University of California, Berkeley
College of Engineering
Computer Science Division – EECS

Spring 2022 Joseph & Kubiatowicz

Midterm III

April 28th, 2022

CS162: Operating Systems and Systems Programming

Your Name:	
SID AND Autograder Login (e.g., student042):	
TA Name:	
Discussion Section Time:	

General Information:

This is a **closed book** exam. You are allowed 3 pages of notes (both sides). You may not use a calculator. You have 110 minutes to complete as much of the exam as possible. Make sure to read all of the questions first, as some of the questions are substantially more time consuming.

WRITE ALL OF YOUR ANSWERS IN YOUR ANSWER BOOK, NOT IN THIS BOOKLET! Make your answers as concise as possible. On programming questions, we will be looking for performance as well as correctness, so think through your answers carefully. If there is something about the questions that you believe is open to interpretation, please ask us about it!

Problem	Possible	Score
1	20	
2	18	
3	29	
4	16	
5	17	
Total	100	

[This page left for π]

3.14159265358979323846264338327950288419716939937510582097494459230781640628620899

Problem 1: True/False and Why [20 pts]

Please *EXPLAIN* your answer in TWO SENTENCES OR LESS (Answers longer than this may not get credit!). Also, answers without an explanation *GET NO CREDIT.*

Problem 1a[2pts]: Direct Memory Access (DMA) is "direct" because it accesses devices via memory directly rather than processor instructions.

☐ True ☐ False

Explain:

Problem 1b[2pts]: If we care about the performance of random accesses to files, it would be better to have a file system with extent-based allocations as opposed to one with linked block-based allocations.

☐ True ☐ False

Explain:

Problem 1c[2pts]: For Project 3, `read`, `write`, and `remove` syscalls should not be supported on directories since there are dedicated directory syscalls `readdir`, `mkdir`, and `rmdir`.

☐ True ☐ False

Explain:

Problem 1d[2pts]: For Project 3, a `struct inode_disk` filled fully with indirect pointers would be a valid approach to support the maximum file size of 8 MiB.

☐ True ☐ False

Explain:

Problem 1e[2pts]: A solution to the General's Paradox is sending large numbers (100's) of messengers in each direction to guarantee a messenger gets through to the other side.

☐ True ☐ False

Explain:

Problem 1f[2pts]: If our file system is prone to short-lived files, we should implement a write-back cache as opposed to a write-through cache.

☐ True ☐ False

Explain:

Problem 1g[2pts]: The File Allocation Table and Fast File System can both support soft and hard links.

☐ True ☐ False

Explain:

Problem 1h[2pts]: In most systems, waiting time scales linearly with respect to utilization.

☐ True ☐ False

Explain:

Problem 1i[2pts]: Just like any other I/O, NFS servers require invocations of `open` and `close` syscalls.

☐ True ☐ False

Explain:

Problem 1j[2pts]: For Project 3, each process is created from another process, meaning all processes are children of the very first process. Since the very first process is initialized with the root directory as its current working directory (CWD), processes can't have a CWD besides the root directory.

☐ True ☐ False

Explain:

Problem 2: Multiple Choice [18pts]

Choose *all* that apply.

Problem 2a[2pts]: For which situations would you require synchronization to resolve race conditions in Project 3? Only consider synchronization at that abstraction level, not below.

A: Simultaneous resizing of different inodes.

B: Changing metadata fields in the inode.

C: Adding directory entries to a directory.

D: Calling `file_deny_write` on a file.

E: None of the above.

Problem 2b[2pts]: What are some advantages of SSDs over HDDs?

A: Random reads are faster.

B: Higher durability over many write operations.

C: Cost per byte of memory is lower.

D: No moving parts, so more physically durable.

E: None of the above.

Problem 2c[2pts]: Which of the following could be used in HDDs to improve performance?

A: Increase disk rotation speed.

B: Internal caching.

C: Track skewing.

D: Internal scheduling algorithms such as a simple elevator algorithm.

E: None of the above.

Problem 2d[2pts]: Which of the following are true regarding device access schemes?

A: Port-mapped I/O memory-mapped I/O, and DMA are all forms of programmed I/O.

B: Memory-mapped I/O uses standard memory instructions such as load and store.

C: Memory-mapped I/O uses a distinct memory space separate from physical memory.

D: DMA maps control registers and display memory directly to physical memory.

E: None of the above.

Problem 2e[2pts]: Which of the following are true about the ACID properties for transactions?

A: Atomicity means that each action within the transaction must be atomic.

B: Consistency means that data must be consistent across all disks even with RAID.

C: Isolation means that transactions appear to be serialized.

D: Durability means that a transactions' committed changes must persist through crashes.

E: None of the above.

Problem 2f[2pts]: Which of the following are true about Project 3?

A: The cache can coalesce writes into a single disk operation.

B: The path `/../..` is considered invalid.

C: The buffer cache contains file data blocks, `inode_disk` structs, and `inode` structs.

D: Different file descriptors can reference the same inode.

E: None of the above.

Problem 2g[2pts]: Which of the following strategies are valid ways to detect congestion in a network?

A: Increases in packet round time.

B: Dropped packets.

C: Sender has not received ACK from receiver.

D: Query the ISP's congestion API.

E: None of the above.

Problem 2h[2pts]: Select all of the following that are true about networking.

A: Sockets can only be used to communicate between two different machines.

B: Congestion control avoids overflowing the receiving buffer of the destination server.

C: MAC addresses are assigned by your ISP.

D: TCP does not provide guarantees on latency of bytes being delivered.

E: Packet loss in a network is the result of network congestion.

Problem 2i[2pts]: Select all of the following that are true about distributed transactions: (*choose all that apply)*:

A: ☐ Distributed transactions consider two or more machines atomically reaching consensus within a fixed time interval.

B: ☐ The General's Paradox states that without reliable messaging, it is impossible for the two generals to come to the same decision.

C: ☐ In the 2PC, a coordinator asks all workers to vote whether to commit a transaction or abort. The election will only succeed if all workers vote to commit.

D: ☐ 2PC is resilient to any failure of a worker node.

E: ☐ Byzantine Generals is the same problem as the General's Paradox.

Problem 3: Short Answer [29pts]

Please write your answer in THREE SENTENCES OR LESS (Answers longer than this may not get credit!).

Problem 3a[3pts]: Consider a network file system backed by hard drives with the following parameters.

- Network latency: 10ms
- Disk seek latency: 3ms
- Disk rotational speed: 6000 RPM
- Disk controller latency: 1ms
- Disk transfer speed: 100MB/s

What is the average latency to read 50 sequential 4KB sectors from this network file system? Assume there are no other pending requests to the system, and the CPU processing overhead is negligible.

Problem 3b[3pts]: Using Little's Law, what is the maximum amount of inflight data in a network? Explain.

Problem 3c[3pts]: Is Two Phase Commit subject to the General's Paradox?

Problem 3d[3pts]: Consider a TCP network implemented over the US Postal Service. The latency to send a packet/letter from Berkeley, CA to Cambridge, MA is 2^{18} seconds (3 days). Each packet/letter can store 32 KiB of written text. If we use the stop-and-wait protocol, what is the bandwidth of this network in bits per second? Explain.

Problem 3e[3pts]: For project 3, you are using a static array for your buffer cache with a clock algorithm for eviction. What metadata should each cache block in the buffer cache have, and what should that metadata be initialized to? Explain.

Problem 3f[2pts]: What is the major drawback of polling strategies versus I/O interrupts?

Problem 3g[3pts]: Consider a system using RAID 5. Ignoring any metadata operations (e.g. inode accesses), how many disk I/O operations must occur when updating a data block? Explain.

Problem 3h[3pts]: 3D XPoint is an emerging memory technology that provides persistent storage with near-DRAM throughput. 3D XPoint supports updating blocks in-place. How does this simplify an SSD's FTL firmware?

Problem 3i[3pts]: Give a *specific* example why the End-to-End Principle says that reliable file transfer should be implemented at end hosts and not in the network.

Problem 3j[3pts]: Journaling file systems require data to be written twice. How can the file system optimize for good response time? How about good throughput? Explain.

Problem 4: Pintos Logs [16pts]

Gojo Satoru decides to add logging to Pintos. He plans for the log to be a fixed contiguous collection of blocks (on disk) in the following format. Each row shown is of `BLOCK_SECTOR_SIZE`.

Header block
block A
block B
...

You are given the following methods and data structures.

```
/* Log */
struct log {
  uint32_t size;        /* Size of log (in number of sectors). */
  block_sector_t start; /* Starting block number of log (i.e. header block). */
  struct logheader lh;  /* In-memory copy of the log header block. */
};

/* Log header */
struct logheader {
  uint32_t n;           /* Number of blocks used in log excluding header. */
  block_sector_t block[BLOCK_SECTOR_SIZE / 4 - 1];
};

static struct log log;  /* In-memory representation of log, initialized at boot. */

#define BLOCK_SECTOR_SIZE 512
struct block* fs_device;
void block_read(struct block* block, block_sector_t sector, void* buffer);
void block_write(struct block* block, block_sector_t sector, void* buffer);
```

For instance, consider the following transaction.

1. Write all 0's to the block at `block_sector_t 2000`,
2. Write all 1's to the block at `block_sector_t 2500`,

The log would look like.

struct logheader lh = {2, {2000, 2500, ...}};
0000000000…
1111111111…
...

Gojo also makes the following design decisions:

- The log can only contain at most one transaction at a time.
- Instead of writing a commit message to the log, a transaction is considered committed when the header block is written to disk.
- All file system syscalls hold the global file system lock for simplicity.
- There is no buffer cache.

With logging, a typical file system syscall will look like the following.

```
lock_acquire(&fs_lock);
...
/* Perform syscall but use log_read/log_write instead of block_read/block_write. */
...
commit();
lock_release(&fs_lock);
```

Problems 4a-4m[1pt each]: Help Gojo implement this logging system. Only one piece of code should be written per blank (i.e. no multiple statements with semicolons and no blank lines). Use proper C syntax with the given APIs. Pseudocode or comments will not receive any credit. You may only use methods and data structures given in this question as well as built-in ones.

```
/* Read the log header from disk into log.lh. */
static void read_head(void) {
  ____________________[4A]____________________;
}


/* Write log.lh to the log header on disk. */
static void write_head(void) {
  ____________________[4B]____________________;
}


/* Add an entry to the log to represent the action "write BUFFER to the block at
   SECTOR." Assume read_head has already been called. */
void log_write(block_sector_t sector, void* buffer) {
  /* Check if block already in log. */
  for (int i = 0; i < log.lh.n; i++) {
    if (____________________[4C]____________________){
      block_write(fs_device, log.start + i + 1, buffer);
      return;
    }
  }

  ____________________[4D]____________________;
  block_write(fs_device, log.start + log.lh.n + 1, buffer);
  ____________________[4E]____________________;
}
```

```
/* Read the block at SECTOR from disk. If the log contains an updated version of
   the block, read that instead. Assume read_head has already been called. */
void log_read(block_sector_t sector, void* buffer) {
  /* Check if block already in log. */
  for (int i = 0; i < log.lh.n; i++) {
    if (_____________________[4F]_____________________) {
      _____________________[4G]_____________________;
      return;
     }
  }
  _____________________[4H]_____________________;
}

/* Copy committed blocks from log to their home location. */
static void install_trans(void) {
  uint8_t* data = _____________________[4I]_____________________;
  for (int i = 0; i < log.lh.n; i++) {
    block_read(_____________________[4J]_____________________);
    block_write(_____________________[4K ]_____________________);
  }
}

/* Commit transaction to disk and clean up. */
static void commit() {
  if (log.lh.n > 0) {
    /* Commit and apply transaction. */
    _____________________[4L]_____________________;
    _____________________[4M]_____________________;

    /* Clear the log. */
    log.lh.n = 0;
    write_head();
  }
}
```

Problem 4n[3pts]: Suppose Gojo's computer crashes unexpectedly. When Gojo powers up his computer again, how can the OS tell if the transaction stored in the log has been committed? Provide a clear criteria and explain.

Problem 5: Filesystem Design [17 pts]

Tony Stark has hired you to create a file system to store the designs for his newest Iron Man suit. He would like to use 32-byte sectors and 2-byte disk pointers. Blocks and sectors may be used interchangeably throughout this question.

Problem 5a[2pts]: Tony wants to optimize for a workload where each file is accessed as a whole (i.e. the entire file is read in). Most of Tony's files are really large. Given the choice between FAT and FFS, which would be better suited for minimizing access time for the given workload? Explain.

Problem 5b[3pts]: You first consider using a filesystem with indexed inodes. Each inode will contain 2 bytes of metadata. Given that each inode must fit exactly into a single sector, how many direct, indirect, and doubly indirect pointers should you use to support a maximum file size of exactly 10,560 bytes? Explain.

For 5c-5e, the indexed inode system you are considering will have 12 direct pointers, 2 indirect pointers, and 1 indirect pointer. **Note that this may be a DIFFERENT design from the previous part**. You are working with a 9,600 byte file called `jarvis.txt`.

Problem 5c[3pts]: Assuming the file number is known (i.e. directory lookup not required), how many disk sectors would need to be accessed to read the last byte of `jarvis.txt` with the indexed inode system? Explain.

Problem 5d[3pts]: Assuming the file number is known (i.e. directory lookup not required), how many disk sectors would need to be accessed to read the last byte of `jarvis.txt` file with FAT? Explain.

Problem 5e[3pts]: With the indexed inode system, what fraction of the disk space is used by `jarvis.txt` for the data? Explain. Leave your answer in an unsimplified fraction.

Problem 5f[3pts]: You move on to making the system reliable. Assume there are 5 disks worth of data. You consider using either RAID 1 or RAID 5 to store your data across these disks. Disk space is expensive, so Tony wants to spend the least amount of money necessary. Which method, between RAID 1 or RAID 5, would minimize the number of extra disks needed?

[This Page Intentionally Left Blank]

[Scratch Page: Do not put answers here!]

[Scratch Page: Do not put answers here!]

Q3a
- Note powers of 10 for certain units.
- Network bandwidth is 50 sectors per 10 ms.
- Network latency refers to round trip time (RTT).
- Refers to a single network request

Q3b
- Not related to Q3a.

Q3d
- Network latency refers to round trip time (RTT).

Q3g
- Assume no cache.

Q4
- Built in means libc.
- Don’t worry about memory leaks.

Q5a
- Assume all files fit in both FAT and FFS

Q5c-e
- “...2 indirect pointers and 1 DOUBLY indirect pointer”
- “...DIFFERENT design…” refers to number of each pointer type, NOT sector or pointer size

Q5e
- “What fraction of the disk space used by jarvis.txt is for the data”

Q5f
- You must explain your answer.