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

Fall 2013 Anthony D. Joseph and John Canny

Midterm Exam #2 *Solutions*

December 4, 2013
CS162 Operating Systems

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

General Information:
This is a **closed book and one 2-sided handwritten note** 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	8	
2	21	
3	23	
4	32	
5	16	
TOTAL	100	

1. (8 points total) True/False and Why? **CIRCLE YOUR ANSWER.**

i) A Remote Procedure Call (RPC) can be used to call a procedure in another process on the same machine.

TRUE FALSE

Why?

TRUE*. The client and server addresses can be the same. Location transparency is a fundamental aspect of RPC. The correct answer was worth 1 points and the justification was worth an additional 1 point.*

ii) Doubling the block size in the UNIX 4.2 BSD file system will exactly double the maximum file size.

TRUE FALSE

Why?

FALSE*. The maximum file size will more than double since larger indirection blocks can contain more pointers to the larger blocks. The correct answer was worth 1 point and the justification was worth an additional 1 point.*

iii) With the NFS distributed file system, it is possible for one client to write a value into a file that is not seen by another client when reading that file immediately afterwards.

TRUE FALSE

Why?

TRUE*. Since NFS only checks every 30 seconds or so, it is possible for a write on one client to go unnoticed by another client for a bit. The correct answer was worth 1 points and the justification was worth an additional 1 point.*

iv) In a replicated Key-Value storage system, Iterative PUTs achieve lower throughput than recursive PUTs on a loaded system.

TRUE FALSE

Why?

FALSE*. On a loaded system, the master is a bottleneck in a* ***recursive*** *system. The correct answer was worth 1 points and the justification was worth an additional 1 point.*

2. (21 points total) Security.

a. (5 points) Are digital signatures on digital documents more or less secure than ink signatures on printed documents? Give a brief (4-5 sentence) justification for your answer.

Digital signatures are more secure than ink signatures because a digital signature is bound to a secure hash value computed over the document contents. Any change to the document will change the hash value, and this allows the receiver to determine that the signature is invalid. An attacker cannot modify the signature for the new hash value unless it knows the signer's public key. In contrast, a signature on a printed document contains no information about the document. A printed signature looks almost the same on any document, and can be copied to a document without the signer's knowledge. If the document can be modified in a way that is not obvious to a reader, then the reader may still consider the signature to be valid.

b. (8 points total) X.509 Certificates.

i) (5 points) The recipient of an X.509 certificate can authenticate the sender of the certificate without contacting a third party (e.g., a certificate authority) at the time of authentication. How does this work?

The certificate is a digitally signed document from the third party endorsing the bearer's public key. Any receiver can determine that the certificate is valid just by examining it, assuming the receiver knows the public key of the third party that issued the certificate. It does not need to contact the third party to validate the certificate. If the certificate is valid, the bearer can then authenticate to the receiver by proving that it holds the private key corresponding to the public key endorsed in the certificate. It can prove this by successfully using its private key to encrypt or decrypt content known to the receiver.

For example, the receiver might issue a challenge with a random number (nonce), and ask the bearer to encrypt the nonce and return it. The receiver *can then decrypt the response with the public key in the certificate: if the nonce matches, then it knows that the sender is the entity named in the certificate, i.e., it possesses the corresponding private key.*

ii) (3 points) Discuss a vulnerability in the use of X.509 certificates as described in part (i).
A compromise at certificate authority means that a certificate could be fraudulent.

c. (8 points) SYN Cookies.

i) (3 points) What type of attack do SYN cookies protect against? Be specific in your answer.

SYN cookies protect against a denial of service attack on a server, where an attacker attempts to exhaust the available socket memory resources by sending many SYN packets with spoofed source addresses.

ii) (5 ponts) Explain how SYN cookies are used?

. *Using SYN cookies, a server does not allocate any resources when it receives a SYN packet. Instead it returns a SYN ACK packet containing the sequence number: y = HMAC(client_IP_addr, client_port, server_key). Only when the client responds with an ACK containing sequence number y+1 does the server allocate any memory. The server can verify the correct ACK by recomputing the HMAC function.*

3. (23 points total) Transactions.

a. (16 points) Consider the following bank transfer process for moving funds from account A to account B:

Read balance from source account:	**Read Source Account**
Decrement source account balance	
Write new balance of source account:	**Write Source Account**
Read balance from destination account:	**Read Destination Account**
Increment destination account balance	
Write new balance of destination account:	**Write Destination Account**

Suppose John and Alice share the same accounts and they each use different ATMs at the same time. John transfers $100 from Checking to Savings; while Alice transfers $200 from Savings to Checking – *note that one transfer goes from Checking to Savings, while the other is reversed*. Suppose that John and Alice's transfers happen simultaneously, and there is more than $200 in the account.

i) List a **serializable schedule** for the transactions, which is not simply already a **serial schedule**

Yes, example:

T1	*T2*
Ra, Wa	
	Rb, Wb
Rb, Wb	
	Ra, Wa

ii) For each of the following three *potential* situations, state whether such a situation can exists, and if it can, give an example. Otherwise, explain why it cannot exist.

(1) Situation #1: A **non-serializable schedule** for the transactions
Yes, any interleaving of the read and write operations to the same account.

(2) Situation #2: A **serializable schedule** using Two-Phase Locking, which is not simply already a **serial schedule**
There are no conflict serializable schedules, so there is no 2PL serializable schedule other than a serial schedule.

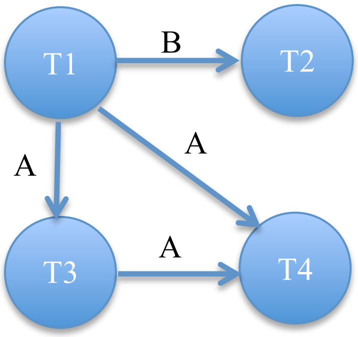
(3) Situation #3: A schedule where Two-Phase Locking causes deadlock
Yes, example:

T1	*T2*
Lock(a)	
Ra, Wa	
	Lock(b)
	Rb, Wb
Lock (b) [DEADLOCK]	
Rb, Wb	
	Lock(a) [DEADLOCK]
	Ra, Wa

b. (7 points total) Consider the following schedule of four transactions:

T1:		R(B)	R(A)	W(A)				W(D)
T2:					W(B)			
T3:						W(A)		
T4:	W(C)						W(A)	

i) (4 points) Draw the dependency graph for this schedule.



ii) (3 points) Is this schedule serializable? If so, give a possible serial schedule. Otherwise, explain why not.

The schedule is serializable because the dependency graph is acyclic. Valid serial schedules are: 1,2,3,4 and 1,3,4,2 and 1,3,2,4

4. (32 points) Filesystems.

a. (8 points) Consider a file system that has 2,048 byte blocks and 32-bit disk block pointers to those blocks. Each file header has 12 direct pointers, one singly-indirect pointer, one doubly-indirect pointer, and one triply-indirect pointer.

i) (4 points) How large of a **disk** can this filesystem support? *You may leave your answer in symbolic form.*

2^{32} blocks x 2^{11} bytes/block = 2^{43} = 8 Terabytes.

-3 missing block or disk size in calculation

ii) (4 points) What is the maximum file size? *You may leave your answer in symbolic form.*

Each 2,048 byte block contains 512 4-byte pointers, so the maximum file size is: blockSize x (numDirect blocks + numIndirect blocks + numDoubly-indirect blocks + numTriply indirect blocks)

$= 2{,}048 \times (12+512+512^2 +512^3)=211 \times (2^2 \times 3+2^9 +2^{9x2} +2^{9x3})$

$= 2^{13} \times 3+2^{20} +2^{29} +2^{38}$ = 24K + 513M + 256 G

b. (6 points) Rather than writing updated files to disk immediately when they are closed, many UNIX systems use a delayed *write-behind policy* in which dirty disk blocks are flushed to disk once every 30 seconds. *List two advantages and one disadvantage of such a approach.*

Advantage 1:

The disk scheduling algorithm (e.g., C-SCAN) has more dirty blocks to work with at any one time and can thus do a better job of scheduling the disk arm.

Advantage 2:

Temporary files may be written and deleted before data is actually written to disk.

Disadvantage:

File data may be lost if the computer crashes before the data is written to disk.
-2 not listing an advantage or disadvantage -1 unclear explanation.

c. (4 points) Briefly (in 2-3 sentences) explain the differences between a hard link and a soft link.

Hard links point to the same inode, while soft links simply contain the name of a directory entry. Hard links use reference counting. Soft links do not and may have problems with dangling references if the referenced file is moved or deleted. Soft links can span file systems, while hard links are limited to the same file system.

d. (6 points) List the set of disk blocks that must be read into memory in order to read the file `/home/cs162/midterm2.txt` in its entirety from a UNIX BSD 4.2 file system (10 direct pointers, a singly-indirect pointer, a doubly-indirect pointer, and a triply-indirect pointer). Assume the file is 15,234 bytes long and that disk blocks are 1024 bytes long. Assume that the directories in question all fit into a single disk block each. *Note that this is not always true in reality*.

1. *Read in file header for root (always at fixed spot on disk).*
2. *Read in first data block for root (/).*
3. *Read in file header for home.*
4. *Read in data block for home.*
5. *Read in file header for cs162*
6. *Read in data block for cs162.*
7. *Read in file header for midterm2.txt.*
8. *Read in data block for midterm2.txt.*
9. *. – 17. Read in second through 10th data blocks for midterm2.txt.*
18. *Read in indirect block pointed to by 11th entry in miderm2.txt's file header*
19. *– 23. Read in 11th – 15th midterm2.txt's data blocks. The 15th data block is partially full.*

-1 point, 5th data block of test.doc is partially full -1 point, missing an inode or a data block -3 points, not listing all direct pointers, indirect pointers, and data blocks for midterm2.txt.

e. (4 points) On a single UNIX machine, if some program B reads a block of a file after it has been updated by another program A, the copy of the file block B reads will include A's updates. In NFS (as described in lecture) this behavior is not guaranteed. Assuming that there are no failures, why doesn't NFS necessarily provide such update semantics when programs A and B are run on different machines? What semantics does it provide instead?

In NFS, cached data is updated only periodically. Thus, it is possible that B could read old data for a while after A has finished updating it. The semantics are those of "weak coherence".

-2 no mention that NFS caches data at the client -2 no mention that NFS periodically updates server

f. (4 points) The Andrew File System (AFS) solves the above problem in part (e) by using state information it maintains at the server. What state is kept? How is the state used to solve the problem?

An AFS server keeps track of which clients have copies of particular files. Thus, when one client writes data ***and closes the file so that the data is flushed to the server****, the server contacts each of the clients that have cached copies of the file and tells them to invalidate the file. -2 points for no mention client updates server when file is closed*

5. (16 points) Networking.

a. (4 points) Consider a TCP network connection with packet size 1000 bytes, and current receiver advertised window size of 100 packets, over a cross-country link with one-way latency (for a 0-byte packet) of 50 milliseconds in each direction, and a link bandwidth of 100 Mbit/second. You may assume that no packets are lost for this particular problem, and that the times to assemble, unpack and process packets at each end of the connection are negligible.

How long does it take TCP to transmit 1 million bytes across the link? That is, how much time elapses from when the first byte is sent by the sender to when the sender *knows* that the receiver has received the last byte?

A TCP transmission window size of 100 packets implies that the sender can send 100 packets before having to wait for an ACK message from the receiver that will allow it to continue sending again. Then the sequence for messages sent is:

1. *100 packets from sender to receiver: requires 50 ms for first packet to get there and another 800kb/(100 Mbpst) = 8 ms for the rest of the 100 packets to get there after that.*
2. *Sender stalls waiting for ACK.*
3. *ACK msg from receiver to sender sent at t=50ms from start: requires 50 ms to get there, now 100ms from time first packet was sent. Note that transmission does not stall for ACKs for the other packets (these should come "just in time" if there is no loss).*
4. *At t=100ms sender receives first ACK and sends next packet. The next 99 ACKs should arrive just in time and the sender should send the next 99 packets without stalling.*

Sending 1M bytes will require 10 round trips of this kind, plus the time to receive the acknowledgements for the last 99 packets in the last round.
*The total time required is: 10 * (2 * 50 ms) + 99 * (1000 * 8 / 100 Mb/sec)*
~ 1,008 ms = 1.008 seconds
-2 points for missing the time to transmit 1000 bytes -2 points for using one-way latency rather than round-trip-time (RTT) -1 for adding the window transmission time every round instead of only at the end (would give answers like 1.08 seconds).

b. (4 points) Assume that the receiver can process incoming data with zero latency, what is the optimal window size that the receiver should advertise?

The optimal window size will keep the "pipe" full during the time it takes for data to arrive at the receiver and the ACK to arrive back at the sender. Thus, the receiver should advertise the Bandwidth Delay Product:

(2 x 50 milliseconds x 100 Mbit/s) = 10 Mbits or 1.25 Mbytes, or 1250 packets.

-1 point for using the one-way latency rather than RTT -2 points for no justification and specifying any window larger than 1.25Mbytes

c. (4 points) Polling a device for the completion of an operation is typically a bad idea because, as with busy-waiting, while polling, the CPU is not doing useful work. However, this is not always true. Describe a situation where polling might be a better choice than using interrupts.

If a process is waiting for an operation that will occur soon, then it is not worth the overhead to do the context switch involved in blocking on an interrupt. More generally, when the time to wait is less than the overhead time for the context switch, it is better to poll. The example given in lecture was that network device drivers typically use interrupts to receive the first packet and then poll for additional packets.

d. (4 points) Explain the difference between an IP address and a MAC address, and why we cannot use only MAC addresses or IP addresses alone.

A MAC address is uniquely associated with a device for the entire lifetime of the device, while an IP address changes as the device location changes (your laptop IP address at school is different from its IP address at home).

Also: MAC addresses non-hierarchical: you cant route with them and you would need address tables with every host on the internet, hence you need IP addresses. Why you need MAC address: before a host even has a global IP address it needs to communicate with routers on its subnet to get one, and it needs a unique address for local routing to work since many network devices use broadcast (e.g. hubs).

This page intentionally left blank as scratch paper

Do not write answers on this page