CPS 110 Final Exam

Spring 2011

Please answer all questions for a total of 300 points. Keep it clear and concise: answers are graded on content, not style. I expect that you can answer each question within the space provided. If you need to make any assumptions that are not clear from the question, then please state them explicitly. Any kind of pseudocode is fine as long as its meaning is clear. You may assume standard routines like lists, queues, hash tables, etc.

You are allowed one sheet of notes; please do not refer to any other sources of information such as other students, laptops, tablets, phones, books, notes written on your hand, etc. Please sign this page to indicate that you have completed your exam within these rules. Good luck!

Name:

Signature:

Problem	Max	Score
1	40	
2	40	
3	50	
4	30	
5	30	
6	30	
7	40	
8	40	
9	!	
total	300	

1 Storage: Short Questions (40 points)

Answer the following short questions about storage.

(a) What is *metadata* in a file system? List a few examples.

(b) File system software often must ensure that a write of some dirty block (call it A) completes before a write of another block (B) completes. Why? Give an example of what could go wrong if write order is violated and B is written to disk before A.

(c) Storage system software that uses logging must have a means to detect any incomplete writes at the end of the log during recovery after a failure. Why? Give an example of what could go wrong if the system failed to detect that some log write was incomplete at the time of the failure.

(d) Why does RAID-5 distribute parity information across the disks in an array, rather than storing it on a designated parity disk, as in RAID-4?

2 Security: Short Questions (40 points)

Answer the following short questions about security and cryptosystems.

(a) A login server must verify a user's password before accepting a login. Secure servers often keep a list of hashes for user passwords, rather than the passwords themselves. Why is it desirable to use a hash function in this case?

(b) Digital signatures are often used to verify the integrity of a document. A digital signature is created by encrypting a hash of a document, rather than by encrypting the document itself. Why is it desirable to use a hash function in this case?

(c) Outline one way in which digital signatures are used in the Secure Sockets Layer (SSL) protocol.

(d) If a malicious entity ("Mallory") controls a Certifying Authority accepted by a user Web browser ("Alice"), then Mallory can mount a man-in-the-middle attack against Alice's session with an arbitrary Web site ("Bob"). Briefly outline how this is done: show the transfers of keys between Alice, Bob, and/or Mallory.

3 Operators are Standing By (50 points)

The customer service line at MegaMurk staffs a phone bank to answer questions from customers about its products. Customers calling in are put on hold, where they wait for the next available customer service advocate to answer their call "in the order it was received". The economy is in a downturn, so business is sometimes slow: if there are no customers waiting, available advocates just wait around for the phone to ring.

You are to write procedures to synchronize the customers and advocates. You need not be concerned with starvation, fairness, or deadlock. Use **semaphores** for synchronization.

When a customer thread has a question, it calls customer() to bind to an advocate, waking one up if necessary, and waiting if none are available. When an advocate thread is available to serve a customer, it calls advocate() to bind to a customer, waking one up if necessary, and waiting if none are available. When these procedures return, the threads converse and go on their way. Write the customer() and advocate() procedures.

4 Back of Napkin Part I (30 points)

This question asks you to sketch some qualitative "back of napkin" graphs for a well-engineered service (RPC, file, web) running on a single server. You may make standard idealized assumptions as in standard queuing theory. For example, we can suppose that performance is limited by a bottleneck resource, such as a CPU, i.e., performance is determined by the utilization and per-request service demand of the bottleneck resource.

(a) Plot throughput as a function of offered load (or utilization). Label any interesting features of the graph.

(b) What server structuring choices might cause performance to fall short of this ideal, and why?

(c) Suppose the bottleneck resource is a disk: each request does a random read from the server's disk. Now suppose that we replace the disk with a RAID array: sketch a new line on the graph to show the likely impact on throughput. Briefly discuss any new assumptions necessary for your answer.

5 Back of Napkin Part II (30 points)

This question continues with sketching qualitative "back of napkin" graphs for the well-engineered service of the previous question.

(a) Plot average (mean) response time R as a function of offered load (or utilization). Label any interesting features of the graph.

(b) Label the graph to show the relationship between average response time and average per-request service demand (D). What is the relationship between R and D when the server is lightly loaded?

(c) Suppose the per-request service demand (D) doubles: sketch a new line on the graph to show the likely impact on response time. Briefly discuss any new assumptions necessary for your answer.

6 Back of Napkin Part III (30 points)

This question continues with sketching qualitative "back of napkin" graphs for the well-engineered service of the previous question.

(a) Sketch a Cumulative Distribution Function (CDF) graph of response time at a moderate load level, say, 50% utilization. Label any interesting features of the graph.

(b) Suppose the server switches to using STCF (also called SRPT or SJF) scheduling for its bottleneck resource: sketch a new line on your response time CDF graph to show the likely impact of this choice on response times. Briefly discuss any new assumptions necessary for your answer.

7 Déjà vu on Project 2, Part I (40 points)

This question pertains to the clock queue in project 2, which is maintained by the page replacement algorithm. The clock queue is an ordered list of all valid, resident virtual pages in the system. It is similar to the active+inactive list in the FIFO with Second Chance page replacement algorithm.

(a) What are the settings of the page protection bits for a page that is placed on the tail (back) of the clock queue?

(b) What does your pager do with a page that arrives at the front of the clock queue?

(c) The project specification says: "The order of pages in the clock queue may differ from the order of their physical page numbers." Why? Illustrate with an example.

8 Déjà vu on Project 2, Part II (40 points)

This question concerns your code in Project 2 for managing backing storage (i.e., disk) for user virtual memory.

(a) Describe the data structure that tracks allocation of disk space (i.e., tracks what storage is free vs. allocated)?

(b) Describe the data structure that tracks the disk location for each virtual memory page.

(c) Under what conditions does your pager allocate disk space for a virtual page? Describe an event that would cause this to happen.

9 Something Extra: Group Experience

Tell me a little about more about your projects.

(a) In Project 3, what shellcode did you use for your exploit?

(b) In Project 3, how did you connect to the exploit shell to execute commands? What tool did you use? Outline how the connection is established.

(c) Briefly describe your most interesting test case for the pager in Project 2, or the most interesting bug exposed by a pager test case.

(d) Did your group obtain any code for Project 1 or Project 2 from any sources outside of your group (excluding instructor or TAs)? Describe.

(e) Does your group have a "leader"?

(f) Does your group have a "straggler"?