CMPSCI 677: Distributed Operating Systems

Final Exam

May 16, 2003

General instructions:

- Please typeset your solutions if possible. If you hand-write your solutions, please be legible.
- This exam is due in 27 hours (at noon tomorrow). You can email your solutions in pdf, postscript or plain text format (no Word documents please) or drop off a hard-copy. Off-campus students can use email or FAX. If you use email, please send your exams directly to Gary Holness (gholness@cs.umass.edu). To drop off a hard copy, please stop by the CS Ed-Lab (room 225 in LGRC) between 11am and noon—Gary will be available to accept hard copies.
- Do not forget to put down your name and student number on the exam books.
- If your answer depends on any specific assumptions, please state them clearly.
- This is an open book, open notes exam. *You shall not seek help from or discuss the exam with ANY human. If you do, you will be awarded a zero for the exam and will be given a grade of "F" for the course.* Send me email if you need a clarification on any question.
- Explain your answers clearly and be concise. *Do not write long essays.*
- Good luck.

---

1. **Short answer questions** (30 points)
   Answer the following questions in brief.

   (a) (6 pts) For performance reasons, you have been asked to rewrite a distributed application written in Java and Java RMI using C++ and RPCs. The Java version of the application passes object references between processes running on different machines. Since RPCs do not support pass by reference, do you think it is impossible to rewrite this program in C++ using RPCs? Justify your answer.

   (b) (6 pts) It is possible to implement a DNS server using LDAP. True or False? Briefly justify your answer.

   (c) (6 pts) Consider a distributed system with N workstations where the *average* system-wide load is very small but there is high variance in the load across workstations. Would you choose a sender-initiated or a receiver-initiated load balancing policy for this system? Explain your answer.

   (d) (6 pts) Totally ordered multicast implemented using Lamport's clocks will always guarantee causal delivery. Provide a simple proof if you think this statement is true, otherwise provide a counter-example.

(e) (6 pts) Consider the ring-based leader election algorithm with 5 processes numbered 1 through 5. Assume that process 5 is the leader and crashes. Further, assume that processes 1 and 3 notice this crash and initiate simultaneous elections. What happens if process 5 rejoins the ring after the *Election* message from 3 has gone by but before the *Election* message from 1 reaches it? (*Hint: think about what happens when the two concurrent elections yield different results.)*

2. **Distributed File Systems** (20 points)

(a) (10 pts) Coda uses transactional semantics to guarantee serializability of writes to a file. Can you use the two phase locking protocol discussed in class to implement serializability semantics within the Coda file system? Justify your answer.

(b) (10 pts) Consider a distributed producer-consumer application that uses a file as a buffer. The producer writes some data to a file and then signals the consumer by sending an ACK on a network socket. Upon receiving an ACK, the consumer reads data from this file and processes it. Assume that you write such an application and debug it fully by running both the producer and consumer on your PC and having the shared file on your local disk. Do you expect your debugged application to work without any problems when you deploy the producer and consumer on two different machines and use a shared file stored on a NFS server. Assume that the machines running the producer and consumer NFS-mount the directory containing this shared file, so that the file is accessible exactly with the same name as before. Explain your answer.

3. **Web and Streaming Servers** (25 pts)

(a) (12 pts) You are asked to implement a streaming server to stream digitized lectures videos for this class to off-campus students. You are given a Pentium 4 server with 512 MB memory, a gigabit ethernet interface (1000 Mb/s) and a SCSI disk with 10,000 RPM rotational speed, 5ms average seek, 5 ms average rotational latency and transfer rate of 10 MB/s. Assume that all lecture videos are compressed with MPEG-1 prior to storage. The data rate of each video is 2 Mb/s and all files are stored using a disk block size of 128 KB. Each lecture is 90 minutes in length. How many off-campus students can be simultaneously supported using your server? Assume a server push architecture where data is pushed to each user once every second. State your assumptions clearly.

(b) (13 pts) Consider a group on n web proxy servers that employ cooperative caching. Upon receiving a request for a web page, a proxy examines its local cache for the file. A cache hit is serviced locally. A cache miss causes the proxy to ask the other $n - 1$ proxies if they have the object. If so, the object is fetched from one of these proxies, and if not, from the server.

Assume that the web server employs the leases protocol to maintain consistency of web pages cached at a proxy. Will the leases protocol discussed in class suffice for maintaining consistency of the cached data in such a n proxy group with cooperative caching? If so, explain how leases can interoperate with cooperative caching . If not, explain how leases might be extended to achieve consistency in the presence of cooperative caching.

4. **Distributed Middleware and Security** (25 pts)

(a) (8 pts) Consider a distributed implementation of a JavaSpace in which tuples are replicated across several machines. Suppose that you wanted to implement an *update* method in Jini, whereby a process can update an existing tuple. Provide a protocol whereby race conditions are avoided when

two process try to update the same tuple. Be sure to explain all steps of your protocol in detail. How would your protocol change if you wanted to prevent races for deleting a tuple?

(b) (7 pts) Suppose that you were to implement the centralized Renaldo game server that you built in Lab 1 in DCOM. What features of DCOM would be especially useful for implementing your game server? What performance improvements or degradation might you notice as a result of using these DCOM features?

(c) (10 pts) Consider a smart card as an example of *digital cash*. A smart card is a credit-card size card with an inbuilt chip and storage. Digital money is stored on your smart card. To add money to your smart card, you go to your bank's ATM, enter your card/PIN and withdraw money from your checking amount; this amount is then digitally deposited by the ATM onto your smart card. To make a payment, you use insert the card into a terminal at the merchant, enter the amount to be paid, and enter a PIN (much like using a debit card or an ATM card at a merchant). The terminal then verifies you have sufficient money on your card, and subtracts that amount from the smart card.

- How can a terminal verify your smart card has "real" money on your card? Assume that the terminal is not on a network and can not contact your bank in real-time to verify that your digital money is not counterfeit. Further assume that the merchant and you use the same bank.
- How can the bank identify someone who double-spends their digital cash? Double spending occurs when a dishonest user creates an identical copy of his/her smart-card and uses these cards at two different merchants. *(Hint: you only need to identify double spending after it has occured and not in real-time.)*