Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

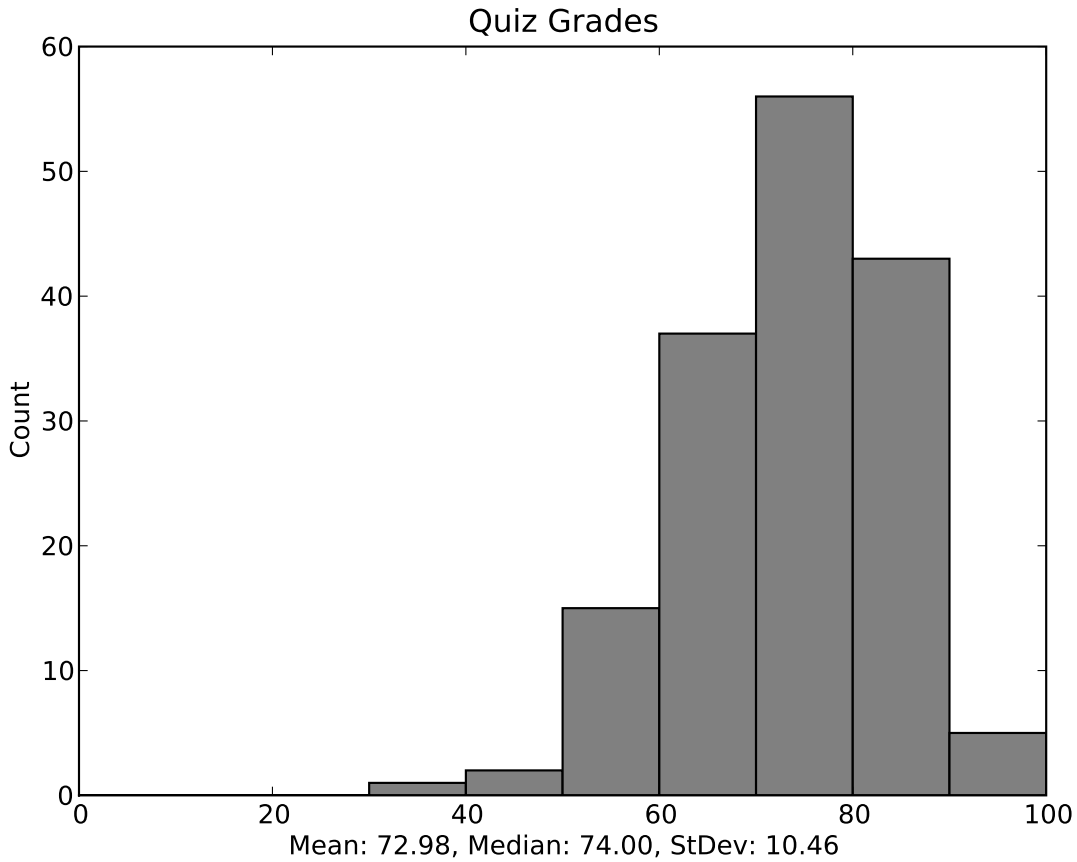
6.033 Computer Systems Engineering: Spring 2011

Quiz 3 Solutions

There are 15 questions and 11 pages in this quiz booklet. Answer each question according to the instructions given. You have **90 minutes** to answer the questions.

THIS IS AN OPEN BOOK, OPEN NOTES, OPEN LAPTOP QUIZ, BUT DON'T USE YOUR LAPTOP FOR COMMUNICATION WITH OTHERS.

Grade distribution histogram:



I Reading Questions

1. [8 points]: Based on the paper "The Recovery Manager of the System R Database Manager", which of the following statements are correct?

(Circle True or False for each choice.)

A. True / False As part of recovery from a crash, System R restores the current version of all shadowed files to their shadow copy.

Answer: True. System R discards all current edits, and restores all files to the shadowed copy which was created at the time of the last checkpoint.

B. True / False The shadow copy of a file reflects only the effects of committed transactions.

Answer: False: Uncomitted transactions in progress at the time of the last checkpoint may be present in the shadow copy

C. True / False System R will undo the effects of certain transactions during recovery.

Answer: True: it will undo transactions started before the last checkpoint but not committed.

D. True / False System R will redo the effects of certain transactions during recovery.

Answer: True: it will redo comitted transactions completed after the last checkpoint.

2. [6 points]: Based on the paper "Beyond Stack Smashing: Recent Advances in Exploiting Buffer Overruns", which of the following statements are correct?

(Circle True or False for each choice.)

A. True / False The basic form of attack described in the paper (stack smashing) allows an attacker to execute arbitrary code by overwriting far enough beyond the end of an application buffer to also overwrite the stack pointer.

Answer: False: it overwrites the return address, not the stack pointer.

B. True / False Even if a system guards against buffer overruns by making the stack non-executable, and uses stack cookies to guard against arc injection / return-into-libc attacks, the system could still be vulnerable to attacks which modify function pointers.

Answer: True: function pointers could exist in local variables without needing to overwrite a cookie value.

C. True / False A heap buffer overrun writes beyond the end of a dynamically-allocated object (e.g., created with `new` in C++ or `malloc()` in C), instead of a procedure's local variables.

Answer: True

Initials:

3. [6 points]: Which of the following statements is true about the Porcupine distributed email system described by Saito, Bershad and Levy?

(Circle True or False for each choice.)

A. True / False In Porcupine, soft state such as the user map is replicated on every node for fault-tolerance.

Answer: False. By definition, soft state can be computed from something else, so it doesn't need to be replicated for fault-tolerance. Porcupine replicates the user map for performance.

B. True / False In order to balance the load across the cluster, when a new message arrives Porcupine always picks a lightly loaded node at random and adds the message to the user's mailbox fragment on that node.

Answer: False. Porcupine stores a new message on a lightly loaded node that already has a mailbox fragment for the user, unless the number of such nodes is less that the spread. This is done to limit the number of nodes that need to be contacted when the user reads their mail.

4. [4 points]: Porcupine's scheme for maintaining cluster membership chooses a new coordinator whenever anyone detects an event: a node going up or down. The coordinator broadcasts a ping and gets a response from every node that is up. The total amount of work done in a cluster of n nodes, each time some node goes up or down, is proportional to which of the following?

(Circle the BEST answer)

- n
- $n \log n$
- n^2

Answer: The number of events per second scales linearly with the number of nodes, and each node has to respond to each event, so the total amount of work scales like n^2.

Initials:

II Lectures

5. [4 points]: According to Prof. Abelson's lecture, the "Good Samaritan Provision" of the Communications Decency Act provided protection to which of the following parties?

(Circle True or False for each choice.)

A. True / False CMU so that it can study the use of pornographic information on the Internet without being liable.

Answer: False. The CMU report wasn't the topic of a case under the CDA.

B. True / False AOL in the case against Ken Zeran, because AOL merely distributed information but was not responsible for the content.

Answer: True. AOL successfully defended itself using the Good Samaritan Provision.

C. True / False Prodigy in the case versus Stratton-Oakmont because Prodigy filtered offensive content.

Answer: False. This case predated the CDA, but was a precedent.

D. True / False The nastiest place on earth because it merely distributed content that others generated.

Answer: False. This case predated the CDA and has no immediate relation to it.

6. [8 points]: Consider the mechanisms for storing and checking user's passwords, as described in lecture.

(Circle True or False for each choice.)

A. True / False Rather than storing user's login passwords directly, Unix stores the result of applying a hash function to the user's password. One reason for this approach is so that users' plaintext passwords are not revealed if an attacker manages to read the password database.

Answer: True. The hash function is difficult to reverse, so it is hard to recover the plaintext passwords.

B. True / False Unix combines *salts* with a user's password before sending them through the hash function to enact a policy that requires users to periodically revise their passwords.

Answer: False. A salt is added such that the same password does not always map to the same hashed value

C. True / False Let $H()$ be a hash function. If given a value p, it is easy to find a value P such that $H(P)$ equals p, then $H()$ is a poor choice of hash function to use as part of a password checking system.

Answer: True. If the hash function were reversible, an attacker could find a working password easily without necessarily guessing the correct one

D. True / False The study of password use described in lecture found that different people almost never choose the same passwords.

Answer: False. It found several very commonly repeated passwords.

Initials:

7. [7 points]: Consider the Same Origin Policy (SOP) used for browser security.
(Circle True or False for each choice.)

A. True / False Javascript code from the page at `http://web.mit.edu/6.033` can access the DOM of another page at `http://web.mit.edu/6.041` loaded in the same browser. Assume the second page runs no Javascript code.

Answer: True. The port, protocol, and host are all exactly the same, so this is permitted.

B. True / False Javascript code from the page at `http://web.mit.edu/6.033` can access the DOM of another page at `http://courses.csail.mit.edu/6.033` loaded in the same browser. Assume the second page runs no Javascript code.

Answer: False. The hosts are different, so this is not permitted. Even though the hosts are part of the same subdomain (mit.edu), the same origin policy treats them as different unless each page declares a common host, but that cannot be done without Javascript code.

C. True / False Javascript code from the page at `http://web.mit.edu/6.033` can access the DOM of another page at `http://www.google.com` loaded in the same browser. Assume the second page runs no Javascript code.

Answer: False. The hosts are different, so this is not permitted.

D. True / False Suppose the 6.033 staff copies the Javascript code for an advertisement from `attacker.com` into the page at `http://web.mit.edu/6.033` (for example, by including a tag like <SCRIPT SRC="http://attacker.com/ad.js">). The Same-Origin Policy will allow that Javascript code from attacker.com to access the DOM of the 6.033 page (`http://web.mit.edu/6.033`).

Answer: True. This is permitted because the 6.033 page explicitely requested the attacker's code.

E. True / False The Same-Origin Policy allows Javascript code from `site1.com` to access the cookie of `site2.com`, as long as the two sites are concurrently loaded in two different tabs in the same window.

Answer: False. The hosts are different, so this is not permitted

Initials:

III Worm Trouble

It is May 2011 and there is a new worm, *nitty*, released on the Internet. Security experts quickly analyze the code corresponding to Nitty to be as follows:

```
1  int X = 0; /* X is a global variable */
2
3  int rand() {
4    /* Note that 32-bit integers obviate the need for a modulus operation here. */
5    /* A = 214013, B = 2531011. */
6    X = X * A + B;
7    return X;
8  }
9
10 srand(seed) {
11   X = seed;
12 }
13
14 main() {
15   srand(get_tick_count());
16   while (1) {        /* infinite loop */
17     for (int i = 0; i < 20,000; ++i) {
18       dest_ip = rand();
19       dest_port = rand() [0:15];      /* low 16 bits */
20       packetsize = 1000;              /* 1000 bytes */
21       packetcontents = top_of_stack;
22       sendto();
23     }
24     int rand_temp = rand();
25     if (open(physicaldisk, rand_temp[13:15]))
26       overwrite_block(rand_temp[0:14] || 0x4e20);
27   }
28 }
```

You learned about network telescopes in 6.033 and decided to see if you can apply the concepts to infer information about this new worm. You set up a router at MIT and configure it to listen on two unused portions of the MIT IP address space corresponding to 18.32.0.0/11 and 18.96.0.0/11. This worm appears to differ from the Witty worm in lines 16 (always loop back to the same place), 18, 20 (always choose the same packet size), and 24–26.

8. [4 points]: What fraction of the total 32 bit IP address space does your network telescope listen on?

Answer: Each unused /11 portion corresponds to $1/2^{11}$ of the 32 bit IP address space. So the combined fraction from the two ranges is $1/2^{10}$ or 2^{-10}.

Initials:

9. [8 points]: Say your network telescope receives two packets from an infected node with the following parameters: (dest_ip_1, dest_port_1) and (dest_ip_2, dest_port_2) respectively. Given the two received packets, how would you identify whether they are likely to be from a single for loop execution?

Answer: There are two calls to rand() in each for loop iteration, and we know that either dest_ip_1 or dest_ip_2 is equal to X when the first packet is sent. We can thus use dest_ip_1 as a seed and repeatedly call rand() to reach dest_ip_2, or try dest_ip_2 as a seed and repeatedly call rand() to reach dest_ip_1. In either case, if the number of applications of rand() is even and less than 40000, then the two packets are from the same loop execution.

10. [6 points]: Finally, you would like to measure the bandwidth of the infected node. To do this you look at two packets, from a single for loop execution, you received at your network telescope at times T_1 and T_2. The two packets had the following parameters: (dest_ip_1, dest_port_1) and (dest_ip_2, dest_port_2). Assume the packet header size is 100 bytes, how would you measure the bandwidth of the infected node?

Answer: As before, start with dest_ip_1 as seed and apply rand() until you get dest_ip_2, or start with dest_ip_2 as seed and apply rand() until you get dest_ip_1. In either case, say the number of rand() invocations needed is C. Then, the number of packets sent between the two packets is $C/2$. The size of each packet is 100 bytes (packet header) + 1000 bytes (packet size from source code) = 1100 bytes. The time in which this happens is $(T_2 - T_1)$.

Thus, the bandwidth of the infected node is $\frac{550C}{T_2 - T_1}$ bytes/second.

Initials:

IV Two-Phase Commit

Consider the two-phase commit protocol for executing atomic transactions across multiple sites. This protocol is described in Sections 9.6.2 and 9.6.3 of the text, as well as in lecture, and is summarized below

One node is designated as the *coordinator*, and the other nodes are designated as *workers*. Under normal circumstances, the protocol proceeds as follows:

- The coordinator sends a `PREPARE` message to each worker.
- The worker responds by writing a `PREPARED` record to its log, then sending a `VOTE COMMIT` message to the coordinator.
- After receiving `VOTE COMMIT` responses from all workers, the coordinator writes a `COMMITTED` record to disk and sends a `COMMIT` message to each worker.
- When a worker receives this message, it writes a `COMMITTED` record to its log.

11. [8 points]: Which of the following statements about this protocol are true? Assume that none of the components (network, coordinator, and workers) crash permanently.

(Circle True or False for each choice.)

A. True / False If any worker commits a transaction (by writing `COMMITTED` to its log), all other workers will also eventually commit that transaction.

Answer: True. This is the goal of two-phase commit. If a worker committed the transaction, that means the coordinator sent it a `COMMIT` message and recorded the transaction commit in the log. Even if there are failures, a worker that queries the coordinator will find that the transaction committed, and commit its part of the transaction.

B. True / False It guarantees that all workers will commit the transaction within a fixed time.

Answer: False. The two-generals problem says that this is impossible. If there are failures, it might take arbitrarily long to complete the protocol.

C. True / False If a worker is using two-phase locking to guarantee before-or-after atomicity, it cannot release the locks held by a transaction until after it receives the `COMMIT` message from the coordinator.

Answer: True. If the worker releases its locks, that makes the effects of the transaction visible to other concurrent transactions. It can't do this until receiving the `COMMIT` message, because until then the transaction might still abort.

D. True / False If the coordinator fails to receive a `VOTE COMMIT` message from some worker in a timely manner, it can decide to abort the transaction.

Answer: True. The coordinator can unilaterally abort the transaction until it writes the `COMMITTED` record to its log. This is fine, because all of the workers are in the tentative commit state and can still be aborted.

Initials:

12. [4 points]: What is the earliest point after which the transaction is guaranteed to commit? Again, assume that none of the components (network, coordinator, and workers) crash permanently.

(Circle the BEST answer)

- After the coordinator sends `PREPARE` messages
- After the last worker sends its `VOTE` message
- **After the coordinator writes the `COMMITTED` record to its log.**
- After all workers have written the `COMMITTED` record to their logs

Answer: The third choice is correct. After the coordinator writes `COMMITTED` record to its log, the decision to commit will survive even if the coordinator crashes. If any of the workers crash, they will resend their `VOTE COMMIT` message when they recover, and learn from the coordinator that the transaction committed.

The second choice is incorrect because the coordinator can still abort the transaction. For example, it might do so if one of the `VOTE` messages is dropped by the network.

Initials:

V Consistency/Replication

The software that runs the retail web site for amazon.com uses an eventually consistent key-value storage system called Dynamo. The basic operations in such a system are to read and write the value for a particular key. For example, a customer's shopping cart is a single object, with the customer's user ID as its key and the set of items in the cart as the value. Versions are labeled by vector clocks, and an update overwrites any earlier versions with smaller clock labels. For simplicity, assume that a set operation contacts only one server, and that server then propagates the new value to the other replicas asynchronously. Each set operation can choose a different server (e.g., if it believes the last server it talked to may have failed or is otherwise unresponsive).

If nodes fail and recover, or respond slowly, it's possible to have two or more versions with clock labels that are not totally ordered; in this case it's the application's job to reconcile them when it reads the cart from Dynamo. In case of doubt the cart application reconciles by keeping any items whose status is in doubt.

13. [8 points]:. Suppose the cart is empty and the user adds item A to the cart, then adds item B, then deletes A. What are the possible contents of the cart that a subsequent read could see?

(Circle True or False for each choice.)

A. True / False {A}

B. True / False {B}

C. True / False {A, B}

D. True / False {}

Answer: All are true. In normal operation the sequence of values will be v1:{A}, v2:{A, B}, v3:{B}, with $v1 < v2 < v3$, and the result will be v3: {B}. If failure makes all these values inaccessible, the result will be {}. If v2 and v3 are written to different servers and v3's server becomes inaccessible, the result will be {A, B}. Or A and B could be written to incomparable versions, leading to versions v1:{A} and v2: {B}. A read that sees both will return {A, B}.

14. [12 points]: With the same sequence of operations, what values could a read see after the system reaches eventual consistency, and no more updates need to be propagated, if there are no other writes to the cart?

(Circle True or False for each choice.)

A. True / False {A}

B. True / False {B}

C. True / False {A, B}

D. True / False {}

Answer: B and C are true. The latter can happen if A and B are written in order, leading to v1:{A}, v2:{A, B}, with $v1 < v2$, and then the delete goes to v1, leading to v2:{A, B}, v3:{} with $v2 \neq v3$. Now after reconciling we end up with {A, B}.

Initials:

15. [7 points]: It's possible to avoid these anomalies completely and have a strongly consistent (single-copy) storage system by running a replicated state machine, using a consensus protocol such as Paxos, to do the read and write operations. The disadvantages of this method, which presumably caused Amazon to reject it in favor of eventual consistency, include which of the following?

(Circle True or False for each choice.)

A. True / False You can't do writes without access to a majority of the replicas.

B. True / False You can't do reads without access to a majority of the replicas.

C. True / False It's more complicated for the application to deal with.

D. True / False It requires the application to contact several servers directly.

E. True / False The latency for both read and write operations is greater.

Answer: A, B, and E are true. B is true because otherwise you might read an older value than was written by the most recent write (the primary can take out a lease, a lock with a timeout, to ensure that it remains the primary for some length of time, and then it can handle reads unilaterally). E is true because the need to contact a majority makes the latencies greater. Not C; single-copy is simpler for the application. Not D; the application can contact one server, usually called the primary, which must talk to a majority to carry out the operation.

End of Quiz

Please double check that you wrote your name on the front of the quiz,
and circled your recitation section number.

Initials: