UNIVERSITY OF WISCONSIN-MADISON
Computer Sciences Department

Operating Systems **Spring 2016**

Instructions: There are *six* questions on this exam; you must answer *all* of the following questions.

Question #1: Exokernels and Hypervisors

Exokernels and hypervisors have many similarities, in that they multiplex and protect hardware with minimal abstraction.

1. How does the exokernel manage address translations and memory reclamation (i.e., page replacement)?
2. How does a hypervisor like Disco or VMware manage address translations and memory reclamation?
3. What performance optimizations are available in one system (Exokernel or hypervisor) and not available in the other system design? Explain your answer.
4. People have proposed running library operating systems in hypervisors. Suppose you want to run all the applications on your system with their own library operating system. Would you be better off using an Exokernel or hypervisor for this configuration? Explain your answer.

Question #2: Nucleus

The designers of Nucleus rejected semaphores as a basic form of synchronization between cooperating processes.

1. Discuss why semaphores are not a good match for Nucleus.
2. Nucleus instead uses messages as the basic means of process communication. Describe some of the advantages of using messages for synchronization and communication in Nucleus.
3. Nucleus administers a common pool of message buffers. Describe some of the complications this could cause, and how Nucleus manages these difficulties.

Question #3: AutoRAID

RAID systems can be difficult to configure to obtain the best performance and reliability for different workloads. HP AutoRAID attempts to solve this problem by adapting how data is stored.

1. On a read operation, where are the possible locations where the data could be located in AutoRAID? How does AutoRAID determine where the data is located?
2. How does AutoRAID store write-active data? Why is this configuration a good match for write-active data? Briefly describe the steps for a read or write to this data.
3. How does AutoRAID store write-inactive data? Why is this configuration a good match for this data? Briefly describe the steps for a read or write to this data.
4. AutoRAID performs some operations in the background. Briefly describe some of these background operations and why they are needed.

Question #4: Marshalling Complex Objects

In an RPC system, programmers often need to pass complex objects or structures to the remote procedures. These complex objects may contain pointer-based data structures and nested objects or structures. As part of the remote call, the client stub needs to be able to package up these complex objects and send them to the remote procedure. This means that the stub needs to understand what data needs to be sent.

1. What additional information, if any, does an RPC stub generator need that a normal compiler does not need?
2. The description of an RPC interface can be embedded in a program's source code or can be specified as a separate file. Describe how each of these two approaches would work and describe the advantages and disadvantages of each approach.
3. Suppose that you wanted to pass an object that represents a node in a tree. Two possible approaches are to pass just the node object itself, or to pass the node and all the objects to which the node object points. Describe how each of these two approaches would work and describe the advantages and disadvantages of each approach.

(continued ...)

Question #5: Fault Tolerance and Isolation

Isolation is a key component of most fault-tolerant systems.

1. What is isolation, and why is it important for fault tolerance?
2. How does Unix provide isolation?
3. Both Nooks and software-based fault isolation (SFI) provide isolation for a subset of code, namely extensions, within an address space. However, they use significantly different mechanisms. What properties of a system would lead you to choose one approach or the other, and why?
4. Google's chrome browser can run extensions in a separate process, with a version of SFI to further restrict what an extension can do. What kinds of security or reliability problems can be handled with this design that SFI or a separate process alone cannot handle?

Question #6: Dynamo and Chord

Dynamo is a key-value store from Amazon. It contains a number of interesting facets to make it work at scale. In particular, Dynamo uses a variant of consistent hashing (from Chord) to partition and replicate keys and values.

1. First, describe the basic idea of consistent hashing; what problem does consistent hashing solve? How does it work?
2. Some consistent hashing approaches map multiple virtual nodes onto each physical node of the system; what is the purpose of such a mapping?
3. Finally, Dynamo had some problems with a pure implementation of consistent hashing, and instead simplified the approach. One problem they had arose when a new node joined the system. What happens during join in such a system? Why was it a problem for Dynamo?