DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.033—Computer System Engineering Handout 28—April 25, 1995

Quiz #2 Solutions

169 students took the quiz.
Median score = 69
Mean score = 68.1
Std. deviation = 12.8

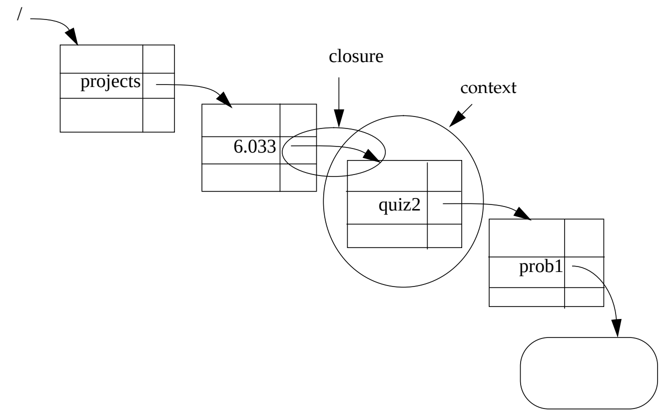
Score Frequency
31-35 ** **(2)**
36-40 * **(1)**
41-45 ******* **(7)**
46-50 ****** **(6)**
51-55 *************** **(15)**
56-60 ************ **(12)**
61-65 ********************* **(21)**
66-70 *************************** **(27)**
71-75 **************************** **(28)**
76-80 ************************** **(26)**
81-85 ********** **(10)**
86-90 ********* **(9)**
91-95 ***** **(5)**
96-100

For Official Use Only

Problem	Your Score	Maximum Score
1		10
2		25
3		25
4		15
5		25
Total		100

Name: ______________________________

1. *(10 points)* Consider the UNIX file system naming hierarchy illustrated below:



You have been handed the following path name:

closure → /projects/6.033/quiz2/prob1

and you are about to resolve the third component of that path name, the name “quiz2”.

1a. Circle the closure, and write the word “closure” next to the thing you have circled.

The closure identifies the context that one should use to resolve “quiz2”. There are two equally valid things in the picture that identify that context to the name interpreter:

a) the pathname “/projects/6.033/”
b) the inode number (shown as a pointer) that is associated with the name “6.033” in the superior directory.

1b. Circle the context, and write the word “context” next to the thing you have circled.

A context is a set of bindings of names to objects. In this problem the context for a single component of a path name is a single UNIX directory. The context that one should use is the one identified by the closure, the third box from the left in the picture.

Name: ______________________________

2. *(25 points)* To obtain high throughput, communication applications pipeline messages: after sending the first packet of a message, they immediately send the next one without waiting for an acknowledgement of the first packet. A key question is how to establish how many packets to pipeline between a request and its acknowledgement (i.e., what is the window size).

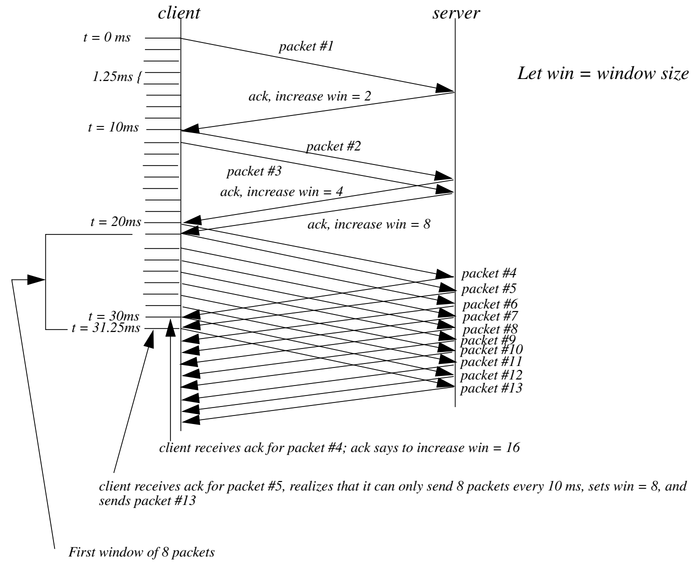
2a. *(5 points)* Assume the client and the server are connected by an 800,000 byte/sec link. Also assume that the client and the server produce and consume packets at the same rate. Using the server's acknowledgements, the client measures the round-trip between itself and the server to be 10 milliseconds. If the client is sending 1000-byte packets, what is the smallest window size that allows the client to achieve 800,000 byte/sec throughput?

In order to achieve 800,000 byte/sec throughput, the client must send 800 packets per second, or 8 packets per 10 ms. If the round trip time is 10 ms, then the client must send 8 packets before it receives an acknowledgement for any of them, making the window size 8 packets.

Alternatively, on an 800,000 byte/sec link the client takes 1.25 msec to send one 1000 byte packet. Thus, the client can send 8 packets in 10 msec and saturate the link bandwidth.

Name: ______________________________

2b. *(5 points)* A popular scheme for establishing the window size in the Internet is *slow start*. In a simple version of slow start, a client starts with a small window size and increases the window size. Assume that the client starts with a window size of one packet. Also assume that for every packet, the server responds with an acknowledgement telling the client to double the window size. How long would it take for the client to know it has reached the maximum throughput?



The client knows it has reached maximum throughput after 31.25 ms. See above figure.

Name: ______________________________

2c. *(5 points)* Another scheme for establishing the window size is *fast start*. In fast start, a client starts by assuming a default large window size (say 8 packets) and sends the server a burst of that size. On the server's response, the window is adjusted to the size that is supplied by the server. If a client starts with a window size of 8, how long does it take for the client to know it has achieved the maximum throughput?

In this case, the client is guessing a window size that is exactly the maximum. The client will know that this guess is the maximum because he will spend 10 ms transmitting 8 packets and receive the acknowledgement for the first packet right at the end of this 10 ms. Thus, the client knows that he will never be able to transmit more than 8 packets in 10 ms, and so he knows that he has achieved the maximum throughput (8 packets per 10 ms, or 800,000 bytes/sec) after only 10 ms.

2d. *(10 points)* In the circumstances described above, fast start seems to be preferable. In what circumstances would it be preferable to use slow start over fast start? You may assume that one of the back-pressure techniques used in the Internet is dropping packets.

One situation where slow start might be preferable is when there are multiple clients but only a single server and the server applies back pressure by dropping packets that it can't handle. With fast start, the clients might all send large bursts of packets, and the server may drop many of them. This will cause the clients to timeout and resend the lost packets. (If the clients do not slow down after this happens then this cycle of dropping packets, timing out, and resending may occur repeatedly.) It will take a certain amount of time to transmit, lose packets, timeout, and resend before the optimal window size is found. It may very well be that slow start would find the optimal window size much more quickly and hence have better performance.

There are other reasons why slow start might be preferable. In general, slow start will cause fewer packets to be dropped than fast start. (However, there is still no guarantee that slow start won't cause any packets to be dropped!) This reason in itself is not sufficient, though; there has to be a reason why dropping packets is bad. Dropping packets can be bad because there is no reason to think that the dropped packets are always from the new client's connection; a fast start may cause the network or server to drop packets from existing connections, forcing those connections to resize their windows.

Name: ________________________________

3. *(25 points)* Secure Inc. is developing an improved version of AFS, Secure AFS (SAFS), which automatically encrypts files to guarantee better privacy of information. When a request to store a file arrives, SAFS enciphers the file using the client's key. On arrival of a request to read a file, SAFS looks up the client key, deciphers the file, and sends the file back to the client. SAFS keeps for each client a separate key to encipher and decipher files.

3a. *(5 points)* The designers of Secure Inc. are wondering how long it would take to crack an enciphered file that is encrypted using RSA with 512-bit key, which is what is typically used today. To crack an RSA-enciphered file one has to factor the key. The designers know from the literature that factoring a 100 decimal digit number takes about 1 month using idle cycles from 300 3-MIPS SUN-3 workstations. It is estimated that factoring an additional 3 decimal digits roughly doubles the computation time needed. How many 3-MIPS computers are needed to factor a 155 decimal digit number (which corresponds to about 512 bits) in one month?

100-digit number ==> 300 workstations
Difficulty doubles with every three digits.

155-digit number ==> 55/3 doublings ==> $300 * 2^{55/3}$ *workstations*
This is about 99 million workstations.

3b. *(5 points)* If processors are doubling computation performance per year, how many computers would it take to factor a 512-bit key in one month in the year 2000?

Performance doubles every year.
2000 - 1995 = 5 years ==> 5 doublings
So, dividing the answer from part (a) by 2^5*, we now need*

$300 * 2^{40/3}$ *workstations, i.e., about 3 million workstations.*

3c. *(5 points)* Let's assume that enciphering and deciphering can be done at 250 Kbyte/sec. How much would the throughput be reduced for reading files stored by SAFS, if the current maximum throughput without enciphering/deciphering is 800 Kbyte/sec? (You may assume that enciphering/deciphering cannot be pipelined with sending/receiving.)

Time per unit data = time to decrypt + time to send
= 1/250 sec/KB + 1/800 sec/KB = 21/4000 sec/KB
The new throughput is the inverse of this number, i.e., data per unit time:
4000/21 KB/sec = 190.4 KB/sec
The problem asked for the reduction in throughput, which is
Old throughput - new throughput = 800 KB/sec - 190.4 KB/sec = 609.6 KB/sec
This is 609.6/800 = 76% of 800 KB/sec or a reduction of (800-609.6)/800 = 24%.

Name: ________________________________

3d. *(10 points)* Secure Inc. is also considering adding automatic compression of files to SAFS. Compression reduces redundancy of information in a file so that the file takes less disk space. Should they first compress files, then encipher them, or should they first encipher files and then compress them? Explain your answer briefly.

It is better to compress a file first, then encipher it. There are several reasons for this. Among them are the following, ordered strongest to weakest in our estimation.

1. Effective compression depends on eliminating redundancy (e.g., patterns of character frequency) in the file being compressed. However, effective encryption eliminates patterns in order to disguise content. Therefore, if the file were enciphered first, little or no compression would be achieved.

2. Compression reduces the size of the file, whereas encryption (we assume) does not. Assuming both operations take time proportional to the length of the source being operated on, the greatest total efficiency is achieved by compressing the file to create a smaller source for the encryption operation.

3. If the file is compressed before it is encrypted, it is harder for an attacker guessing a key to realize he has correctly decrypted the file, as the decrypted file still looks pretty random. Even if he knows the decompression algorithm to apply, it still slightly increases the work factor for cracking the encryption.

4. *(15 points)* Baybank is struggling to convince itself of the validity of a message it just received, and has asked your help in what to do next. So far, they know the following two facts to be true:

Louis says (Ben says (Transfer $1,000,000 to Alyssa))

Jim speaks for Ben

According to Ben's account representative, Ben's account has enough money for such a transaction, so if they can convince themselves that Ben really said that, they will do the transfer.

4a. *(5 points)* Which of the following things should they attempt to establish the truth of?

a. Louis speaks for Jim

b. Ben speaks for Louis

c. Ben says (Jim speaks for Louis)

4b. *(10 points)* Why?

Choice (a) allows Baybank to establish the following, two-step line of reasoning:

1. Jim speaks for Ben [given]
Louis speaks for Jim [choice a]

Reasoning rule #2 (chaining of delegation) says that
(A speaks for B) and (B speaks for C) implies (A speaks for C)

Applying this rule tells Baybank that (Louis speaks for Ben)

2. Louis says (Ben says (Transfer $1,000,000 to Alyssa)) [given]
Louis speaks for Ben [established in step 1]

Reasoning rule #1 (delegation of authority) says that
(A says (B says X)) and (A speaks for B) implies (B says X)

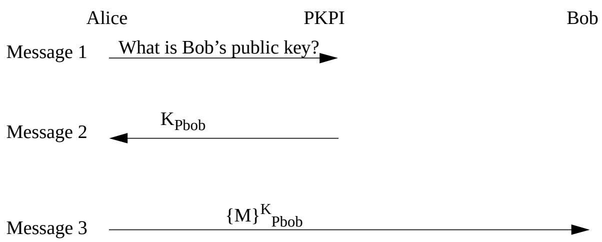
so if step 1 is sound, Baybank can be sure that

Ben says (Transfer $1,000,000 to Alyssa)

which is what Baybank needs to know before it will perform the transfer.
Thus Baybank should attempt to establish the truth of choice (a).

5. *(25 points)* Ben Bitdiddle and Louis Reasoner have founded a startup company, named Public Key Publication, Inc. (PKPI), whose business is distributing public keys. Their idea is that people who have a key-pair for use with a public-key system need a way of letting other people know the public key of their key-pair. Ben and Louis are not interested in creating keys, but just in acting as a public key distributor.

Ben and Louis have designed the following protocol, in which Alice sends a private message to Bob. They need your help in debugging the protocol. K_{Pxyz} is the public key of principal xyz.



Messages 1 and 2 constitute the PKPI protocol; message 3 is the beginning of Alice's protocol with Bob and is not under the control of PKPI; message 3 is shown here only to place the PKPI protocol in context.

5a. *(5 points)* Louis believes that Eve, the passive eavesdropper, will find that she cannot learn anything by overhearing the PKPI protocol in use. Give an argument that supports Louis' position, or an example demonstrating that Louis is mistaken.

Louis is almost correct. Neither Message 1 nor Message 2 contains any secrets, so passive eavesdropping won't reveal anything. Eve may, however learn something from traffic analysis: that Alice has asked for Bob's public key. If Alice had hoped to keep that fact secret, this was not a very good protocol to use.

5b. *(10 points)* Louis originally hoped that Lucifer, the active attacker, wouldn't be able to cause any problems, either, but since hearing the 6.033 lecture on the topic he is not sure. Give an example of an active attack that demonstrates that Louis needs to revise the PKPI protocol to protect against Lucifer.

Lucifer can actively attack the protocol in several ways:
1) He can intercept and replace Message 1 with, "What is Lucifer's public key?" which will cause Message 3 to be encoded with Lucifer's public key. He will then be able to decrypt Message 3.
2) He can intercept and replace Message 2 with Lucifer's public key instead of Bob's public key, which will cause Message 3 to be encoded with Lucifer's public key. He will then be able to decrypt Message 3.

5c. *(10 points)* Ben suggests that the protocol could be improved by changing Message 2. What should Message 2 contain, and how should it be enciphered, so that Alice can be confident that no one but Bob can decipher message 3?

The protocol lacks two things:
1) Authentication: Alice needs proof that the message she is receiving is from PKPI and not from Lucifer. This can be accomplished by encoding Message 2 in the secret key of PKPI, K_{Spki}*, which anyone can decrypt but only PKPI could have constructed.*
2) Connection with Bob's key: Alice needs proof that the key she is receiving in Message 2 is Bob's, not Lucifer's. This proof can be accomplished by including the name "Bob" inside the now-encrypted Message 2: $\{Bob, K_{Pbob}\}^{K_{Spkpi}}$.

Note that if public keys never change, replays aren't of concern--even if Lucifer intercepts and replaces Message 2 with a month-old copy of Message 2, as long as that stale message was encrypted by PKPI and contains Bob's name it is OK.