

Name:

email:

ID:

Final Exam
CS555
12 May 2009

You have 2 hours for this exam. The exam has 7 pages. There are 100 possible points. Show all your work for partial credit.

Definitions

Each question is worth 1 point. Answer all questions in this section.

1. Define the following

a) Strongly Typed Language

Answer: Language that strictly controls operations on types, especially conversions between types.

b) Paravirtualization

Answer: Virtualization that changes interfaces to the virtual hardware that enable use of existing unvirtualized hardware. A Xen innovation.

c) Priority Inheritance (as used by Monitors)

Answer: System that assigns a process holding a monitor lock the same priority as the highest priority process waiting for the lock. Avoids priority inversions.

d) UNIX Streams

Answer: System of stackable modules that control the behavior of UNIX ttys. An early model for extensibility using standard interfaces.

e) User-centric naming (or process-centric naming)

Answer: Plan 9 system that allows a process or user to control the hierarchical file system that is visible to the process.

f) Error Correcting Code (also called a Hamming Code)

Answer: Data encoding that can restore data from errors rather than simply detect changes.

g) CSP rendez-vous

Answer: CSP language construct that combines synchronization and message passing.

h) Read or Write Quorum

Answer: The number of replicas one must acquire in order to read or write (respectively) in a weighted voting system.

i) Allocation Group (XFS)

Answer: Unit of disk sector allocation in XFS used primarily for scaling.

j) Atomic action

Answer: An indivisible operation. Either the action occurs completely or not at all.

Name:

email:

ID:

Short Answer

Each question gives its value in the question. Answer all questions in this section. This section is worth 50 points.

2. Describe a microkernel, including a contrast with a monolithic kernel. (4 points). Explain 2 advantages that microkernel advocates claim for microkernels and one disadvantage. (6 points)

Answer:

A microkernel removes as much functionality from the monolithic kernel as possible to provide a simple portable system abstraction on which different operating system interfaces can be implemented. Such abstractions are implemented in servers outside the microkernel.

Microkernel advocates claim that microkernels are more easily portable to new hardware because the trusted and hardware dependent codebase is small and isolated. They also claim that multiple operating system interfaces to the same hardware can be presented simultaneously by abstracting them into separate servers that address the same microkernel.

One disadvantage of microkernels is the increased overhead of interprocess communication between servers compared to simple procedure calls within a monolithic kernel.

3. We were introduced to the tightly related ideas of *checkpointing* and *rollback* in the beginning of the semester when we studied Virtual Time/Time Warp. Define each of these terms (you may define them in terms of Time Warp if you wish) (4 points). Pick two other systems (not Time Warp/Virtual Time) that uses checkpointing and rollback and describe how the system uses them. (6 points)

Answer: Checkpointing is capturing sufficient state about a running process (or other system object) that the system can restart the process from that state. Rollback is the process of restoring a process or system object to a checkpointed state.

Any process migration system will make use of these two operations to encapsulate process state for the move to another system. Both Condor and Sprite use systems that checkpoint and rollback in different ways to move processes. Also Soft-Updates creates checkpoints of file system structures when writing disk blocks that have had multiple changes to them. These checkpoints are constructed from the current state of the file system and the information in the dependency data structures.

Name:

email:

ID:

4. The Content Addressable Network (CAN), the Intentional Naming System (INS) and the Google File System (GFS) are all overlay systems. They use an existing network or system as direct building block. Pick one of these systems, tell what system it is overlaid on (2 points) and what advantage the system gets from using that building block (3 points).

Answer:

The two network systems (INS and CAN) are built on IP networking and the GFS is built on the Linux file system (chunks are Linux files). In each case the system need not reimplement the lower level functionality – the CAN and INS do not have to reconstruct all the details of message forwarding in an IP stack and the GFS does not need to recreate the low-level management details in Linux's file system.

5. Define capability (2 points). Pick three systems we discussed this semester and briefly describe (one line) how they prevent capability forgery (3 points)

Answer:

A capability is a combination of a reference to an object and the rights that the holder can exercise on the object. The systems we studied are:

- Amoeba: cryptographic forgery protection
- SPIN: type safe language prevents forgery
- Hydra: Capabilities are not allowed outside the kernel, so cannot be forged

Name:

email:

ID:

6. We discussed several papers that conducted performance evaluations of operating systems or their components. Each of these papers found examples where operating system design had a marked influence on performance as captured in a microbenchmark. Pick one of these and explain the operating system design decision (2 points) and how it affected performance (3 points).

Answer:

There are several, of course. For example, NT's microkernel design means that it will have more interprocess communication than monolithic kernel designs, which showed up in several performance metrics in Chen's operating system performance comparisons. Windows choice to support a variety of backward compatibility modes resulted in many changes of processor state that resulted in higher overhead for a variety of system calls from the same study. Similarly, NetBSD did not support graphical interfaces in the kernel, but as a user process, which results in slower performance than NT or Windows which made the opposite decision due to more system calls per rendering.

7. Resource-constrained operating systems often use different approaches to operating systems problems than conventional operating systems. Briefly describe the how synchronization is addressed in Emeralds (2 points) and in TinyOS (3 points)

Answer:

Emeralds presents a fairly conventional interface to semaphores, used for synchronization, though the implementation is tuned to enable optimizations to avoid unnecessary context switches. Status messages provide another mechanism for one to many communication without using standard synchronization systems.

TinyOS's event-based paradigm is organized to minimize the need for explicit synchronization primitives. Handlers and tasks run to completion rather than waiting for one another, and synchronization is largely handled by careful handshaking in the code.

Name:

email:

ID:

8. The designers of Xen mention that one of their goals is to simplify installation and operation of multiple pieces of software on a single machine. One important way this is accomplished is that each virtual machine has its own file system. How does this simplify software configuration? (2 points) Does running in a VM prevent software from sharing data with other software running on the same physical machine through a file system? (1 point) Explain why or why not (2 points)

Answer:

Various software packages may require incompatible configurations of system services through system files, and allowing each package to see a distinct file system can simplify that. Each change is seen in its own file system and affects its own VM. While the systems do not share a file system intrinsically, any network file system could be used to share information. If you talked about the file systems by isolating the VMs, you get the 2 points for the reason, but they could share data through a shared filesystem.

9. Briefly describe each of the RAID disk layouts (one line is enough). (5 points)

Answer:

- Disk mirroring
- Bit-sliced data with an error correction code on the check disks
- Bit sliced data with error isolation provided by hardware (1 check disk)
- Sector sliced data with error isolation provided by hardware (1 check disk)
- Sector sliced data with error isolation provided by hardware and a distributed check disk (1 check disk)

Long answer

This section is worth 40 points. Each question gives its value. Do all questions in this section.

10. We studied several file systems this semester, each of which addressed a different aspect of this system service. For each file system problem below, name a file system that addressed the problem (1 point) and describe an insight that the designers had into the problem (2 points) and a specific technique in the file system design that insight lead to (2 points).

- a) Small disk writes

Answer:

The LFS addressed this problem by writing consistent segments of the file system together as logs rather than distributing the data on the disk in patterns optimized for reading. The insight was that collecting data from multiple reads into groups would allow the OS to issue large contiguous writes. The log was broken into such segments and sized to make optimal use of the disk bandwidth.

Other systems used clustering of writes to address this, including the system underlying Soft-Updated (the Berkeley Fast file system) and XFS, and they are acceptable answers as well.

- b) Synchronous meta-data writes

Answer:

Soft-updates specifically addressed the write speed bottleneck created by synchronous meta-data writes. The fact that file system consistency was maintained by waiting for file system structures to be written to disk rather than to the file cache is a bottleneck. The insight was that as long as asynchronous writes were ordered to present a consistent image of file system state, they needn't be made synchronously. Soft-Updates consists largely of a system to track the dependencies between disk writes and ensure that they are sent to disk in a valid order.

- c) Slow searching of file system structures at scale

Answer:

XFS addressed various issues of scale including the slow access to these structures (such as free lists and directory entries). The insight was that more memory, CPU cycles, and disk space was available to use in managing more complex data structures that had better scaling characteristics. XFS replaces most linear lookups with one or another form of B+ tree, which has lookup times that are guaranteed to be $O(\log(n))$

- d) Reliable access to files in the presence of failures

Answer:

Several systems took on this problem, including Locus, Ficus, Coda, and the Google File System. In each case the insight was that replication increased availability, though each of these had different ideas about the scope and nature of failures being protected against. Locus used protocols to detect and correct for network partition combined with optimistic reconciliation of changed files. Ficus took similar approaches, scaled to a much larger scale by relaxing the consistency semantics. Coda dealt primarily with failures as disconnections, using aggressive caching and optimistic reconciliation. Google uses a simpler replication and centralized control (with the centralized data also replicated).

Name:

email:

ID:

11. Designing a system depends not only on providing a useful design today, but in providing for ways for later researchers or designers to extend the system. These questions discuss various aspects of extensibility.

- a) In the process of creating the Network File System its designers added an extension interface to the Unix file system. Name and briefly describe this interface. (1 point, 4 points)

Answer:

NFS added the virtual file system (VFS) interface to unix, changing i-nodes to v-nodes. The key point of the new system was that rather than tying operations implicitly to kernel routines, the VFS defines the various file system operations abstractly and each file system provides pointers to routines that implement those operations for the particular file system.

- b) Plan 9 services are mapped into a hierarchical name space and accessed through a common interface. Name the interface (2 points) and name the protocol that specifies the interface in detail (2 points). This protocol is an instance of a general distributed programming model; name it (1 point).

Answer:

Plan 9 services look like files – that is they export a file system interface. The specific interface that a service must export is defined by the 9P protocol. 9P is an RPC protocol.

- c) Briefly describe the x-kernel's model of a protocol stack. (5 points)

Answer:

Protocol objects, the static representation of overall protocol data, are interconnected. When a protocol object is asked to initiate or respond to a connection it creates a session object that contains the instance data for that connection. Messages, representing data exchanged between protocols, actively traverse the chain of protocols and sessions. A user process is modeled as a protocol/session as is a device.

- d) The file system interface is a well-understood interface that has been extended for use in particular environments. The Google File system provides many standard file system operations, but is tuned for a set of new operations. Explain one new operation that the Google file system implements (3 points) and explain a new requirement that the GFS imposes on clients (2 points).

Answer:

GFS supports a record-based append operation that adds records at least once to a file. The data may be partially (or totally) repeated in the file. This requires clients to be able to deal with partial or total repeats of data; for example it might assign sequence numbers to identify data already seen and checksum records to detect partial copies.