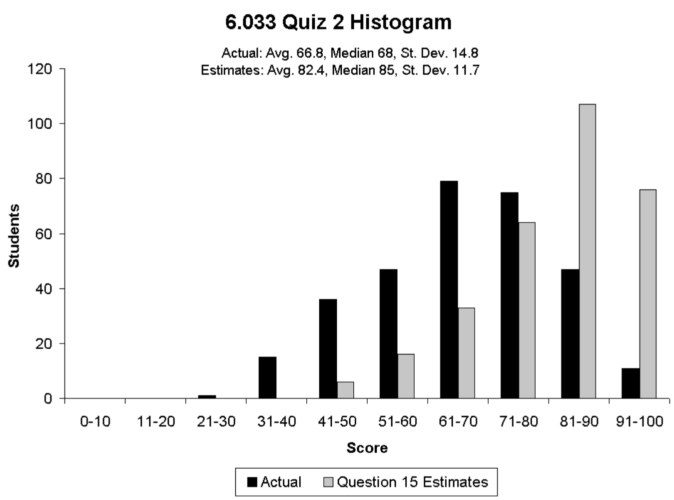
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.033 Spring, 1998 Handout 23: April 17, 1998

6.033 Computer System Engineering: Spring, 1998

Quiz 2



Do not write in the boxes below

1-6 (XX/36)	7-14 (XX/63)	SubTotal (XX/99)	15 (XX/1)	Total (XX/100)

Name: Official Solutions

I. Any of this sound familiar?

1. [6 points]: The best example of an *end-to-end argument*, as Saltzer *et al.* uses the term, is:

A. If you laid all the web hackers in the world end to end, they would reach from Cambridge to CERN.

B. Every byte going into the write end of a UNIX pipe eventually emerges from the pipe's read end.

C. Even if a chain manufacturer tests each link before assembly, he'd better test the completed chain.

D. Per-packet checksums must be augmented by a parity bit for each byte.

E. All important network communication functions should be moved to the application layer.

(circle letter to left of best answer)

2. [6 points]: Ethernet cards have unique Ethernet addresses built into them. What role do these play in an IP network?

A. None. They are there for Macintosh compatibility only.

B. A portion of the Ethernet address is used as the IP address of the computer using the card.

C. They provide routing information for packets destined to non-local subnets.

D. They are used as private keys in the Security Layer of the ISO protocol.

E. They provide addressing within each subnet for an IP address resolution protocol.

F. They provide secure identification for warranty service.

(circle letter to left of best answer)

3. [6 points]: A UNIX directory is a file comprising a sequence of *directory entries*, each of which contains

A. The ASCII name and logical file number of a file.

B. A list of physical disk blocks occupied by a file.

C. Permission information for a file for the owner, for members of the owner's group, and for all users.

D. Time and date of last modification of a file.

E. All of the above.

F. None of the above.

(circle letter to left of best answer)

Name: Official Solutions

4. [6 points]: To client software, a notable difference between Birrell's RPC and ordinary local procedure calls is:

A. None. That's the whole point of RPC!
B. There may be multiple returns from one RPC call.
C. There may be multiple calls for one RPC return.
D. Recursion doesn't work in RPC.
E. The runtime system may raise a new type of exception as a result of an RPC.
F. Arguments to RPCs must be scalars.

(circle letter to left of best answer)

5. [6 points]: When an Ethernet station detects interference while transmitting, it

A. Completes the transmission of the packet normally.
B. Immediately stops transmitting, and retransmits the packet later.
C. Records the address of the interfering station, and sends it a subsequent *interference notification* packet.
D. Jams the Ethernet briefly.
E. Sends Email to Janet Reno.
F. Sets the *possibly corrupt* bit at the very end of the packet.
G. None of the above.

(circle letter to left of best answer)

6. [6 points]: Which, if any, of the following "flaws" in UNIX software were exploited by the internet worm? **Circle all answers that apply:**

A. Unchecked array bounds allowed buffer overruns.
B. Unsafe debugging commands were available to untrusted traffic.
C. At many sites the password file was still encrypted with the default key.
D. "Trusted" machines could bypass authorization protocols.
E. Unencrypted passwords were sent over networks where malicious software could capture them.

(circle letter to left of ALL THAT APPLY)

Name: Official Solutions

II. JailNet

The Computer Crimes Correction Facility, a federal prison for perpetrators of information-related crimes, has observed curious behavior among their inmates. Prisoners have discovered that they can broadcast arbitrary binary strings to each other by banging cell bars with either the tops or bottoms of their tin cups, making distinct sounds for "0" and "1". Since such sounds made in any cell can typically be heard in every other cell, they have devised an Ethernet-like scheme for communicating varying-length packets among themselves.

The basic communication scheme was devised by Annette Laire, a CCCF lifer convicted of illegal exportation of restricted information when the GIF she emailed to her cousin in El Salvador was found to have some bits in common with a competent encryption algorithm.

Annette defined the basic communication primitive

```
Send(char *message, char *from, char *to)
{
  Bang(AllOnes);             /* Start with a byte of 8 1's */
  Bang(to);                  /* destination inmate number */
  Bang(from);                /* source inmate number */
  Bang(message);             /* the message data */
  Bang(Checksum(to,          /* Checksum of whole message */
            from, message));
}
```

where the operation `Bang(data)` is executed by banging one's tin cup to signal the sequence of bits corresponding to the specified null-terminated character string, including the zero byte at its end. The special string `AllOnes` sent initially has a single byte of (eight) 1 bits (followed by the terminating null byte). The high-order bit of each 8-bit character (in `to`, `from`, `message`, and `Checksum`) is zero.

Annette specified that the `to` and `from` strings be the unique numbers printed on every inmate's uniform, since all of the nerd-inmates quickly learn the numbers of each of their colleagues. Each inmate listens more or less continuously for packets addressed to him, ignoring those whose `to` field don't match his number or whose checksums are invalid.

7. [11 points]: What function(s) are served by sending the initial byte of all 1s? **Circle all answers that apply:**

A. Bit framing.
B. Byte (character) framing.
C. Packet framing.
D. Packet Reassembly.
E. None of the above.

(circle letter to left of ALL THAT APPLY)

Name: Official Solutions

Typical higher-level protocols involve sequences of packets exchanged between inmates, e.g.:

Annette: `Send`("I thought the lobster bisque was good tonight", `Annette`, `Ty`);

Ty: `Send`("Yes, but the filet was a bit underdone for my tastes", `Ty`, `Annette`);

where the symbols `Annette` and `Ty` are bound to character strings containing the uniform numbers of Annette and Ty, respectively.

Of course, prison guards quickly catch on to the communication scheme, and even inject messages of their own:

Guard: `Send`("Oh yeah? Then it's ALPO for you tomorrow!",
`JimmieTheGeek`, `Annette`);

Such experiences motivate Ty Debole, the inmate in charge of cleaning, to add security measures to the JailNet protocols. Ty reads up on public-key cryptography and decides to use it as the basis for JailNet security. He chooses a public-key algorithm and asks each inmate to generate a public/private key pair and tell him the public key.

- `Key` represents the inmate's public key. Since Ty runs the CCCF laundry, he prints the numbers on inmate's uniforms. He replaces each inmate's assigned number with a representation of `Key`;
- `$Key` is the inmate's secret key. This key is known only to the inmate whose uniform bears `Key`.

Ty assures each inmate that so long as they don't reveal their secret `$Key`, nobody else—inmates or guards—will be able to determine it. Inmates continue to address each other by the numbers on their uniforms, which now specify their public `Key`s.

8. [6 points]: What is an assumption on which Ty bases the security of the secret `$Key`?

A. `$Key` is theoretically impossible to compute from `Key`.

B. `$Key` takes an intractibly long time to compute from `Key.`

C. `$Key` takes at least as long to compute from `Key` as the generation of the `Key`, `$Key` pair.

D. There is a reasonably efficient way to compute `$Key`, but it's not generally known by guards and inmates.

E. None of the above.

(circle letter to left of best answer)

Ty then teaches inmates 4 cryptographic algorithms for messages of up to 1,500 bytes.

- `Encrypt(message, Key)` returns a cyphertext string
- `Decrypt(cyphertext, $Key)` returns a message string
- `Sign(message, $Key)` returns a signature string
- `Verify(message, Key, signature)` returns true or false

These functions have the following properties true for every choice of `Key` and `message`:

- All 4 functions are efficiently computable given the appropriate arguments.
- It is computationally infeasible to derive any information about `message` from `Encrypt(message, Key)` without knowing `$Key`.
- Without `$Key`, it is computationally infeasible to create a `signature` such that there exists a `message` on which `Verify(message, Key, signature) = true`.
- `Decrypt(Encrypt(message, Key), $Key) = message`.
- `Verify(message, Key, signature)` verifies a digital signature. It returns true if and only if `signature` is a result of `Sign(message, $Key)`; otherwise it returns false.

Ty proposes improving the security of communications by replacing calls to Send by calls like

```
Send(TyCode(message, from, to), from, to);
```

where `TyCode` is defined as

```
char *TyCode(char *message, char *from, char *to)
  {  return Encrypt(message, to); }
```

Ty and Annette are smugly confident that although the guards might hear their conversation, they won't be able to understand it since the encrypted message appears as gibberish until properly decoded.

The first use of `TyCode` involves the following message, received by Annette:

`Send`(`TyCode`("Meet me by the wall at ten for the escape",`Ty`, `Annette`),
`Ty`, `Annette`);

9. [6 points]: What computation did Annette perform to decode Ty's message? Assume **`rmessage`** is the message as received, **`message`** is to be the decoded plaintext, and that **`$Annette`** and **`$Ty`** contain the secret keys of Annette and Ty, respectively.

A. **`message = Verify(rmessage, Ty, $Annette);`**
B. **`message = Encrypt(rmessage, $Ty);`**
C. **`message = Encrypt(rmessage, Ty);`**
D. **`message = Decrypt(rmessage, $Annette);`**
E. **`message = Sign(rmessage, $Ty);`**
F. **`message = Decrypt(rmessage, Annette)`**;
G. None of the above

(circle letter to left of best answer)

After receiving the message, Annette sneaks out at ten to meet Ty who she expects will help her climb over the prison wall. Unfortunately Ty never shows up, and Annette gets caught by a giggling guard and is punished severely (early bed, no dessert). When she talks to Ty the next day, she learns that he never sent the message; she expects that it was sent by a guard, but is puzzled since the cryptography is secure.

10. [6 points]: What is the most likely explanation?

A. Annette's **`$Key`** was compromised during a search of her cell.
B. Some other message Ty sent was garbled in transmission, and accidentally came out "Meet me by the wall at ten for the escape".
C. Annette's secret key was broken by a dictionary attack.
D. Ty's secret key was broken by a dictionary attack.
E. Annette was victimized by a replay attack.
F. None of the above.

(circle letter to left of best answer)

Name: Official Solutions

Annette's friend Cert Defy, on hearing this story, comes up with a new cryptographic function. He defines the procedure

```
char *Cert(char *message, char *A)
  { return Append(message, Sign(message, A)); }
```

Here the **`Append`** operation concatenates its arguments with recognizable separators so that the result can easily be parsed into its original components. Unfortunately, Cert is placed in Solitary Confinement before fully explaining how to use this function, though he did state that sending a message with

`Send(Cert(message, A), from, to)`

can assure the receiver of the integrity of the message body and the authenticity of the sender's identity.

11. [6 points]: When Ty sends a message to Annette what value should he supply for **`A`**?

A. **`Encrypt(Annette, $Ty)`**
B. **`Ty`**
C. **`$Ty`**
D. **`Annette`**
E. **`$Annette`**
F. None of the above

(circle letter to left of best answer)

After determining the answer to question 11, Annette receives a packet purportedly from Ty. She splits the received packet into **`message`** and **`signature`**. **`Verify(message, Ty, signature)`** returns true.

12. [11 points]: Which of the following can Annette conclude about **`message`**?

A. **`message`** was initially sent by Ty.
B. The packet was sent by Ty.
C. **`message`** was initially sent to Annette.
D. Only Annette and Ty know the contents of **`message`**.
E. If Ty sent **`message`** to Annette and Annette only, then only they know its contents.
F. **`message`** was not corrupted in transmission.

(circle letter to left of ALL THAT APPLY)

Name: Official Solutions

Annette, intrigued by Cert's contribution, decides to combine **Send**, **TyCode**, and **Cert** to achieve both authentication and confidentiality. She proposes to use **NewSend**, combining both features:

```
NewSend(char *message, char *A, char *from, char *to)
{ Send(TyCode(Cert(message, A), from, to), from, to); }
```

Annette engages in the following conversation:

Ty: **NewSend**("Let's escape tonight at ten", **Ty**, **Annette**);
Annette: **NewSend**("Not tonight, Seinfeld is on", **Annette**, **Ty**);

The following night, Annette again gets the message

Ty: **NewSend**("Let's escape tonight at ten", **Ty**, **Annette**);

Once again Annette goes to meet Ty at ten, but Ty never shows up. Eventually Annette gets bored and returns. Ty subsequently disclaims having sent the message. Again, Annette is puzzled by the failure of her allegedly secure system.

13. [6 points]: What is the most likely explanation?

A. A guard has discovered a general way to determine **Key** from **$Key**, and is exploiting this at Annette's expense.

B. Some other message Ty sent was garbled in transmission, and accidentally came out "Let's escape tonight at ten".

C. Annette's secret key was broken by a dictionary attack.

D. Ty's secret key was broken by a dictionary attack.

E. Annette was victimized by a replay attack.

F. None of the above.

(circle letter to left of best answer)

Name: Official Solutions

Pete O'Fender, who has been in and out of CCCF at regular intervals, wants to extend the security protocols to deal with JailNet key distribution. Whenever he's jailed, Pete is placed directly into Solitary Confinement where he has no contact with inmates (except via bar banging), and where the TV gets only 3 channels. The problem is complicated by the facts that (a) Everyone (including Pete) forgets Pete's uniform number as soon as he leaves, so when he returns he can't just re-use the old key; (b) Pete may not even remember the key for Ty or other trusted long-term inmates; (c) Pete is issued an unnumbered uniform while in Solitary, and (d) guards often pose as newly-jailed Solitary occupants to learn inmate secrets. Pete asks you to devise JailNet key distribution protocols to address these problems.

14. [11 points]: Which of the following are true of the *best* protocol you can devise, given the assumptions stated about Encrypt, Decrypt, Sign, and Verify? **Circle all answers that apply:**

A. Assuming Pete is thrust into Solitary remembering no keys, he can devise a new **Key**/**$Key** pair and broadcast **Key**. Using this **Key**, Ty can be assured that messages he sends to Pete are confidential.

B. Assuming Pete is thrust into Solitary remembering no **Key**s, he can't convince inmates that they aren't communicating with a guard.

C. If Pete remembers Ty's uniform number and trusts Ty, an authenticated broadcast message from Ty could be used to remind Pete of other inmates' uniform numbers without danger of deluding Pete.

D. Even if Pete remembers a trusted inmate's uniform number, any communication *from* Pete can be understood by guards.

E. Even if Pete remembers a trusted inmate's uniform number, any communication *to* Pete might have been forged by guards.

(circle letter to left of ALL THAT APPLY)

Name: Official Solutions

III. Final Jeopardy

15. [1 point]: Give your best guess as to your grade on this quiz, *outside of this last problem*. Note that the maximum score on these earlier problems is 99; you get credit for this problem if your guess is within 3 points of your grade on those.

Your estimate of Quiz 2 Score: ___________________

```
99 (precise)
96 (conservative)
102 (show-off)
```

END OF QUIZ 2

(some day you'll look back on all this fondly...)