Midterm II *Solutions*

CS 414 Operating Systems, Spring 2007
April 26th, 2007
Prof. Hakim Weatherspoon

Name: ________________________________NetId/Email:__________________________

Read all of the following information before starting the exam:
Write down your name and NetId/email NOW.

This is a **closed book and notes** examination. You have 90 minutes to answer as many questions as possible. The number in parentheses at the beginning of each question indicates the number of points given to the question; there are 100 points in all. 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 a question is unclear, please simply answer the question and state your assumptions clearly. If you believe a question is open to interpretation, then please ask us about it!

Good Luck!!

Problem	**Possible**	**Score**
1	24	
2	20	
3	32	
4	16	
5	12	
Total	**104**	

1. (24 points) True/False.
Circle either True or False. In the following, it is important that you EXPLAIN your answer in *TWO SENTENCES OR LESS* (Answers longer than this may get ***no credit***!). Also, answers without an explanation GET ***NO CREDIT***.

a. (3 points) "Marshaling" is the process by which Byzantine Generals are forced into making good decisions.

TRUE / [✓ FALSE]
EXPLAIN: *Marshalling is the process of packaging data items (such as arguments for an RPC) into a network message.*

b. (3 points) A "broadcast network" is one which uses radio-frequency transmission to send data from one party to another.

TRUE / [✓ FALSE]
EXPLAIN: *A "broadcast network" is one in which multiple receivers can receive a message at the same time from a single sender.*

c. (3 points) Randomness is essential to achieving good performance from the Ethernet communication algorithm (CSMA/CD).

[✓ TRUE] / FALSE
EXPLAIN: *Randomness is required to break up collisions so that they are eventually resolved*

d. (3 points) A Remote Procedure Call (RPC) can be used to call a procedure in another process on the same machine.

[✓ TRUE] / FALSE
EXPLAIN: *Just make the client and server addresses to be the same. Location transparency is a fundamental aspect of RPC.*

e. (3 points) Using the TCP/IP protocol over an unreliable network, the receiver can receive the same IP packet multiple times.

[✓ TRUE] / FALSE
EXPLAIN: *If an ACK is lost, the sender must assume that the packet could habe been lost. As a result, it will resend the packet.*

f. (3 points) 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.

[✓ TRUE] / FALSE
EXPLAIN: *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.*

g. (3 points) The fastest way to send a large document securely to a third party that you have not interacted with yet is to encrypt it with their public key. Assume you know everyone's public key.

TRUE / [✓ FALSE]
EXPLAIN: *Since public key encryption is so slow, it is much faster to use the public key to exchange a secret key with the receiver, then use private-key encryption for the bulk of the document.*

h. (3 points) Doubling the block size in the UNIX 4.2 BSD file system will exactly double the maximum file size.

TRUE / [✓ FALSE]
EXPLAIN: *File size more than doubles since bigger blocks can contain more pointers to larger blocks.*

In general,
- *-1 if true / false incorrect*
- *-1 if explanation partially incorrect*
- *-2 if explanation completely incorrect*
- *-3 if true / false and explanation incorrect*

EXTRA CREDIT

(3 points) Consider two processes P and Q that are communicating using mailboxes. From P → Q they use mailbox A, and from Q → P they use mailbox B. Assume both the processes are asynchronous, mailboxes are currently empty, and the communication links between P and Q are unreliable. If P now wants to determine that Q has crashed, what is the sequence of instructions P should execute?
You may only use the following blocking instructions: send, reply, re-send, receive, deleteMailbox, createMailbox, and setAlarm. Note that an alarm will interrupt a blocking instruction.

Not possible to determine that Q has crashed since network is asynchronous and unreliable

2. (20 total points) Disks.

a. (12 points) Disk requests come into the disk driver for cylinders: 10, 22, 20, 2, 40, 6, 38, in that order. The disk has 60 total cylinders and the disk head is currently positioned over cylinder 20. A seek takes 6 milliseconds per cylinder moved. What is the sequence of reads and total seek time using each of the following algorithms?

i) (4 points) First-come, first-served:

10, 22, 20, 2, 40, 6, 38
10 + 12 + 2 + 18 + 38 + 34 + 32 = 146 cylinders = 876 milliseconds.

ii) (4 points) Shortest Seek Time First:

20, 22, 10, 6, 2, 39, 40
0 + 2 + 12 + 4 + 4 + 36 + 2 = 60 cylinders = 360 milliseconds.

iii) (4 points) LOOK (initialing moving upwards):

20, 22, 38, 40, 10, 6, 2
0 + 2 + 16 + 2 + 30 + 4 + 4 = 58 cylinders = 348 milliseconds.

-1 answer given in only cylinders and not milliseconds
-1 calculation mistake
-2 wrong concept but correct sequence (confused between SCAN and LOOK)
-3 wrong concept and wrong sequence (confused C-LOOK and LOOK)

b. (7 points) RAID

i) (4 points) Give a brief (2-3 sentences) description of RAID 5

Redundant Array of Inexpensive (or Independent) Disks level 5 stripes blocks of data across at least three drives along with an interleaved XOR-based parity block for each stripe set.

-2 no mention of block striping
-2 no mention of interleaved parity

ii) (2 points) How many disk failures can RAID 5 tolerate without losing data?

One.

iii) (2 points) How would you reconstruct a failed disk?

Read blocks from the other drives and use an XOR of their blocks to reconstruct the data.

3. (28 points total) File Systems.

a. (8 points) Consider a file system with 2048 byte blocks and 32-bit disk and file block pointers. Each file has 12 direct pointers, a singly-indirect pointer, a doubly-indirect pointer, and a triply-indirect pointer.

i) (4 points) How large of a disk can this file system support?

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?

There are 512 pointers per block (i.e. 512 4-byte pointers in 2048 byte block), so:
blockSize x (numDirect + numIndirect + numDoubly-indirecty+numTriply indirect)

$$2048 \times (12 + 512 + 512^2 + 512^3) = 2^{11} \times (2^2 \times 3 + 2^9 + 2^{9x2} + 2^{9x3})$$
$$= 2^{13} \times 3 + 2^{20} + 2^{29} + 2^{38}$$
$$= 24K + 513M + 256\ G$$

b. (4 points) Briefly (2-3 sentences) state the differences between a hard link and a soft link.

Hard links point to the same inode, while soft links simply list 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.

c. (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 scheme.

Advantage 1: *The disk scheduling algorithm (i.e. 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 written to disk.*

Disadvantage: *File data may be lost if the computer crashes before data is written to disk.*

-2 not listing an advantage or disadvantage
-1 unclear explanation

d. (6 points) List the set of disk blocks that must be read into memory in order to read the file /home/cs414/test.doc 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 cs414.
6. Read in data block for cs414.
7. Read in file header for test.doc.
8. Read in data block for test.doc.
9. – 17. Read in second through 10th data blocks for test.doc.
18. Read in indirect block pointed to by 11th entry in test.doc*'s file header*
19. – 23. Read in 11th – 15th test.doc *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 test.doc

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 this behavior is not guaranteed. Assuming that there are no failures, why doesn't NFS necessarily provide such update semantics when A and B are run on different machines? What semantics does it provide instead?

In NFS (version through 3), 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 (e) 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 read-only 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

4. (16 points total) Network Performance.

a. (8 points) Consider a TCP network connection with a current window size for unacknowledged bytes of 1,000 bytes, over a cross-country link with a one-way latency of 50 milliseconds, and a link bandwidth of 100 Mbit/second. You may assume that no packets are lost for this particular problem, and that the size for an acknowledgement is essentially 0 bytes long.

How long does it take TCP to transmit 100,000 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 for 1000 implies that the sender can send 1000 bytes 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. *1000 bytes from sender to receiver: requires 50 ms for first byte to get there and another 1000/(100 Mbps / 8 bytes/bit) secs for the rest of the 1000 bytes to get there after that.*
2. *ACK msg from receiver to sender: requires 50 ms to get there.*
3. *To send 100,000 bytes will require 100 round trips of this kind.*

The total time required is:
*100 * (2 * 50 ms + 1000/(100,000,000/8)) = 100 * (100 ms + 0.08ms)*
= 10,008 ms
= 10.008 seconds

-2 points for missing the time to transmit 1000 bytes
-2 points for using one-way latency rather than round-trip-time (RTT)

b. (4 points) Assume that the receiver can process incoming data at greater than 100 Mbit/s, 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

-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) If the link is shared by N pairs of senders and receivers, does your answer for part (c) change? If so, how? If not why?

To fairly share the link, the window size for each pair would be 1/N times the answer from (c).

5. (12 points total) Security.
   a. (4 points) Assume Alice and Bob have never met. Explain how Alice can use a public key infrastructure to prove her identity to Bob. You can assume that a PKI is essentially a list of everyone's public key signed by a Certificate Authority (CA). *Hint: make sure to prevent replay attacks.*

   The essential insight here is that Alice needs to sign something with her private key that can be verified with her public key. To prove to Bob what her public key is, she has a CA sign her public key. This is the PKI.

   Once Bob is certain of Alice's public key, proving her identity is easy: she signs a message (e.g. "Hi! I am Alice.") with her private key asserting that she is Alice. To avoid replay attacks, she asks Bob for a random value which she also includes in that message.

   -2 points for replay vulnerability

   b. (4 points) Explain how to utilize a PKI to establish a private session key between both parties for fast symmetric encryption.

   Both parties use the PKI to make sure that they have each other's appropriae public key. Then, they each chose a random number, encrypt it with the public key of the other party, and send it off. Now, both of them have the same two random numbers (which are private). They can combine them anyway they like, then use the result as a private session key.

   -1 points if vulnerable
   -2 points if vulnerable and cannot tolerate replays

   c. (4 points) What are two desirable properties for secure hash functions (ignoring the property where half the bits change for small changes in input). Why are these properties important for signatures?

   1. *Given message M_1 with hash $h_1=h(M_1)$, it is computationally infeasible to find another message M_2 ($M_1 \neq M_1$) such that $h_1=h(M_2)$. Known as Second Pre-image Resistant.*
   2. *Further, it is computationally infeasible to find any two messages M_A and M_B such that $h(M_A)=h(M_B)$. Known as Collision Resistant.*

   The type of signature discussed in lecture involves encrypting the hash of a document with the private key of the signer. The properties above prevent the possibility of a single signature from referring to multiple documents (which would defeat the purpose of a signature).

   -1 point for no mention of pre-image resistant property.
   -1 point for saying "collision impossible." Collisions exist (i.e. input set larger than output), just collisions computationally infeasible to find

-2 points for no mention of collision resistant property
-2 points for no mention of second pre-image property