CS 111 Final exam

CHEN; ALEXANDER QI-LI

TOTAL POINTS

110 / 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 10 / 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 0 / 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 4 / 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 **6 / 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: Alexander Chen

Student ID Number: 404 837 697

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?

Scatter/gather I/O is a procedure in which data from multiple different locations in memory needs to be "gathered" for a write to an I/O device and data needs to be "scattered" to many different noncontiguous memory locations when reading from an I/O device.

Two mechanisms that lead to the need for scatter/gather I/O are paging and virtual memory. In paging, data is stored in fixed-sized chunks and if a file corresponds to 2 pages, the pages are likely not contiguous in physical memory, making "scatter/gather" necessary for I/O requests. The reasoning is similar for virtual memory. 2 addresses in virtual memory could correspond to very distant addresses in physical memory, making "scatter/gather" necessary.

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?

The data block must be written to the device before the journal so that the metadata stays valid. Writing the data blocks is more likely to fail (since they are larger than the metadata/journal), so writing the metadata afterwards ensures that the metadata is accurate.

The process requesting the write is informed after the journal is written so that the process will only continue once the entire operation is done and no data is corrupted.

The order of operations is important because it provides greater data integrity.

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

A URL more closely resembles a soft link. One reason is that just like a symbolic link, a URL can be a dangling pointer. If the web server hosting the website is down, the URL can still be used, but an error will occur. Also, multiple URLs can point to the same webserver, but if all of the URLs are remored, it does not mean the web server is deleted. Thus, URLs have the dangling pointer property of a soft link, but don't have the file deletion on link_count = 0 property of a hard link.

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

Password salting is adding extra characters to the end of a password before sending it over a network. The benefit of password salting is improved security. This improves security because if there is no salting and passwords are encrypted before being sent over a network, hackers could try to reverse engineer the encryption system through matching common hashes to common passwords. If passwords are salted, this reverse engineering is much harder since the size and contents of the salt are unknown.

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

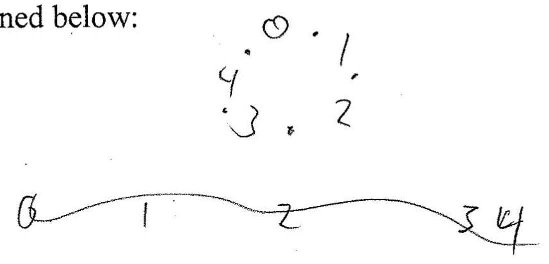
A factor is a certain metric by which you can evaluate the performance of the systems software. For example, common factors include throughput, latency, seek time, etc. Choice of factors is important because they give a standard by which to measure performance and allow performance to be improved.

For example, sequential read/write and random read/write are common factors for hard disk drives. If the user often transfers large files, choosing sequential read/write as a factor would improve performance. If the user often transfers many small files random read/write would be a more important factor.

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

A capability is a security technique that describes which files a certain user is allowed to access and in what ways they are allowed to access it. File descriptors are similar in that a program can only access a file if they have a file descriptor for it. Also file descriptors specify read and write permissions. In short, a capability shows what files a user is allowed to access and file descriptors show what files a program is allowed to access.

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)]);
}
```

dead lock in get forks

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

This is not a correct solution because of dead lock. If each of the 5 philosophers grabs the left fork (run sem_wait(forks[left(p)])), then tries to grab the right fork (run sem_wait(forks[right(p)])), they would not be able to grab the right fork because it is not available (another philosopher already grabbed it). In this situation, each philosopher is holding a fork and waiting for a fork that another philosopher is holding. Since no philosopher will give up their fork until they grab the other, we have deadlock and a correct solution will never be reached.

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

Synchronization using object-oriented monitors is usually much more complicated than Java synchronized methods, but allows for greater customizability and control.

Object-oriented monitors such as locks, semaphores, and condition variables can be very complex, but allow the user to control certain factors such as fine vs. coarse-grained synchronization and other sychronization implementations. This allows object-oriented monitors to be faster when implemented well.
Java synchronized methods hide a lot of the details through abstraction, so the same level of "fine-tuning" cannot be achieved, but it is much simpler to use.

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

A callback allows some code to be run after a certain action has been completed.

In AFSv2, specifically, a callback will rerun a read or write command to an I/O device if it originally failed. This improves data integrity.

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?

If we are trying to obtain a public key, it is possible that an attacker is trying to give us a fake key. A certificate can help us ensure that the public key corresponds to the correct party. The certificate is attached to the key and verifies that it is accurate. We also must know that the certificate comes from a trusted third party such as a cyber security company like Symantec. Certificates can be forged, so knowing the source of the certificate is trusted, ensures the certificate's validity.

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

The final state occurs when the program has completed running, but some cleanup must occur before the process's PCB is removed from the process list. This cleanup largely includes freeing memory that the process was using.

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

Turnaround time is likely to be damaged by optimizing for fairness. This is because fair algorithms cannot make unfair decisions to improve turnaround time. For example, round robin is the optimally fair pre-emptive algorithm. In this algorithm, processes take turns with equal time slices. Because of this the average turnaround time is much longer since processes usually do not run to completion on the first time slice.

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?

One advantage to ordering by size is that a chunk of free memory with the correct size can be found more easily. For example, if using the worst fit algorithm and ordering the free list by decreasing size, the correct block of memory is always at the beginning of the list.

One advantage of ordering by address is that it could improve spatial locality. Ordering by address makes it easier to store related data in addresses that are near each other. This spatial locality can improve runtime due to caching addresses near a memory access location.

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?

LRU could handle this workload poorly if the working set is larger than the amount of memory available, because this would cause thrashing. If the working set is 5 pages, but memory can only fit 4, then each step of the cycle removes the page that will be needed on the next step. In this case LRU is the worst possible algorithm. One page replacement algorithm that would perform better is the random algorithm. Another algorithm is one that removes the page with the lowest spatial locality. (rather than temporal like in LRU).

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

Load testing is testing the performance of a system when it is under constant, heavy workload. Stress testing is testing a system with short bursts of heavy workload. Stress testing is most likely to be used to simulate environments that change quickly and rapidly. For example, a web server may be stress tested to simulate many users going onto a website at once or something like a DDos attack.