CS 111 Final exam

CHU; EDWARD (EDWARD TSUN MAN)

TOTAL POINTS

140 / 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

"additional layer of cryptography" is inaccurate here.

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 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 9 / 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

- 1 Point adjustment

Did not mention decrypting 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 8 / 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: Edward Chu

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

Paging

Paging and virtual memory. Paging divides a process' address space up into smaller, fixed sized pages, and virtual memory makes the address space appear contiguous to the process, but instead makes the virtual addresses correspond to non-continuous locations in physical memory.
Thus, since the location of pages for a process is non-contiguous in physical memory, we need scatter/gather I/O to properly scatter writes to the different physical location of pages in memory, or to gather up the pages from different locations to properly read/write to them.

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 is because if the order is reversed, i.e. the metadata is written to the journal, then the data blocks are written to the storage device, there is the possibility of the system crashing between the two operations. This will lead to the journal showing that the write is successful (metadata is updated), but the write itself is not actually committed to the device. This results in a corruption to the file system.
Thus, we need to make sure that the write is already committed to the device, before writing the metadata to the journal, which shows the updated changes to node/free node bitmap etc.. We can then inform the process that the write is actually successful, as the data has been written to the storage device and journaling is completed.

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

A soft link. A hard link points directly to an inode, while a soft link only contains the user-readable path to another hard link. Hard links increments the link count on an inode, while soft links does not. All hard links to an inode must be removed before an inode can be deleted, while the number of soft links does not matter. A URL resembles more of a path to a server, and the server can be removed without the URL to it being removed (404 error). Thus, URLs behave more like a soft link rather than a hard link.

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

Password salting increases the security of a system, as even if someone steals all of the hashes for passwords for users in a system, they cannot use the salted hashes to obtain the plaintext passwords of users. If the passwords are not salted, the hackers can brute force the hashes and obtain the passwords in plaintext, as they likely know your hashing algorithm, or there is only so many common hashing algorithms they can try, but they don't. If the passwords are salted, it introduces an additional layer of cryptography, and even if they have the salted hashes and your hashing algorithm, it is very difficult for them to brute-force the salted hashes and obtain them in plaintext.

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 parameter that you can change in a performance evaluation of a system. Choosing the right factors can help us achieve the objective of the evaluation, such as understanding the performance of the system under a live load or a heavy load. We would be able to obtain more accurate data using the correct factors, and thus a better understanding of how the system behaves under different conditions. We can use the data to make adjustments and improvements to the system. Thus, choosing the correct factors can help us evaluate the system better under different circumstances, which can help us modify and improve the system, thus better achieve the objective of the system.

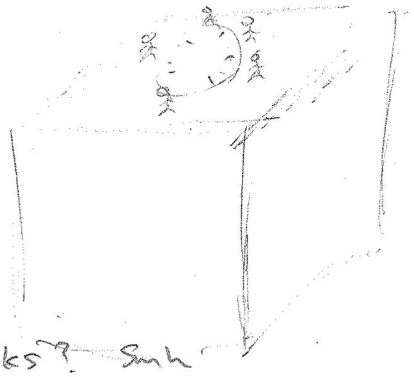
6. In what way is a file descriptor like a capability?

Capabilities are a method of access control, and presenting a capability (ie. a set of bits) to the OS will allow you to do operations associated with the capability. A file descriptor is presented to a process when we do an open() call from the process, and the OS checks that we are allowed to read/write/operate on the file. Afterwards, we can just present the file descriptor to read/write/operate on the file, and the OS will not ask for anything else for access control or check. Thus, since we only need to present the file descriptor to operate on the file, the file descriptor is like a capability.

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)]);
}
```



Why do you need 2 forks? Smh

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

No. This is because there is a possibility for a deadlock in this system. The deadlock happens when all 5 philosophers calls getforks() at a similar time, and after the first line is completed for philosopher 0, there is a context switch to philosopher 1. This is repeated for all philosophers (ie they all get their left fork, then context switch to the next philosopher) Now, none of the philosophers can make progress, as they all are blocked on the second line (i.e. waiting for right fork). Since all philosophers are blocked, there is no chance for any of them to call putforks() and release their left fork, resulting in a deadlock.

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

Both monitors and Java synchronized methods have a lock associated with an object. For monitors, the lock is obtained when any of the methods in the object is called, while for Java synchronized methods, the lock is only obtained if any of the synchronized methods are called. Thus, monitors uses a more rigorous locking mechanism, as it enforces locking, thus guaranteeing correctness, but sacrifices performance. Java synchronized methods are not locking-enforced, but relies on the programmer to know which methods needs to be locked, thus achieving better performance if correctly implemented, as it will not hold the lock for non-synchronized methods.

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

The Andrew File System uses a local cache to speed up reads and writes to a file in a distributed file system. Instead of reading and writing to the remote copy of a file, the whole file is copied to a local location. A callback is then placed on the file, which allows the server to alert the local machine if the remote file has been modified (replacing the need for machines to call getattr() repeatedly to file server). The local machine now knows that the cached version of the file is inconsistent with the version on the remote file system, and can fetch the new version or overwrite the new version on the remote server. push and

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 helps us authenticate the identity of the owner of the public key is indeed who they are, and not someone else pretending to be them. Certificates are unique for each party, and are generally issued by a trusted third party, like a certificate issuing agency. Since they are unique for each party, we can be sure of the identity of a party if we receive the correct certificate from them. However, there is a risk that the certificate may be tampered with in the transmission when we receive them initially. Thus, we need a known public key for the third party to transfer them securely, to make sure that the certificate is not tampered with in the network.

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

Scheduler / Garbage collection

When a process is done, it is not in the ready or running state, as it has already finished running, and it is not in the blocked state, as it is not waiting on response for a blocking operation. The final state tells the scheduler not to put the process into the schedule, and tells the parent process that it is ready for clean up. The final state is useful for garbage collection too, if the system uses it, as it shows which processes are ready for clean up.

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

The performance/throughput of the system is likely to be damaged. This is because by emphasizing fairness, we are making sure that all processes get a equally distributed timeslice, and that all process gets equal CPU time.

However, not all processes require the same amount of timeslice, such as process that does a lot of I/O would require a shorter timeslice, since they're blocked on I/O a lot of times, or processes that does computationally intensive operations would require a longer time slice to finish their operations. By enforcing fairness, it is difficult for processes to achieve their optimal time slice length, resulting in a drop in performance.

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?

The advantage of ordering them by size is that if we need a large piece of free memory, it is easy to find the element, as we can just pick an element from the top of the list, thus it is much faster in terms of performance if for requests for large pieces of memory.

The advantage of ordering by address is that it will be easy to coalesce free neighboring elements together into a bigger free element, thus reducing external fragmentation. When we return an element to the free list, we can simply check the element in front of it and behind it, and if they are neighboring elements, we can coalesce them together into a larger free block.

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?

This is because for a LRU page replacement algorithm, it assumes that recently used pages are likely to be used (temporal locality) again in the future, thus it replaces pages that are least recently used — i.e the page last used furthest in the past. However, with this workload, the page 'last used' is actually the page that will be used furthest into the future, as the workload will complete a whole cycle before the page is used again. A LIFO (last in, first out) page replacement algorithm (i.e. most recently used page replacement) would handle this better, as we're sure with this workload, the page most recently used will be used furthest into the future, so we throw away this one first.

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

Load testing is when we apply a load to system, for example, a simulated load that emulates user behavior, and records the result of the performance of the system under the load (latency, throughput, memory usage etc.)

Stress testing is when we scale up the load on the system gradually, and recording the behavior and performance of the system throughout the increase in the load.

Stress testing is most likely going to be used when we want to figure out the maximum capacity of the system — i.e. how much load it can bear while still functioning.
It is also useful for knowing what adjustments are needed for the system to function under a large load.