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

Fall 2016 Anthony D. Joseph

Midterm Exam #3

November 30, 2016
CS162 Operating Systems

Your Name:	
SID AND 162 Login:	
TA Name:	
Discussion Section Time:	

General Information:
This is a **closed book and *three* 2-sided handwritten notes** examination. You have 80 minutes to answer as many questions as possible. The number in parentheses at the beginning of each question indicates the number of points for that question. You should read **all** of the questions before starting the exam, as some of the questions are substantially more time consuming.

Write all of your answers directly on this paper. ***Make your answers as concise as possible.*** If there is something in a question that you believe is open to interpretation, then please ask us about it!

Good Luck!!

QUESTION	POINTS ASSIGNED	POINTS OBTAINED
1	34	
2	35	
3	16	
4	15	
TOTAL	100	

NAME: ____

1. (34 points total) Short answer.

a. (12 points) True/False and Why? **CIRCLE YOUR ANSWER.**

i) The Pintos and BSD 4.2 Fast File Systems use a linked-list on-disk data structure to track the allocation of inodes and data blocks.

TRUE FALSE

Why?

ii) Two processes communicating via a mailbox could be considered an example of a producer-consumer model of process cooperation.

TRUE FALSE

Why?

iii) A remote procedure call requires network access.

TRUE FALSE

Why?

iv) If machines were guaranteed not to crash, we would not need two-phase commit: the coordinator could, in one phase, decide whether a distributed transaction would commit, and then instruct the workers to apply their piece of the transaction.

TRUE FALSE

Why?

b. (4 points) Briefly, in two to three sentences, explain what a DMA controller does and what benefit it provides.

c. (9 points) Abstraction layers.

i) (2 points) Let us explore how abstraction layers help in the design of an operating system. Give a concrete example of a use of abstraction layers in an operating system.

ii) (4 points) Briefly, in one to two sentences, explain the advantages of using the abstraction layers in your example.

iii) (3 points) Briefly, in one to two sentences, explain a disadvantage of using the abstraction layers in your example.

NAME: ____

d. (9 points) Networking: Two computers, A and B, are connected by a network link that runs at 1 Gigabit per second (1×10^9 bits/second). The one-way propagation delay from A to B (and vice versa) is 20 milliseconds. Assume the following conditions:

- The link does not drop or duplicate packets
- Processing time at the two endpoints is zero
- A sends B 625-byte packets – the 625 bytes includes all headers, framing, and inter-packet spacing

i) (3 points) What is the maximum number of 625-byte packets per second that A could in principle send into the wire? *You can leave your answer in unsimplified form.*

ii) (6 points) Now, consider the above link and the following protocol, and assume that all packets are again 625 bytes. In the protocol, A sends 4,000 packets into the network as quickly as it can and then waits for a one-byte ACK from B. Whenever A receives an ACK from B, A immediately sends another 4000 packets into the network. Meanwhile, B ACKs packet 1, packet 4001, packet 8001, etc. That is, B ACKs every 4,000 packets starting with the *first* one in a burst by *A*.
What is the long-term throughput of this protocol, expressed as *both* (1) bits per second and (2) a percentage of the link's bandwidth? *You can leave your answer in unsimplified form. Briefly explain your answers.*

NAME: ____

2. (35 points total) Filesystems.

a. (4 points) One of the innovations of the BSD FFS is that it tries to allocate large files in long contiguous chunks. Assuming that a disk transfers at a peak rate of 100 MByte/s, and that a combined seek and rotation take, on average, a total of 20 milliseconds. What is the **minimum** size of *each* contiguous run of a large file, in order to achieve 75% of peak transfer rate for large files when they are accessed sequentially?

b. (15 points) Large files. We have seen different filesystems that support fairly large files. Now let's see just how large a file various types of filesystems can support. Assume, for all of the questions in this part, that filesystem blocks are 4 KBytes. *Show your solutions in unsimplified form for partial credit.*

i) (3 points) Consider a really simple filesystem, **directfs**, where each inode only has 10 direct pointers, each of which can point to a single file block. Direct pointers are 32 bits in size (4 bytes). What is the maximum file size for **directfs**?

ii) (3 points) Consider a filesystem, called **extentfs**, with a construct called an extent. Extents have a pointer (base address) and a length (in blocks). Assume the length field is 8 bits (1 byte). Assuming that an inode has exactly one extent. What is the maximum file size for **extentfs**?

iii) (3 points) Consider a filesystem that uses direct pointers, but also adds indirect pointers and double-indirect pointers. We call this filesystem, **indirectfs**. Specifically, an inode within **indirectfs** has 1 direct pointer, l indirect pointer, and 1 doubly-indirect pointer field. Pointers, as before, are 4 bytes (32 bits) in size. What is the maximum file size for **indirectfs**?

iv) (3 points) Consider a compact file system, called **compactfs**, tries to save as much space as possible within the inode. Thus, to point to files, it stores only a single 32-bit pointer to the first block of the file. However, blocks within **compactfs** store 4,092 bytes of user data and a 32-bit next field (much like a linked list), and thus can point to a subsequent block (or to NULL, indicating there is no more data). First, draw a picture of an inode and a file that is 10 KBytes in size.

v) (3 points) What is the maximum file size for **compactfs** (assuming no other restrictions on file sizes)?

NAME: ____

c. (5 points) File system crash recovery.

The Linux journaling file system writes the content of all modified disk blocks to the log. Your friend Bob considers such logging to be wasteful since copying the content of modified disk blocks to the log doubles the amount of disk writes for each logged file system operation.

Bob decides to implement a more efficient journaling file system. In particular, he decides to only record an operation's name and parameter in the log file instead of recording the content of all modified blocks. For example, for an operation that creates file "`/d/f`", the file system would append the transaction record of the form [create "`/d/f`"] to the log. Bob's file system ensures that the corresponding transaction record is written to the log before the modified disk blocks are flushed to disk. Upon crash and recovery, Bob's file system re-executes the logged file system operations and truncates the log.

Bob's new logging mechanism is certainly more efficient since each transaction record is much smaller than that with Linux's logging. Is his design also correct? Specifically, can it recover a file system correctly from crashes? Explain your reasoning and give concrete examples.

d. (3 points) Your friend suggests doubling the Pintos block size from 512 bytes to 1,024 bytes, since that means you will be able to reach twice as much data from the direct pointers (as a result, medium sized files could fit entirely within the direct region). Why might it be a bad idea to increase the block size?

NAME: ____

e. (4 points) Suppose you wanted to implement hard links in Pintos. One thing you have to add is a count to ensure you only delete sectors that have no links remaining. Which struct would be the best choice to add this count? Explain your choice.

f. (4 points) The NFS authors had a goal of *transparency*. They wanted applications to be unable to distinguish whether a file system was (a) a remote file system served from an NFS server; or (b) a typical, local Unix file system. They did not succeed (in fact, their goal was impossible). Briefly state one way in which application code can experience different behavior when interacting with a remote NFS file system versus a local Unix file system. Your answer should be in terms of what application code sees, rather than in terms of what a global observer sees.

NAME: ____

3. (16 points total) Cybersecurity.

a. (8 pts) For each of the following terms, provide a one sentence definition:

i) Authentication

ii) Data integrity

iii) Symmetric encryption

iv) Asymmetric encryption

b. (2 points) Briefly, in one to two sentences, explain the role of a nonce.

c. (2 points) Briefly, in one to two sentences, explain the role of a HMAC.

d. (4 points) Consider two approaches to delegating your access privileges to a server:

- Give your login name and password, so that the server can authenticate to a third party as you, or
- Give the server a capability (e.g., your Kerberos ticket) to act as you.

From a security standpoint, which approach is superior? Briefly explain why.

NAME: ____

4. (15 points total) Queuing Theory.

Consider a single server/single queue (M/M/1) system:

- 5 jobs per second arrive to the server
- The server is capable of executing 8 jobs per second.

Write your answers in unsimplified form.

a. (3 pts) Calculate server utilization (*u*).

b. (3 pts) Calculate how long it takes the server to execute one job (T_{ser}).

c. (3 pts) Calculate the amount of time spent in the queue (T_q).

d. (3 pts) Calculate the number of jobs in the queue (L_q).

e. (3 pts) Calculate the amount of time a job spends in the system (T_{sys}).