Stanford University
Computer Science Department
CS 240 Sample Quiz 3 Questions
Spring 2004

May 31, 2004

This is an open-book exam. You have 50 minutes to answer all questions. Write all of your answers directly on the paper. Make your answers as concise as possible. Sentence fragments ok.

NOTE: We will take off points if a correct answer also includes incorrect or irrelevant information. (I.e., don't put in everything you know in hopes of saying the correct buzzword.)

Stanford University Honor Code

In accordance with both the letter and the spirit of the Honor Code, I did not cheat on this exam nor will I assist someone else cheating.

Name and Stanford ID:

Signature:

Answer all of the following questions and, in a sentence or two, say *why* your answer holds. Your answer should be concrete enough that it is readily distinguishable from someone who has jotted down a string of high-level buzzwords.

1. Pick some useful optimization from one of the papers that we discussed this quarter (besides the superpage paper) and describe how to implement it within a virtual machine monitor. Be concrete as to any difficulties that come from doing it in the the VMM rather than in the OS itself.

2. **Question 5: Yes, we have no RAID (20 points)**

   (a) (5 points) Assume you have a simple RAID system: four data disks (A,B,C,D) and one dedicated parity disk (P) that holds the bitwise xor of the data disks ($P = A\ xor\ B\ xor\ C\ xor\ D$).

   If the OS crashes while writing to the RAID system, what do you have to do after the crash to recover to a good state? What happens if a disk blows up during this recovery? (Assume that all redundant information lives on the parity disk and that you do not have a log.)

   (b) (10 points) Why might a log structured file system (LFS) run substantially faster than an update-in-place file system such as FFS on a RAID system? Give two weak points of LFS: do you expect these to be worse or better on a large RAID system?

   (c) (5 points) Jimbo likes your work and gives you an additional dedicated parity disk for the system above. Draw a picture with all six disks (two parity, four data) that shows how you'd organize your system so that it can survive *any* two disks blowing up.

3. You are cleaning segment 7, which contains a file block 10 immediately followed by its inode, 12. If we can reclaim the space for block 10 are we guaranteed that we can reclaim the space for 12? Similarly, if we can reclaim the space holding inode 12 do we know that the space for block 10 is free as well?

4. Suppose you have a RAID 5 array with 6 disks. What is the best case number of simultaneous writes we could make? What is the worst case? (Please clearly state any assumptions you make.)

5. Assume you are still using RAID 5: what RAID-specific recovery will you have to do after a crash to get to consistent state? Is there a problem with crashing during recovery?

6. Disk drives typically only guarantee that individual 512-byte sectors are written out atomically. What implications does this have for logs?

7. Log checkpoints are supposed to allow for faster recovery after a failure. Describe a scenario where checkpoints can make recovery slower.

8. Butler makes the point that quantizing a resource into fixed sized chunks can significantly help resource management. (E.g., clothing is quantized into S, M, L, or XL rather than a continuous set of sizes.) Give three examples from the papers we read where such a trick has been used.

9. Eraser: give a *short* code example (and the intuition behind it) showing how we can get a race if only one thread ever writes to a shared variable.

10. IRIX does not have superpages. Jimbo wants superpages. *Briefly*: how could you trick IRIX into using superpages using disco? Give a concrete example of doing so is worse than doing it in the OS itself. If you statically partitioned memory across guest OSes, might you actually see performance improvements?

11. Last time I checked, Jerry Saltzer and Dave Clark believed that CPUs should not have errors. Can you reconcile their stance with the end-to-end argument?

12. Now that you've had to wade through a bunch of experimental evaluations, what experiments would you use to evaluate the VMS approach to virtual memory?

13. From Figure 6-1 in the livelock paper: Why does the line go up? What determines how high the peak is? What determines how fast it goes down?

14. Jimbo likes everything, including LBFS. It's his birthday so you implement LBFS as a local file system on his disk using the Boneh-in-a-Box co-processor, which makes both fingerprint and signature computations infinitely fast. Compared to a traditional local file system, give

one way that local-LBFS (1) reads will be slower, (2) reads will be faster, (3) writes will be slower, (4) writes will be faster.

15. You implement LBFS in Disco so that you can slip it underneath the guest OS. Does this cause any problems?

16. What would be the dual of a race condition in a MPP system?

17. Give two situations where Disco could have better sharing than ESX.

18. Give the intuition behind the explanation for why Disco is 36% slower in Figure 7, while the overheads in Figure 5 range from $3 - 16\%$.

19. You've read a lot of papers. Obviously, each paper does not blast into new territory with completely new methods. Rather, research follows a small set of patterns (e.g., speed research is a big one) and focuses on a small "canon" of accepted problems (e.g., making file systems fast).

    Give a concrete pattern you have observed and at least three concrete examples that fit the pattern. (A true, but shallow observation will get less points than a deep, interesting one.) *Please do not write a "war and peace" essay — succinct intuitions are fine.*