University of Maryland
CM SC 412, Spring 2024
Professor Pete Keleher
May 6, 2024

Name: ______________________________________

DirectoryID:

Mid-Term Exam 2: The One With File Systems

⊕ ***Show your work or you may not be considered for partial credit.***

⊕ *We will not grade the back of the sheets.*

⊕ ***Avoid asking questions*** *- If something is unclear, document your assumptions and move on.*

⊕ *Do not forget to write your name on the first page. Initial each subsequent page.*

⊕ *Be* **neat** *and* **precise**. *We will not grade answers we cannot read.*

⊕ *If you have written something incorrect along with the correct answer, you should* **not** *expect to get all the points. I will grade based upon what you* **wrote**, *not what you* **meant**.

⊕ *You should draw simple figures if you think it will make your answers clearer.*

☐ *means "choose many"*

◯ *means "choose one"*

If a question has a big empty box, ***put your final answer, including units, in it.***

1. (9 pts) Disk performance

   Define a disk w/ four 30,000 rpm platters (yes, this is fast), 2^{16} tracks on each, and 1000 4k-sectors per track. Assume a seek time of 3 msec.

   1.1. What is the total size of the disk? You may phrase your answer as a sum power of two (e.g. "2^{13}").

   answer: $2^2 * 2^{16} * 2^{12} * 1000 = 2^{40} =$ 1000 GB or about 1TB

   1.2. What is the cost of a random 4k-byte read? (in msec)?

   answer: 30000 rotations/minute $\Rightarrow$ 500 rotations/sec $\Rightarrow$ one revolution is 2 msec, half rotation is 1 msec.
   seek + rot + transfer $= 3 + 1 +$ (2ms / 1000) $=$ 4.002 msec per rotation.

   1.3. Max read bandwidth (this answer may be approximate)?

   answer:
   Track is (1000 * 4k), or about 4 MB (2^{22}) bytes
   about 512 (2^9) rotations/sec
   4 (2^2) platters
   so about $2^{22} \times 2^9 \times 2^2 = 2^{33}$ bytes, or 8GB, per sec

2. (12 pts) Disk scheduling Assume a single-platter hard drive with track 0 the outermost track (sectors 0-99) through track 9 (sectors 900-999).

   The read queue contains requests for sectors 115, 212, 654, 962. If the read head is currently over track 7 moving towards the innermost track, give the total number of additional tracks traversed to satisfy all requests, for each of the following disciplines.

   Count all tracks, whether the head is reading or not. For the scan approaches, assume the head does not need to go all the way to the outside and inside tracks unless required by the requests.

   answer: this is really just tracks 1, 2, 6, and 9, starting at track 7.

   2.1. SSTF

      answer: 6, 9, 2, 1 $\Rightarrow 1 + 3 + 7 + 1 = 12$

   2.2. CSCAN

      answer: 9, 1, 2, 6$\Rightarrow 2 + 8 + 1 + 4 = 15$

   2.3. SCAN

      answer: 9, 6, 2, 1$\Rightarrow 2 + 3 + 4 + 1 = 10$

   2.4. Despite the performance, why list at least two reasons why CSCAN might be preferred over the others.

      answer: It avoids starvation, and is fair.

3. (10 pts) RAID

3.1. Assume a RAID enclosure with six disks, each capable of 1GB/sec sustained read or write performance and holding 1TB, and with no faults. For each of type of RAID, give max read bandwidth, max write bandwidth (except RAID 5), and usable capacity. Assume no failures.

3.1.1. (2 pts) RAID 0

answer: read 6 GB, write: 6 GB, capacity: 6 TB

3.1.2. (2 pts) RAID 1

answer: read 6 GB, write: 3 GB, capacity: 3 TB

3.1.3. (1 pts) RAID 5

answer: read 6 GB, capacity: 5 TB

3.1.4. (5 pts) For RAID 5, list all accesses and computations needed to write a modified version of block i (call it i') from the buffer cache to the disk:

answer: read i, read parity P, $P' = P \oplus i \oplus i'$, write P', write i'

4. (9 pts) FFS

4.1. What is in a cylinder group?

answer: superblock, inodes and datablocks from same dir, bitmap

4.2. What performance advantages does the above organization give?

answer: Grouping block and nodes of file, and multiple files in a dir, together means seeks are much shorter, given any locality in the workload

4.3. List, in order, a valid sequence of writes that could occur when creating a 100-byte file `bar.c` in the directory `/foo` for a generic, non-journaling FFS file system. Indicate which writes are asynchronous.

Note that there are multiple acceptable answers, list only one.

answer:
data-block bitmap (async)
bar.c data (async)
bar.c inode
inode bitmap (async)
/foo data
/foo inode

5. (9 pts) Metadata Journaling

5.1. How are partial transactions distinguished from complete transactions during recovery?

answer: If TxE there, it's good.

5.2. Does a complete transaction in the log during recovery guarantee that relevant data blocks have been updated? Explain.

answer: TxE is delayed until the data is written, so if the transaction is complete (we see the TxE), it's good.

5.3. What happens if the data block write is initiated, the system crashes, and TxB never written. Is the system in an inconsistent state? Explain.

answer: If the TxB (and therefore the TxE) is not present, but the data write was initiated, the file system metadata **is** still consistent. **However:** data has been modified even though the metadata does not reflect it. .

6. (10 pts) SSDs

Assume reading a page takes 1 usec, writing a page takes 10 usec, and erasing a block requires 1 msec. Our SSD initially is as follows:

Block:	0				1				2			
Page:	00	01	02	03	04	05	06	07	08	09	10	11
Content:												
State:	i	i	i	i	i	i	i	i	i	i	i	i

6.1. (2 pts) If we write logical blocks 100, 101, 102, 103, and 104 to pages 4, 5, 6, 7, and 8, how long will it take?

answer: 2 erases, 5 programs $= 2 \times 1 + 5 * .01 = 2.05$ msecs

6.2. (2 pts) Give the contents of a simple mapping table after the above writes:

answer: $100 \rightarrow 4$, $101 \rightarrow 5$, $102 \rightarrow 6$, $103 \rightarrow 7$, $104 \rightarrow 8$

6.3. (3 pts) Do the same assuming a hybrid mapping:

answer:
Log table: $104 \rightarrow 8$
Data Table: $25 \rightarrow 4$

6.4. (3 pts) And if we then write a new version of logical block 101 to page 0?

answer:
Log table: $101 \rightarrow 0$, $104 \rightarrow 8$
Data Table: $25 \rightarrow 4$

7. (5 pts) The End-to-End Argument (EtE)

   The original EtE paper used the above figure to motivate EtE. A file transfer between two computers could have errors in the (1) network, (2) intermediate devices like a router, and (3) traversing the software stack on the computers at either end. The authors argue that the entirety of these errors can only be checked at the endpoint on the destination, so a reliable transport protocol (which does integrity checking, retries, duplicate elimination, etc.) like TCP is not useful.

   So why do application programmers universally use TCP? List at least two reasons.

   answer: (1) TCP catches the majority of errors (the end-system error rate is very low), and (2) implementing reliable transfer protocols is hard and expensive, (3) TCP can be avoided.

8. (4 pts) Google File System

   8.1. How do GFS coordinators maintain information about chunkserver chunk assignments, and how does this affect recovery?

   answer: In memory. During recovery, coordinators poll all chunkservers to reconstruct mapping of chunks to chunkservers.

   8.2. Give the sequence of operations (requests, replies) that occur when a client opens 'foo.c'.

   answer: client asks coordinator for mapping, client asks chunkserver for data.

9. (10 pts) LFS

9.1. (8 pts) Let's say we want a write cost (write amplification) of $\underline{8}$ at most. What max utilization, u, can we tolerate?

answer:
Let x be our write cost.
$x = \frac{2}{1-u}$ *// the original write cost equation*
$x(1-u) = 2$
$1 - u = \frac{2}{x}$
$u = 1 - \frac{2}{x} = 1 - \frac{2}{8} = 0.75 \Rightarrow \underline{75\%}$

9.2. (2 pts) What types of accesses is LFS primarily targeting?

answer: writes and creates of small files

10. (10 pts) NFS / AFS

10.1. How do NFS and AFS differ in their attempts to address the update visiblity problem?

answer:
Both push writes to server by the close. However, NFS might push before then, whereas AFS does not.

10.2. How do NFS and AFS differ in their attempts to address the stale cache problem?

answer:
NFS only requires a check (sometimes) when opening the file. It never notices a cached file is stale while it is open.

AFS uses callbacks to explicitly inform remote clients that their versions are stale.

11. (12 pts) AFS and NFS

Assume:

- We have five unique blocks of data: "A", "B", "C", "D", and "Z", each 4k in length.
- "`write(F, C, 8k)`" means "write the entire block C starting at offset 8k of file F."
- File "F" is initialized to "Z,Z,Z,Z", i.e. it's a 16k file w/ four copies of Z.

C1	C2
open(F)	
write(F, A, 0)	
	open(F)
write(F, B, 4k)	
	write(F, D, 4k)
write(F, C, 8k)	
close(F)	
	close(F)

Which of NFS and AFS could result in the following final contents of "F"?

11.1. `A, B, C, Z`:
- ○ AFS only
- [✓ NFS only]
- ○ both
- ○ neither

11.2. `A, D, C, Z`:
- ○ AFS only
- [✓ NFS only]
- ○ both
- ○ neither

11.3. `A, D, Z, Z`:
- ○ AFS only
- ○ NFS only
- ○ both
- [✓ neither]

11.4. `Z, D, Z, Z`:
- [✓ AFS only]
- ○ NFS only
- ○ both
- ○ neither