CS 111 Final exam

WANG; KEVIN

TOTAL POINTS

134 / 150

QUESTION 1

1 Scatter/gather I/O 5 / 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 9 / 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 8 / 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 **10 / 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

- 2 **Point adjustment**

Did not mention digital signature

QUESTION 11

11 Zombie states **8 / 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: Kevin Wang

Student ID Number: 304783303

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 and gather I/o can be found in any modern system, such as filesystems, that store data in blocks and index them in a multi level index. They appear, since to a process, virtually, the memory looks contiguous, but in reality, the data blocks in physical memory may be fragmented. Scatter refers to the "scattered" writes as a write to a file may update different parts of physical memory. Gather refers to a read, where data blocks from potentially different parts of memory must be brought together under a contiguous portion of virtual memory.

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?

Writing the meta data first without writing the data could result in corrupting that data block on the disk if there is a sudden power loss or other crash while writing the data block. Namely, the OS would think that the write succeeded since the meta data was written to the journal. Writing the metadata afterward fixes this issue since the OS views the write to disk as unsuccessful since the metadata would not have finished writing to the journal.

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

A URL is more similar to a symbolic link. A hard link is a direct reference to an inode of a particular file. A soft link is merely a pathname of some file in the filesystem. Just like a soft link, a URL can be dangling, and point to a resource that no longer or never existed. The hard link would be more closely resembled by an IP address in this analogy, a direct address of some server, unlike URLs and softlinks which are merely "namespaces".

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

Passwords for authentication are stored by hashing them, producing a value where one could not easily deduce the input (password). Yet most of the time, systems use identical hash functions for this purpose, leading to the issue where common password hashes are easily known. Also users with identical passwords will have identical hashes. Thus passwords should be salted: a small random value appended to every password to produce different hashes, even with the same original password. This protects against dictionary attacks and rainbow tables if the password hashes are compromised or stolen.

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 particular variable with multiple levels that affect the performance of systems software. This could be the operating system used, cpu architecture, instruction set, HDD vs SSD, size of input, etc. The choice of factors is important since not controlling them could lead to unwanted variance in the test results. Furthermore, software could be performant, but in a different environment or usage scenario than that of the test. Thus factors should be picked that closely resemble the usage scenario.

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

A capability is a series of (unforgeable and unguessable) bytes that give a certain user or process authorization to perform a certain task or access a certain resource. This is similar to a system level file descriptor, of which the operating system assures that any process that attempts to access that file must be permitted to do so. Otherwise, the process should not have been able to guess the file descriptor.

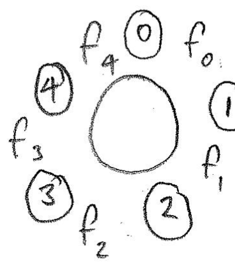
7. Consider the following proposed solution to the Dining Philosophers problem. Every of the five philosophers is assigned a number 0-4, which is known to the philosopher. The philosophers are seating at a circular table. There is one fork between each pair of philosophers, and each fork has its own 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.

The solution may be technically "correct", but it has the potential to deadlock, as each philosopher could pick up their left fork and no one can pick up the right fork. A solution to this could involve removing the circular wait condition for a deadlock by asking each philosopher to acquire an odd numbered fork then an even one.

e.g.



e.g. f_1, f_3, should be acquired first

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

Object oriented monitors ensure that only one thread can access a particular resource at any given time. Java synchronized methods ensure that only one thread can be executing the method at any one time. O.O. monitors are more efficient as the resource is locked for as long as it is used while sync. methods lock the method for the entire duration of the method. Sync. methods also only synchronize the method call in question and do not necessarily lock down any resource that that method may access.

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

The callback was introduced since the incessant polling for updates in AFSv1 lead to the server cpu to bottleneck as it had to handle requests for file update polls. The callback allowed the server to notify clients whether a file changes thus eliminating polling and reducing network overhead. This did introduce some state on the server, however as v1 was completely stateless.

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 is a signed public key by a certificate authority or other trusted entity that ensures that the public key in the certificate has not been tampered with and is legitimate. This requires that the user already know what the public key of the CA or trusted entity is, as any entity could generate a public and private key and sign certificates, including malicious ones.

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

The final state of a process allows the OS to view whether the process completed successfully or whether it errored such as seg faulting. This is important since other processes may be dependent on the successful execution of this process. Also, the final state contains any opened resources by the process, such as file descriptors, that must be closed and cleaned up by the OS.

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

Fairness will sacrifice throughput as giving each process a fair share of cpu time will require preempting the currently running process. That requires a context switch which has its own associated overhead. Fairness optimal may also hinder throughput by giving a process cpu time that does not currently need it, such as a process waiting on I/O.

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?

Ordering by size is useful in a worst fit allocation algorithm which would allow quick access of the largest unallocated block which reduces external fragmentation when allocated. Ordering by address is advantageous in minimizing overhead when maintaining the free list as freed blocks can be easily coalesced into larger ones as they are adjacent.

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?

A LRU might handle this poorly as the pages that are least recently used will be used again prior to the most recently used in this scenario. Evicting the least recently used pages thus will decrease performance as those pages will have to be fetched out of memory again. Surprisingly, a most recently used eviction scheme might work better here as the currently used page will not be required until the next cycle thus increasing the chances of a page hit.

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

Load testing a system involves testing a ——————

system, whether that be under an artificial, random, or pre-saved realistic workload to determine what the performance characteristics of that system are. Stress testing involves testing particular parts of a system to determine where bottlenecks lie. Stress testing is most likely used to debug an underperforming system under particular workloads to see what parts of a system are bottlenecks.