agree on, while in Consensus, all processors broadcast a value, and all nonfaculty processors can agree on any value (e.g., the average of the values of all nonfaulty processors).