Final Examination
CS 111, Fall 2016
UCLA

Name: _____

This is an open book, open note test. You may use electronic devices to take the test, but may not access the network during the test. You have three hours to complete it. Please remember to put your name on all sheets of your answers.

There are 3 questions on the test, each on a separate page. You must answer all of them. Each problem you answer is worth 33% of the total points on the test. You get 1% for putting your name on the test.

You must answer every part of every problem. Read each question CAREFULLY, make sure you understand EXACTLY what question is being asked and what type of answer is expected, and make sure that your answer clearly and directly responds to the asked question.

I am looking for depth of understanding and the ability to solve real problems. I want to see specific answers. Vague generalities will receive little or no credit (e.g., zero credit for an answer like "no, due to the relocation problem."). Superficial answers will not be sufficient on this exam.

Organize your thoughts before writing out the answer. If the correct part of your answer is buried under a mountain of words, I may have trouble finding it.

1. Many file systems end up storing the same bit patterns over and over again for many different files, effectively wasting a lot of disk space storing redundant data. Some file system deal with this issue by using deduplication, a technique that tries to ensure that any bit pattern occurring more than once anywhere in the file system is only stored on disk one time. Other files containing the same bit pattern do not store unique copies of the data, but instead "point to" (in some fashion), the single copy. Of course, one must choose some granularity (in terms of length of bit patterns) at which one deduplicates. There would be no benefit in deduplicating all occurrences of the bit pattern "0", for example. (Note: if you propose some technique other than deduplication for dealing with this problem and do not address the points below concerning deduplication, you will get 0 on this question. Don't do that.)
    a. What granularity would you suggest deduplicating at? Why?
    b. What algorithms and data structures will you use to detect duplicates?
    c. Should deduplication be applied to directories or just to regular files? Why?
    d. Does the use of deduplication cause any new security problems? Why?
    e. Describe how you would alter the FFS to allow it to perform some useful form of deduplication. Describe the changes to metadata structures that your alteration would require. How would the following file operations change (in their underlying OS implementation) in your design: `create()`, `open()`, `read()`, `write()`, `unlink()`?
    f. Will deduplication improve, degrade, or not alter file system consistency concerns? Explain.
    g. Will deduplication perform better when the underlying storage medium is a hard disk drive or a flash drive, or will it have the same performance for both? Why?

2. Each of the following three situations poses interesting allocation or locking problems (figuring out what or how to lock while avoiding deadlocks, hangs and bottlenecks). For each resource, (1) describe the best approach (2) justify why the situation demanded this approach, and (3) tell me specifically how you would apply that approach to the problem.

(a) A cellular communications processor that controls calls, power-levels, and the movement of calls across adjacent cells supports millions of calls and hundreds of concurrent operations. To prevent conflicting updates, there are locks associated with each transceiver, call, and cellular sector. Some operations start by locking a single object (e.g., a call) but as they progress it becomes clear that it will also be necessary to lock other objects (e.g., one or more sectors). How can we prevent concurrent multi-object operations from deadlocking?

(b) Swap space on secondary storage. If there is not room on the swap device to swap out one of the in-memory processes, we will be unable to make room to swap in and run any of the processes that are currently swapped out. How can we prevent such a situation?

(c) A network lock manager provides locking services for a very wide range of distributed resources that are shared by thousands of clients. The clients are a wide variety of applications running on many different operating systems, and not all of the resources they need are managed by the network lock manager. How can we ensure network locks will not contribute to deadlocks among the client applications?

3. A company with around 500 employees in several offices around the globe wishes to provide its employees with a wide range of software services hosted at several company server machines. They expect to add, remove, and alter services regularly, and they expect to host hundreds of such services. They wish to maintain tight access control on the services, with much more specificity than merely enforcing that particular users are or are not allowed to use the service. Instead, for each service, the service designer must be allowed to specify particular elements of the service that are under access control. (For example, one service might want access control on whether the user is allowed to print results from the service, while another might want access control on whether a particular user is allowed to escalate a problem to a higher ranking employee, and a third service might use access control to permit only system administrators to determine how much of a server's CPU use a particular user is expending, while not allowing the administrators to see the specific tasks being performed.) Since the company doesn't know all services they will ultimately offer, they cannot specify all service elements that might need access control, but they know that not all services will need access control for all possible elements. The service designer should be able to specify which actions offered by his service are under access control. (For instance, service A might want access control on printing, while service B does not.) The access control mechanism is to be built into the operating system, to ensure trustworthiness, uniformity, and to protect from application bugs. Users will occasionally be added or deleted, and access permissions for particular users and services will change often.

   How would you design the operating system mechanism to provide this kind of access control? Consider the required scale, flexibility, performance, distributed nature of the problem, and security issues, at the minimum.