CS 111 Final exam

Adam Christopher Cole

TOTAL POINTS

68 / 100

QUESTION 1

1 Distributed deadlock 10 / 10

✓ **- 0 pts Correct**

- 10 pts Incorrect/ No Attempt

- 3 pts On the right track, incomplete

- 6 pts Shows some understanding of the concepts, but misunderstood the question

- 1 pts Small error

QUESTION 2

2 Block I/O cache 10 / 10

✓ **- 0 pts Correct**

- 10 pts Incorrect/ no attempt

- 6 pts Answer is too high level

- 3 pts No mention of input devices

- 2 pts You don't cache a device

QUESTION 3

3 LFS and sparse files 0 / 10

- 0 pts Correct

✓ **- 10 pts Incorrect/ no attempt**

- 8 pts Not answering the question

- 3 pts Not mentioning inodes

- 3 pts Vague

- 2 pts Some errors

QUESTION 4

4 RBAC 5 / 10

- 0 pts Correct

- 8 pts Not discussing difference between RBAC and basic access control.

- 0 pts No answer

- 1 pts RBAC does not itself authenticate.

- 2 pts More details on utility.

✓ **- 5 pts Not distinguishing RBAC from group permissions. Also not clarifying utility of multiple roles for one user.**

- 3 pts Key issue is ability for ordinary users to take on different roles with disjoint privileges.

- 4 pts Insufficient description of benefits.

- 3 pts Not clear what's meant by "easy transfer control". Doesn't necessarily express what's crucial to RBAC.

- 8 pts RBAC is not a cross between ACLs and capabilities.

- 9 pts Very little correct information in this answer.

- 9 pts Not a device-oriented mechanism. Not a proper description of critical features of RBAC.

- 10 pts Temporary score - need to get physical copy of test.

QUESTION 5

5 Caching and performance measurement 10 / 10

✓ **- 0 pts Correct**

- 10 pts The issue is unfair performance advantage.

- 9 pts Caching the measurement data isn't the issue. Unfair performance advantage for runs benefiting from the cache is.

- 8 pts Correctness is not going to be an issue with a cache. Performance is.

- 4 pts More details on why one must be concerned.

- 5 pts Main issue is possible unfair performance advantages for some trials.

- 3 pts Need to be more specific about how locality affects measurements

- 10 pts Temporary score - need to get physical copy of test

QUESTION 6

6 Dynamically loadable device drivers 10 / 10

✓ **- 0 pts** Correct

- 1 pts Hot plug issues.

- 4 pts Primary issues related to many possible devices, but few in any given computer.

- 10 pts No answer

- 9 pts Didn't actually answer the question.

- 8 pts Processes don't load device drivers. The OS does. Main benefit is handling vast numbers of possible devices.

- 10 pts Irrelevant answer.

- 3 pts Why does this mandate dynamic loading, though?

- 1 pts Just because others offer it doesn't matter, unless there's a value in offering it.

- 10 pts Temporary score - need to get physical copy of test

QUESTION 7

7 Horizontal scalability 10 / 10

✓ **- 0 pts** Correct

- 10 pts No answer

- 3 pts What is characteristic that allows these benefits?

- 5 pts Why is this beneficial?

- 2 pts Definitely not inodes.

- 4 pts Poor definition of horizontal scalability.

- 8 pts What's the special characteristic of horizontal scalability, as opposed to scalability in general?

- 3 pts There are a lot of ways to add machines to a distributed system. What's particular to horizontal scalability?

- 10 pts Not about clients. About hardware.

- 2 pts Key characteristic is the ability to easily add low power nodes to increase capacity.

- 3 pts Describing map-reduce, not basic horizontal scalability. No need to combine results in that method.

- 10 pts Temporary score - need to get physical copy of test

QUESTION 8

8 Idempotent operations 0 / 10

+ 2 pts Indicate the meaning of idempotent: Idempotent operations produce the same result regardless of how many times it perform

+ 5 pts Indicate the scenario showing why idempotent is useful: Due to fault in the system, when in doubt about the successful execution of an operation, it is always safe to re-execute idempotent until succeed.

+ 3 pts Indicate that due to idempotency, re-execution will not yield a wrong result

✓ **+ 0 pts** Wrong

QUESTION 9

9 Multicore scheduling 4 / 10

+ 5 pts Single queue is easy to implement. Multiple queues are harder to build

+ 5 pts Single queue balance work load well, leading to better fairness. Multiple queues are harder to achieve this goal

+ 5 pts Multiple queues scale better than single queue with respect to increasing number of cores.

+ 5 pts Multiple queues have better cache affinity than single queue

+ 0 pts Wrong

+ 4 Point adjustment

QUESTION 10

10 TLB misses 9 / 10

✓ **+ 1.5 pts 1. Access causes TLB Miss**

✓ **+ 1.5 pts 2. MMU searches PTE in page table**

✓ **+ 2 pts 3. Cause a page fault Exception**

+ 1 pts 3. 1 Indicate OS is involved to handle page fault

✓ **+ 1 pts 3.2 Indicate process is blocked while handling page fault**

✓ **+ 1 pts 4. Page frame loaded from disk to memory**

✓ **+ 1 pts 5. Page table entry X is updated to point to the new page**

✓ **+ 1 pts 6. TLB is updated to reflect changes in PTE**

+ 0 pts Wrong

Final Exam
CS 111, Principles of Operating Systems
Summer 2019

Name: Adam Cole

Student ID Number: [redacted]

This is a closed book, closed note test. Answer all questions.

Each question should be answered in 2-5 sentences. DO NOT simply write everything you remember about the topic of the question. Answer the question that was asked. Extraneous information not related to the answer to the question will not improve your grade and may make it difficult to determine if the pertinent part of your answer is correct. Confine your answers to the space directly below each question. Only text in this space will be graded. No question requires a longer answer than the space provided.

1. A distributed system has N nodes, each of which uses its own correct independent mechanism to guarantee no deadlocks based on locks for just local resources. We will be running cooperating processes on the N nodes that will require both local locks and locks on remote resources, which are acquired by requesting the lock from the remote node controlling the resource. A system designer proposes that we handle multi-machine deadlocks by numbering the nodes 1 through N and requiring processes to obtain all locks from node X before acquiring any locks from node X+n (n>=1). Will this approach prevent the system from deadlocking? Why? (Note that I did not ask you design another mechanism.)

Yes it will. Since each node correctly prevents deadlock internally, we must only ensure deadlock between nodes externally does not happen. By ordering the nodes and requiring the locks to be accessed in order, the design removes potential circular dependency from the system. Without circular dependency, no deadlock can occur.

2. What is the purpose of the block I/O cache? What gets cached there?

The purpose of the block I/o cache is to cache blocks of memory from recent I/O operations from Disk. The purpose of this cache is to help reduce disk operations, since most programs read & write to the same file multiple times. With this design, the block I/o cache can be updated until eventually written back to disk to mirror cache contents of the file. This reduces the number of required disk accesses, making our program more efficient.

3. Is the Log Structured File System (LFS) well designed for handling large sparse files? Why?

Large sparse files are handled well with extra metadata, since storing the offsets & layout require more detailed information. Due to this, log file systems do not handle large sparse files well. Just the metadata alone is often too much data to efficiently store in a log file. In order to do this, extra memory would have to be allocated to the log file. This would hurt the performance of the file system because the extra data would cause longer searches to find normal data blocks. This also raises scalability issues that infer this design is not well suited for sparse files.

4. What is role based access control (RBAC)? Why is it useful in many organizations?

Role based access control is a mechanism of security to allow access to certain resources based on a role within an organization. Many organizations find role-based access control useful because logically, different roles assume different permissions. For example, a manager should be able to access personal information and salary information on his workers, but the workers should not be allowed to see each others. In this case, Role based access control (RBAC) would allow a manager to access a database with this information, but the retail worker would be denied access.

5. When designing a system performance measurement experiment that is not itself concerned with cache measurement, why do we need to be concerned about caching?

Caches affect the performance of a system greatly. If the caches are updated well throughout the program (storing memory w/ spacial and temporal locality) the performance of our system will be much higher. If the caches are invalidated (due to a context switch), this will hurt our performance. Since caches can affect the performance of our system in many different scenarios, tests may report inaccurate conclusions about a seperate part of the system.

6. Why do modern full-function operating systems like MacOS and Windows provide the capability for dynamically loadable device drivers?

Just as with dynamic linking, dynamic loading will load the necessary device driver when necessary. Full-functioning OSes like Mac and Windows need to provide device compatibility will almost all devices. If the user wants to use a device that is not essential / frequently used by the system, the Operating System saves space in kernel memory by not loading a copy of the driver. When it is referenced, the OS will pay a little extra overhead to dynamically load the infrequently used driver.

7. What is meant by horizontal scalability in distributed systems? Why is it beneficial?

Horizontal Scalability refers to the ability to scale by adding resources. In distributed systems, horizontal scalability is achieved by adding servers/machines to the distributed system. It is beneficial because each server linearly scales the processing power of the whole system. Furthermore, if a server/machine fails, it only affects performance by reducing resources by 1 - the rest of the servers still remain operational. This also increases system reliability.

8. Why are idempotent operations particularly useful in building distributed systems?

Idempotent operations are operations that will execute the same code every time without fail. These operations are critical in web browsing and distributed systems because it allows load balancing. If every server in a distributed system runs the same idempotent operations, then the front-end switch can balance the load of requests amongst the servers. This critical ability keeps the performance of the distributed system optimized and efficient.

9. Many modern computers have 2, 4, or even 8 cores, which allows the computer to run that many threads of control in a truly simultaneous manner. The operating system for such computers could maintain a single scheduler queue for all of its cores, or it could maintain a separate scheduler queue for each core. What are the advantages and disadvantages of each approach?

Single Scheduler

→ Advantages
- Priority scheduling accounts for all processes
- multithreading easy since 1 process would need multiple cores

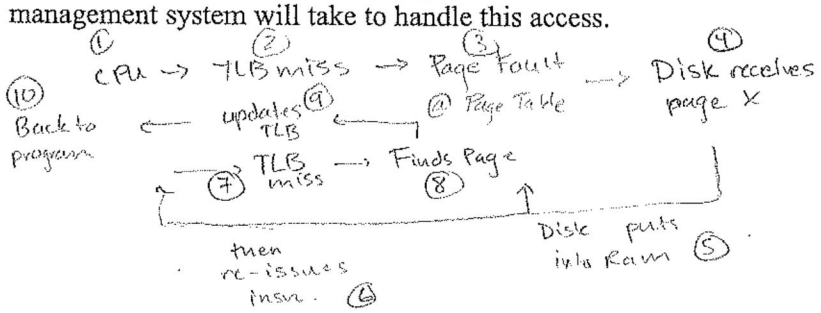
→ Disadvantages
- slower scheduling per core than if each core had a queue of blocked processes

Seperate Scheduler Queues

→ Advantages
- faster scheduling per core

→ Disadvantages
- multithreading must cooperate / preempt other cores' scheduling
- Priority could be corrupted if one core has a lot of high priority processes, and another has lots of low priority processes.

10. A process has a page X that is currently stored on disk, rather than in RAM. The process' page table containing the information describing the location of page X is itself in RAM, but the entry for X is not in the MMU's TLB. The process issues an instruction accessing a memory location Y on page X. Describe the actions that the memory management system will take to handle this access.



The process issues memory request at VM address. The MMU then checks the TLB for an entry and has a TLB miss. The MMU then consults the process's page table, where a page fault occurs. An I/O request is issued to retrieve the memory from disk, and the process blocks. Resource manager then transfers pages into RAM over the bus and updates the page table. From here, the CPU re-issues the request for VM address. Again, there is a TLB miss and the page table is consulted. Here we get a page hit. The TLB is updated to reflect page existence and the CPU regains control, continuing the process.