Systems Comprehensive Exam, Fall 2005

August 16, 2005

1 Instructions

This is a closed-book, closed-notes exam with a total of 100 points. You may not use any external source for answering these questions. Please direct any questions about this exam to Professor Bridges. Professor Bridges may be reached either in person in his office in 301B Farris, by phone at 277-3032 or 363-8798, or by email at `bridges@cs.unm.edu`. Turn your exam in to Professor Bridges or the front office by 5:00 PM MDT on Tuesday, August 16, 2005. Exams *will not* be accepted after this time except by prior arrangement with Professor Bridges.

Type or write your answers to the stated number of questions in each of the following three sections. Make any *reasonable* assumptions necessary to answer the question, but be sure to state any assumptions that you make.

2 Short Answer - Answer 3 of 4 (30 points)

Briefly answer 3 of the following 4 questions. Your answer should be no longer than *one* paragraph.

1. **Deadlock.** What is deadlock, and what constitutes necessary and sufficient conditions for causing deadlock?

2. **Memory Management.** What is the optimal page replacement strategy in a paged virtual memory system? Name one algorithm frequently used to approximate this strategy.

3. **File Systems.** What is a journalling file system, and what advantage does it offer over more traditional UNIX file system implementations?

4. **Network Routing.** What is the difference between distance vector routing and link state routing?

3 Medium Answer - Answer 2 of 4 (40 points)

Provide detailed answers to two of the following four questions. Be sure to state any assumptions you make and to fully justify your answers. Limit your answers to approximately one to two pages in length.

1. **Caching.** Modern cache systems have a wide range of design parameters, including:

    - Unified versus split caches
    - Direct-mapped versus associative caches
    - Cache size, cache access time, line length, and number of cache levels

    Define each of these terms, the trade-offs associated with each parameter, and how changes to them effect capacity, compulsory, and conflict miss rates.

2. **Virtual Memory.** Describe *inverted page tables*, how they are used for memory mapping, and their advantages and disadvantages compared to traditional hierarchical page tables.

3. **Consistency in Group Communication Systems.** Causal consistency in group communication systems is normally implemented using one of two approaches: vector timestamps or a sequencer process. Describe each method and compare and contrast the advantages and disadvantages of each.

4. **Policy and Mechanism in Modern Operating Systems.** Hypervisors and microkernels are both modern operating system design strategies, one oriented toward multiplexing multiple operating systems on top of a virtual machine monitor, and the other oriented toward dividing OS functionality between multiple servers. How do these two approaches affect how policy and mechanism are traditionally divided in the system?

4 Design - Answer 1 of 2 (30 points)

Provide a *full* and *detailed* answer to one of the following two questions. Be sure to state any assumptions you make and to fully justify your answer.

1. **Scheduling and Load Management.** Scheduling and load management are core operating systems problems that also occur in real life systems, not just computer systems. Consider an elevator system of four elevators cars with limited occupancy that services a tall building. In this system, in addition to servicing floors that elevator occupants select, each elevator also stops on every floor with an external call signal active (i.e., where the elevator call button is pushed).

   - What metrics are appropriate to determining the quality of service provided by such an elevator system?
   - How can such a system fail to provide adequate service under heavy load?
   - Are the metrics the system seeks to optimize different under light and heavy loads?
   - How can the system be changed to provide adequate service under heavy load, for example by having separate service policies for light and heavy load?
   - In such a system, how and when would the elevator system switch between policies for light and heavy loads?

2. **Future Operating Systems.** Architectures are advancing to the point where CPU cycles are almost completely free on emerging systems; memory and I/O systems are now becoming the predominant bottlenecks in these systems. For example, desktops in 5 years will likely have 2-4 processors each with 8 processing cores on them. Operating system designers may take advantage of this wealth of CPU power by changing OS designs by, for example, dedicating a processor to OS processing. How will this wealth of processors and cycles change fundamental OS policies, including scheduling, I/O, and memory management? How could hardware be changed to allow the operating system to make more effective use of this abundance of CPU power?