CS 111 Final exam

SARKAR; NATASHA

TOTAL POINTS

143 / 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 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

The certificate is not decrypted, the digital signature is decrypted.

QUESTION 11

11 Zombie states 10 / 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 10 / 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: Natasha Sarkar

Student ID Number: 904743795

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?

Modern memory management uses a paging system so that pages of a file may be scattered all throughout memory. Modern memory management also uses virtual memory, so the different pages of a file may not be near each other. Thus, when doing I/O with a device through a bus, to read a file from memory, all the pages need to be gathered so they can be transported via the bus in one contiguous transfer, and to write to memory, the pages are sent in one contiguous block and then scattered throughout 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?

If the data blocks are <u>not</u> written to the storage device before the journal is written, then it is possible that there is a crash after the journal is written but before the data has been written. In this case, when rebooting, the process will look at the journal and update the inode/inode map and other metadata as if the write has been successful, but the data hadn't actually been written yet & could point to garbage.

If the process is informed of the write's success before the journal is written to the device, then if there is a crash prior to the journaling being done, the process believes that the write has been successful even though it hasn't; the metadata contained in the journal haven't been written, so the contents of the metadata are lost.

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

A URL more closely resembles a soft link, because it is more like an indirect reference to the data (a soft link), as it stores the information about the path to the data rather than containing the data itself.

A soft link is an indirect reference to a file, where as a hard link is a direct reference to a file and refers to the file directly by its inode. A URL is more like the former.

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

Password salting makes it more difficult for attackers to keep a list of hashes of commonly used passwords, and compare it to the hashes of passwords of users logging into the system.

A salt is a random sequence added to the end of the password, and this whole thing is then put through a cryptographic hash, which the system authenticating the user checks. Without the salt, someone who obtains the hashed version of the password could try to find it among a list of hashes of commonly used passwords, and figure out the password from there.

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

A factor is some variable element of the system/software that can be changed from experiment to experiment so that evaluators can see the effects of that factor on the performance of the system. The choice of factor is important to help understand performance problems related to that specific factor and see what level of that factor produces desirable outcomes. A poorly chosen factor may result in meaningless results from experiments.

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

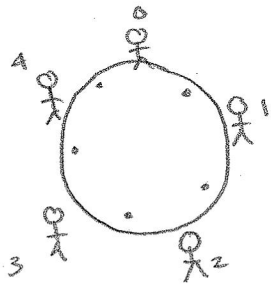
A capability is a permission associated with a process to access specific resources. If a process has a read-access capability for a file, for example, the process is allowed to read that file.

Similarly, file descriptors allow processes to access a specific resource; in this case that resource is a file. Along with a file descriptor comes certain things the process is allowed to do, like read or write to the file. A capability acts in very much a similar way, granting specific accesses of a resource to the process.

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.

No, this is not a correct solution to the dining philosophers problem. It will lead to deadlock in the case that every philosopher obtains his left fork. Now, all the forks are being held and all philosophers are waiting for their right fork to be released, which can never happen.

A correct solution would be if philosopher's 0-3 tried to get their left fork first and then their right fork, while philosopher 4 tried to get his right fork first and then his left fork. This way, at least one philosopher will be able to make progress.

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

For a monitor, every method associated with the object will be synchronized, i.e. it will obtain the lock before performing the method and release it afterwards.

For a Java synchronized method, the programmer explicitly declares each method that they want synchronized with the synchronized keyword. This way, only methods that require sychronization will obtain/release locks, but it is more difficult on the programmer's part as they have to be able to correctly determine which methods to sychronize.

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

A callback is sent from the server to all the clients to let clients know that a file has been updated. This way, if any of the clients have a cached version of the file, the client knows that this is an old, outdated version of the file and that they will need to retrieve the updated version from the server.

This was a solution to the problem in AFSv1 where every time a client wanted to access a file in its cache, it would have to ask the server if the file had been updated; the server was receiving a lot of these messages, so callbacks were introduced in AFSv2.

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 created by a trusted third party. It contains a public key and a signature of the third party. This information is put together and hashed using a cryptographic hash. All of this information is put into a certificate and encrypted using the third party's private key and sent to a user. The user then remakes the hash that should be in the certificate, and then decrypts the certificate using the third party's public key and checks that the hashes are the same.

In order for this to work, the user must have the third party's public key in order to decrypt the message and authenticate it, and thus must have gotten the third party's key from a trusted source.

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

If a process dies, it may have some information (such as its return code) that its parent or other processes may find useful. Thus, when a process dies, it remains in a zombie state, holding this potentially useful information until someone comes to retrieve it, after which the process can be 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?

The desirable property that is likely to be damaged is throughput, as a scheduler algorithm that optimizes fairness tries to make sure all processes get their fair share of the CPU, resulting in many context switches which come with a lot of overhead.

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 a free list by size makes it easier to find free chunks of data that are close to the size of the data requested (for a best fit algorithm) or as large as possible (for a worst fit algorithm).

Organizing a free list by address makes it easier to coalesce adjacent free blocks; a free block can more easily check its neighbors in the free list to see if they are the next/previous addresses of data, and if they are then they can coalesce.

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 page replacement algorithm would work poorly because an LRU algorithm swaps the least recently used page out of memory to disk, but in this case, the next page to be used is the least recently used one, so every time there is a new page, it must be fetched from disk.

A page replacement algorithm that would handle it better is most recently used; the most recently used page is the page that will be accessed the furthest into the future, so it is the best algorithm for this particular scenario.

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

Load testing means adding more work to the system, adding more requests that simulate realistic situations to see how a system responds to a larger workload.

Stress testing means adding more requests to the system that could cause it to fail. It means taking scenarios that may happen once or twice a year, and feeding them to the system at a rate of potentially hundreds of times per minute, to see how the system responds under this stress.

Stress testing is more likely to be used when there are a lot of scenarios that could potentially cause the system to fail, or when it is absolutely vital that the system does not fail under any circumstances.