CS 111 Final exam

LONGERBEAM; ALEX ANDREW

TOTAL POINTS

127 / 150

QUESTION 1

1 Scatter/gather I/O 7 / 10

- **0 pts** Correct
- **10 pts** No answer

✓ **- 3 pts** **Not identifying DMA**

- **3 pts** Not identifying non-contiguity of virtual RAM pages
- **2 pts** not identifying data copying as main issue
- **2 pts** Memory mapped I/O is not a motivation
- **2 pts** Not about accumulating I/O operations.
- **2 pts** Files and inodes not relevant.
- **10 pts** Totally wrong
- **2 pts** Scattering and gathering is over RAM, not I/O device.
- **2 pts** Not related to TLB misses.
- **1 pts** Segments are not necessarily contiguous in physical memory, either.
- **2 pts** Memory mapped I/O != paged virtual memory
- **1 pts** Which mechanisms of a VM system?
- **8 pts** DMA and the paging aspect of VM lead to problems without scatter/gather.
- **2 pts** File system issues irrelevant.
- **4 pts** Scatter/gather typically unrelated to demand paging.
- **2 pts** DMA requires physically contiguous memory.
- **3 pts** Defragmentation has nothing to do with scatter/gather.
- **2 pts** Swapping not relevant.
- **2 pts** Double buffering is irrelevant.
- **3 pts** Poor explanation.
- **2 pts** Fragmentation is not directly related to this issue.
- **9 pts** One tiny bit of correct information
- **1 pts** Internal device memory not relevant.

QUESTION 2

2 Metadata journaling 10 / 10

✓ **- 0 pts** **Correct**

- **10 pts** No answer
- **3 pts** Didn't provide enough discussion about what could happen if we write data blocks after metadata/journal is modified.
- **7 pts** Not very correct.

QUESTION 3

3 URLs and links 7 / 10

- **0 pts** Correct
- **10 pts** No answer
- **4 pts** A URL is more like a soft (symbolic) link
- **3 pts** In both cases, the link is a name describing a traversal through a set of linked data items - files and directories in the case of a soft
  link, web pages in the case of a URL.

✓ **- 3 pts** **There is no guarantee in either case that the data**
item named by the URL or soft link actually exists.

- **10 pts** wrong answer
- **1 pts** mixed the concept of domain and URL
- **1 pts** do not explain how a URL works

QUESTION 4

4 Password salting 10 / 10

✓ **- 0 pts** **Correct**

- **10 pts** No answer
- **3 pts** Did not correctly explain in detail the definition of salt
- **4 pts** Did not correctly discuss in detail preserving password secrecy in the context of hashes
- **3 pts** Did not correctly explain dictionary attacks / brute force attacks

QUESTION 5

5 Factors 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 5 pts A factor is an aspect of the system that you intentionally alter in controlled ways during the evaluation.

- 5 pts Proper choice of factors will allow the experimenter to gain insight into the likely performance outcome of design choices and varying use cases

- 1 pts The reason is not clearly or correctly explained

- 10 pts wrong answer

- 2 pts not proper answer "why"

- 3 pts It's the variables we alter

QUESTION 6

6 File descriptors and capabilities 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 1 pts OS can easily revoke a file descriptor by removing it from the process control block.

- 3 pts Uniqueness not really a property of either capabilities or file descriptors. Important point is that possession grants access.

- 2 pts Important point is mere possession of each grants access.

- 2 pts Capabilities do not necessarily have any "position" information associated.

- 1 pts Users can also access files by opening them via ACL, so FDs alone don't specify their possible available files.

- 7 pts Both capabilities and file descriptors are about access control, not identification and/or authentication.

- 2 pts Changing the ACL does not invalidate existing file descriptors.

- 2 pts File descriptors are R/W specific.

- 3 pts File descriptors tell us nothing about why someone could access a file, merely that they can.

- 8 pts Insufficient detail.

- 5 pts Important point is that both are access control mechanisms providing security based on mere possession of a data structure.

- 1 pts Capabilities usually do not contain a list. Rather, you have a list of capabilities.

- 7 pts How is a FD like a capability?

- 5 pts Misdefinition of capabilities.

QUESTION 7

7 Dining philosophers 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 9 pts Wrong answer.

- 3 pts Needs a better explanation. A good example is when all philosophers call getforks() at the same time and all of them get the left fork.

- 3 pts Partial correct.

QUESTION 8

8 Monitors and synchronized methods 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 4 pts More detail on granularity.

- 2 pts All synchronized methods in an object share one lock.

- 2 pts OO monitors provided by language, not OS.

- 6 pts Monitors lock entire object for any method, synchronized methods only lock on specified methods.

- 6 pts Sync methods more fine grained than object monitors, since the latter locks object on ANY method.

- 10 pts Totally wrong.

- 3 pts Monitors do not prevent inter-object deadlocks.

- 2 pts Monitors lock a class instance, not an entire class.

- 1 pts Java sync methods require identification of the methods. They don't try to determine if the object is modified.

- 3 pts With synchronized methods, non-

synchronized methods can be used in parallel.

- 1 pts Java synchronized methods provide enforced locking.

- 3 pts Object oriented monitors are often provided in the language, and need not be implemented by the programmer.

QUESTION 9

9 Callbacks in AFSv2 0 / 10

- 0 pts Correct

- 10 pts No answer

- 2 pts Callbacks occur when a file is updated, not to check if the cached copy is still OK.

✓ **- 10 pts Not the purpose of an AFS v3 callback. It's for cache consistency.**

- 5 pts Callbacks go from server to caching clients when a file is updated.

- 8 pts More detail required.

- 10 pts AFS is a file system.

- 5 pts Callback is to notify caching client of updates at other sites, not to validate that data has been received.

- 5 pts Why does this have to happen?

- 2 pts Not just for directories.

- 2 pts Why would a file's status change without the client knowing about it?

QUESTION 10

10 PK certificates 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 2 pts Did not mention public key of issuer in certificate.

- 2 pts Did not mention digital signature of trusted 3rd party in certificate

- 2 pts Did not say that a mutually trusted third party is needed to sign the digital signature

- 4 pts Did not correctly say that the trusted 3rd party's public key, which matches the 3rd party's private key used to sign the digital signature, is needed to decrypt the digital signature

QUESTION 11

11 Zombie states 5 / 10

- 0 pts Correct

- 10 pts No answer

- 5 pts A final state indicates that a process has finished executing all of its code. However, it has not yet been cleaned up.

✓ **- 5 pts It allows the parent process to check its exit status and possibly perform other cleanup tasks.**

- 10 pts wrong answer

- 2 pts all of the memory and resources associated with a zombie process are deallocated

- 2 pts The parent process checks the exit status

- 5 pts Parent process waits for child process

QUESTION 12

12 Fairness and scheduling 10 / 10

✓ **- 0 pts Correct**

- 10 pts No answer

- 5 pts Performance is a vague term. What precisely do you mean? Your example is unclear.

- 1 pts Precisely what do you mean by performance here? Fairness itself is one aspect of performance.

- 10 pts That's not a property.

- 5 pts Why is continuity desirable?

- 2 pts Even a fair scheduler would not insist on a blocked process getting an equal time slice.

- 2 pts Need better description of why.

- 3 pts Fairness and preemption aren't the same thing. Unfair algorithms can also use preemption.

- 1 pts You're talking about turnaround time, not response time.

- 2 pts Your description does not say why throughput is damaged.

- 2 pts Disk latency not really relevant here.

- 2 pts That's not throughput. Throughput is the amount of useful work completed in a unit time. You're talking about turnaround time.

QUESTION 13

13 Free list ordering 10 / 10

✓ **- 0 pts Correct**

- **10 pts** No answer
- **8 pts** Incorrect understanding of memory free list.
- **2 pts** Missing details or not a very good explanation for ordering by size.
- **2 pts** Missing details or not a very good explanation for ordering by address.
- **4 pts** Wrong answer for ordering by size.
- **4 pts** Wrong answer for ordering by address.

QUESTION 14

14 Page replacement for looping sequential workloads 10 / 10

✓ **- 0 pts** **Correct**

- **10 pts** No answer
- **3 pts** More specifics on the alternate algorithm.
- **4 pts** Clock algorithms approximate LRU, so they aren't likely to do well.
- **1 pts** How could we know this?
- **5 pts** What other algorithm to use?
- **2 pts** How to practically implement your chosen algorithm?
- **3 pts** How will you do lookahead at the end of the loop area? How can you know?
- **1 pts** How to practically order the pages?
- **3 pts** How to choose which chunks to replace? Bad if you choose the LRU chunks.
- **2 pts** How do you know when you've reached the end of the loop and need to move to the head?
- **5 pts** Problem is vast number of page misses.
- **5 pts** This algorithm is no better than LRU, since it guarantees maximum paging.
- **3 pts** Why would you see constant page replacement?
- **3 pts** Which pages do you designate for swapping?

QUESTION 15

15 Load and stress testing 8 / 10

- **0 pts** Correct
- **10 pts** No answer
- **4 pts** Did not say that load testing measures system performance under particular loads, usually loads that are expected to occur in actual operation
- **4 pts** Did not say that stress testing is used to understand how a system will perform in unusual circumstances.

✓ **- 2 pts** **Did not mention that stress testing is most likely to be used in systems that cannot afford to fail.**

Final Exam
CS 111, Principles of Operating Systems
Winter 2018

Name: Alex Longerbeam

Student ID Number: _____

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. What two mechanisms of a modern memory management system lead to the need for scatter/gather I/O? Why do they do so?

A modern memory management system implements virtual memory and paging. Virtual memory means that the addresses a process deals with (virtual addresses) are not the same as actual physical addresses. Paging builds on that concept in that memory is broken up by pages, and contiguous addresses of pages in virtual memory do not correspond to contiguous physical pages. For certain I/O, contiguous physical memory is required. Because of this, scatter/gather I/O is required to translate between virtual addresses of pages in non-contiguous physical locations, and contiguous physical memory locations for/from I/O.

2. For a journaling file system that only puts metadata in the journal, the data blocks must be written to the storage device before the journal is written to that device. The process requesting the write is informed of its success once the journal is written to the device. Why is this order of operations important?

This safeguards against possible sources of system failure happening during a write. For example, if instead the metadata is first written to the journal, and the system loses power before the actual data blocks can be written, when the OS recovers it will appear as though the write was successful. However when the user tries to access this data, nothing is there in storage. This order of operations also ensures the process only gets a success message once both parts of a data write have occurred. In addition, if the method described in the question is used, and there is a power failure in the middle, we will just be left with storage blocks with no metadata referencing it, which can easily be cleaned up using garbage collection.

3. Does a URL more closely resemble a hard link or a soft (symbolic) link? Why?

A URL more closely resembles a soft (symbolic) link. A hard link in a file system means the directory entry or name-mapping points directly at the file being referenced (for example, referencing the I-node number of that file). An IP Address can be thought of as a hard link, because it points directly at the server or destination. A softlink is a special type of file that tells those who access it to go somewhere else. A URL is like this because it is used to find the actual IP address to go to.

4. What is the benefit of using password salting? Why does it provide this benefit?

Salting a password involves placing a specific set of characters before a password before storing it. Salting is typically used in conjunction with hashing, and when a password is hashed after it has already been salted, this provides an added layer of security. It complicates somehow brute force reversing the hash, and also hides the real password if the unhashed version is obtained.

5. In performance evaluation of systems software, what is a factor? Why is the choice of factors important in such evaluations?

A factor is basically a variable that gets changed during performance evaluations. The choice of factors forms the basis of what type of comparative conclusions one can make from the results of the evaluations. For example, if someone is comparing different file systems, the factor would be the file system in use. If someone was load testing a system, the factor would be the load the system is put under. This choice defines the conclusions that can be made.

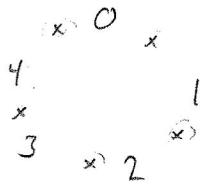
6. In what way is a file descriptor like a capability?

A capability is like a ticket, specifying what objects the holder has access to and what it can do with them. Similarly, a file descriptor held by a process specifies that the process has access to a certain file, and the parameters that the file descriptor was obtained with (read only, write only, read/write, create) specify what that process is allowed to do with the file.

7. Consider the following proposed solution to the Dining Philosophers problem. Every of the five philosophers is assigned a number 0-4, which is known to the philosopher. The philosophers are seating at a circular table. There is one fork between each pair of philosophers, and each fork has its own semaphore, initialized to 1. `int left(p)` returns the identity of the fork to the left of philosopher p, while `int right(p)` returns the identity of the fork to the right of philosopher p. These functions are non-blocking, since they simply identify the desired fork. A philosopher calls `getforks()` to obtain both forks when he wants to eat, and calls `putforks()` to release both forks when he is finished eating, as defined below:

```
void getforks() {
sem_wait(forks[left(p)]);
sem_wait(forks[right(p)]);
}

void putforks() {
sem_post(forks[left(p)]);
sem_post(forks[right(p)]);
}
```



Is this a correct solution to the dining philosophers problem? Explain.

It is not. Let's take the situation where each philosopher simultaneously calls getforks() i.e. They all try to pick up the fork to the left of them at the same time. Since everyone picks up the fork to the left, this first step works fine. However, when they then all try to move on and pick up the fork to the right of them, they all put themselves to sleep waiting because it has already been picked up (the semaphore has already been acquired). We are then left with a deadlock situation, as no philosopher will release the fork until he can pick up the other one, but no philosopher will ever be able to pick up the other one because nobody else will release it.

8. What is the difference between synchronization using object-oriented monitors and synchronization using Java synchronized methods?

Object oriented monitors are special classes where each object has its own lock object. Built into the class is the functionality of acquiring the lock every time any method of that object is run. This is a very conservative approach that ensures accuracy, but can really slow things down. Java synchronized methods are similar in the fact that each object has its own lock object, however only certain class methods specified by the programmer acquire that lock before running. This requires more forethought by the programmer but can reduce unnecessary bottlenecks.

9. What is the purpose of a callback in AFSv2?

A callback allows notification when a request of the file system or made by the file system has completed. This allows the file system, and those waiting on it, to continue to do other things instead of blocking. In addition, it allows easier integration into distributed storage, as a network request handler can continue to process other requests while waiting for a specific callback.

10. Describe how a certificate allows us to securely obtain a public key for some other party. What information, in addition to the certificate itself, must we have to be sure of the certificate's validity? Why?

A certificate guarantees the validity of the public key for a certain entity. It is obtained through a certificate authority, who we inherantly trust to give us the correct public key certificates. However, we must ensure that we are actually receiving the certificate from who we think we are. When a certificate authority gives a certificate to a requester, they sign it using their private key. Therefore, we must know the public key of a certificate authority in order to ensure the validity of a certificate.

11. What is the purpose of a final state (also known as a zombie state) for a process?

When a process is in a zombie state, it is no longer being scheduled, or executing instructions. However, the OS may have not yet reclaimed its resources. Having this final state allows the OS more control over when it can choose to kill the process and reclaim the resources. This also allows any child processes to have still have a valid parent process id if that is something they need, while not having that parent process clogging up scheduling queues (or perhaps owning resources if the OS decides to take resources back from its zombie processes.

12. If we use a scheduler algorithm that optimizes fairness, what other desirable property is likely to be damaged? Why?

If we use an algorithm that optimizes fairness, optimizing for priority or real-time needs of processes is likely to be damaged. This is because the scheduler will make sure every process gets a fair share of the CPU, and gets the same amount of time before being pre-empted. High priority processes will have to wait the same amount of time as low priority processes.

In addition, it is possible that this could reduce throughput, if processes that block for I/O or another request are left on the CPU while blocked.

13. Elements in a memory free list could be ordered by size or could be ordered by their address. What is an advantage of ordering them by size? What is an advantage of ordering them by address?

If elements in a memory free list are ordered by size, a worst-fit algorithm (best for limiting external fragmentation) could be easily implemented, as the worst fit for any desired allocation would always be the largest chunk of free memory, most likely first in the list. If ordered by address, this makes it extremely easy to coalesce contiguous smaller chunks of free memory into larger ones. If they are ordered by address, all it would take to coalesce is to go through, delete headers, manipulate pointers, and change the length of segments as specified by their headers.

14. A looping sequential page workload runs sequentially through a set of pages of some fixed size, cycling back to the first page once it is finished with the last page. Why might an LRU page replacement algorithm handle this workload poorly? What kind of practical page replacement algorithm would handle it better?

An LRU page replacement algorithm will not handle this workload well because at every point, the next page desired also happens to be the least recently used page, and we could therefore frequently see the next desired page being swapped out. A better algorithm for this workload would be most recently used replacement algorithm, as the workload guarantees that the most recently used page is the page that will be needed farthest in the future, which were were taught is the optimal page to swap out.

15. What is the difference between load testing and stress testing? When is stress testing most likely to be used?

Load testing obviously deals more with testing how a system responds to the load it has to deal with (more related to the size and complication of jobs) while stress testing deals more with the number of concurrent jobs or requests being made. Stress testing is more popular for web servers or other systems that are meant to service many clients at once.