Histogram of exam 1 scores

Score:	0–4	5–9	10–14	14–19	20–24	25–29	30–34	35–39	40–44	45–50
# students:	2	5	3	6	4	6	7	5	1	0

Exam 1 is very similar to practice exam 1. It's hard to see how someone who studied the practice exam, let alone did it under exam conditions, could score less than 10 points.

My scoring is harsh but my cutoffs are low:
A ≥ 32; B ≥ 25; C ≥ 16; D ≥ 10

Why did I lose points

An attack or an explanation (or argument or proof) should not be a puzzle that I have to augment to make sense of. If I encounter a step that is not justified based on what is written so far (e.g., the common mistakes in the solutions below), I usually do not attempt to fix the hole and try to make sense. (Doing so would unfairly penalize students who leave their attacks or explanations incomplete because they saw the hole but did not know how to fix it.)

When explaining why a property, e.g., an assertion *Inv P*, holds for program, one common mistake is to consider only a particular case. For example, the following is not valid:

> *Inv P* would not hold if the attacker can generate response `enc(K,nA)` for challenge `nA`. But it cannot do this because it does not have key `K`. So *Inv P* holds.

One very common mistake you make is to assume that variables in different programs have the same value just because the variables have the same name. For example, suppose client `A` sets `nL` to a random value and sends message `[A,B,nL]`, and server `B` receives a message `[A,B,.]` and saves the last entry of the message in variable `nL`. You often use `nL` to denote the value of `A.nL` and the value of `B.nL`, implicitly assuming that `A.nL` and `B.nL` are always equal (when they may not be) and reaching incorrect conclusions.

Should I drop the class

In addition to exam 1 scores, your current course total (out of 100) using the following weights is posted:

Project 1	2
Project 2	12
Homework 1	0
Homework 2	5
Homework 3	5
Exam 1	25

[The final course total would also have project 2 (11%), homeworks 4, 5, 6 (5% each), and exam 2 (25%). There is no "extra credit" work to make up for a weak exam 1 score.]

If I were to give a course grade today, I'd probably use the following cutoffs:
A ≥ 78; B ≥ 68; C ≥ 58; D ≥ 50

3 problems over 3 pages. 50 points. *Closed book. Closed notes. No calculator or electronic device.*

Problem 1 [20 points]

Part a [10 pts]

Does *Inv* A_1 hold, where

A_1 : ((i in hst.keys) and i > 0 and hst[i] = [B,p]) $\Rightarrow$ hst[i-1] = [A,p]

Solution

Yes.

Attacker cannot obtain K, i.e., *Inv* ψ(K) holds. **[2 pts]**
(Proof: K is not in α initially. Any K-expression (i.e., an expression involving K) that enters α has the form enc(K,x), where x is random or received from the channel or both. The only K-expressions the attacker can add to the channel are ones it has received previously from the channel. So x cannot be dec(K,K) or a simple function of K.)

Suppose B updates hst at time t_0. Hence at t_0: step B.2 receives [A,B,2,enc(K,xB+1)] and appends [B, enc(-K, xB+xA)] to hst, where [xB,xA] is the value of [B.nL,B.nR] just before t_0.

So B's previous step, say at time t_1, is B.1. It receives [A,B,1,xA], sets B.nL to xB, and sends message [B,A,1, enc(K, [xB,xA+1])]. **[2 pts]**

Because xB is random and the attacker does not have K, entry enc(K,xB+1) in the message received at t_0 was generated by A at some time t_2 between t_1 and t_0. This happens only if step A.1 receives message [B,A,1, enc(K, [xB,yA+1])] where yA = A.nL and adds [A, enc(-K, yA+xB)] to hst. **[2 pts]**
Entry 3 of this message is generated by B.1 (because xB is random and the attacker does not have K). Hence yA equals xA (because xB is a new random value each time B sends a [B,A,1,.] message). Hence [A, enc(-K, xA+xB)] is added to hst at t_2. **[2 pts]**

A.1 cannot update hst between t_2 and t_0 (because that would require B.1 to generate a response to the new A.nL value, which does not happen because B is idle during t_1 to t_0). **[2 pts]**

So [A, enc(-K, xA+xB)] remains the last entry in hst just before t_0.

Common mistakes

- Most of you (correctly) inferred that if B receives [A,B,2,enc(K,xB+1)] at t_0 then A sends [A,B,2,enc(K,xB+1)] at t_2. But then most of you (incorrectly) inferred from this that A receives [B,A,1, enc(K, [xB,xA+1])] at t_2; all you can infer from the code (without additional analysis) is that A receives [B,A,1, enc(K, [xB,yA+1])] where yA equals A.nL (but not necessarily xA).

  This mistake effectively skips the arguments corresponding to the last 4 points, and hence costs you 4 points.

Problem 1 [20 pts] (continued)

Part b [10 pts]

Can the attacker obtain K by dictionary attack, assuming that K is a weak key.

Solution

Yes.

Here is an evolution that yields values for a dictionary attack: **[4 pts]**

1. Initial step: add msg [A,B,1,xA] to chan and α, where xA is the value of A.nL.
2. B.1: adds msg [B,A,1,enc(K,[.,xA+1])] to chan and α.

Attacker now has xA and z equal to enc(K,[xB,xA+1]). It can do the following off-line dictionary attack: **[6 pts]**

```
for (?pw in Dictionary) {              // ?pw: candidate password
   ?K ← pwToKeyFunction(?pw);      // ?K: candidate password key
   if (dec(?K,z)[1] = xA+1)
      done;  // K = ?K
}
```

(In fact, a dictionary attack is possible with just message 2; message 1 is not needed. How?)

Common mistakes

- Saying that a dictionary attack is not possible because the attacker does not obtain a ciphertext-plaintext pair.

Problem 2 [10 points]

Can the attacker obtain K by dictionary attack, assuming K is a weak key?

Solution

Yes.

Here is an evolution ending with the attacker obtaining data. **[4 pts]**

1. Initial step
   After: [A,B,1,.] in chan.
2. Attacker:
   remove msg 1; generate random xA; send msg [A,B,1,tA]), where $tA = g^{xA} \bmod p$.
   After: [A,B,1,tA]) in chan.
3. B.1: receive msg 2
   After: [B,A,1,enc(K,tB)enc(L,['HELLO'])] in α, where $L = tA^{xB} \bmod p = g^{xA \cdot xB} \bmod p$.

Attacker now has

- xA // step 2
  tA
- z1 = enc(K,tB) // step 3
  z2 = enc(L,['HELLO']) where $L = g^{xA \cdot xB} \bmod p$.

Attacker can then do the following off-line dictionary attack: **[6 pts]**

```
for (?pw in Dictionary) {          // ?pw: candidate password
   ?K ← pwToKeyFunction(?pw);      // ?K: candidate password key
   ?tB ← dec(?K,z1);               // ?tA: candidate tA
   ?L ← ?tB^xA mod p;
   ?msg ← dec(?L, z2);
   if (?msg.size = 1 and ?msg[0] = 'HELLO')
      done;  // K = ?K, pw = ?pw
}
```

Common mistakes

- Attacker only eavesdrops, obtaining tA, z1 = enc(K,tB), and z2 = enc(L,['HELLO']), but not xA. A dictionary attack is not possible after that. This gets you 2 points in total. (2/4 for the evolution, and 0/6 for the attack.)
- Searching through the Diffie-Helman exponent space, e.g., given tA, search for xA such that $g^{xA} \bmod p$ equals tA. That's hopeless.
- Saying that a dictionary attack is not possible because the attacker does not obtain a ciphertext-plaintext pair.

Problem 3 [20 points]

Part a [12 pts]

Does *Inv* A_1 hold, where

A_1 : ((j in hst.keys) and j > 0 and hst[j] = [B,p]) $\Rightarrow$ hst[j-1] = [A,p]

Solution

Yes.

[2 pts] Attacker cannot obtain kAZ or kBZ; in terms of assertions, *Inv* ψ(kAZ) and *Inv* ψ(kBZ) hold. Proof: similar to that of *Inv* ψ(K) in problem 1a.

[2 pts] Attacker cannot obtain any session key; in terms of assertions,

Inv ((enc(kAZ,[B,p,jkt]) in α) or (enc(kBZ,[p,A]) in α)
or (A.kAB = p) or (B.kAB = p) or ([.,p] in hst))
$\Rightarrow$ ψ(p)

Proof: similar to that of *Inv* ψ(K) in problem 1a.

Suppose B updates hst at time t_0. Let p = B.kAB and nB = B.nL and nA = B.nR just before t_0. Then at t_0: step B.2 receives [A,B,enc(p,nB-1)] and appends [B,p] to hst.

[2 pts] So B's previous step is B.1, say at time t_1. At t_1: step B.1 receives message [A,B,x,y], where x = enc(kBZ,[p,A]) and y = enc(p,nA); sets B.nL to a random value, say nB, and sends message [B,A,enc(p,[nB,nA-1])].

[2 pts] Because xB is random and the attacker does not have p, entry enc(p,[nB,nA-1]) in the message received at t_0 was generated by A at some time t_2 between t_1 and t_0. Hence at t_2: step A.2 receives message [B,A,enc(q,[nB,mA-1])], where q = A.kAB and yA = A.nL; and adds [A,q] to hst.

[2 pts] The received message is the same as the one sent by B.1 at t_1. The message's enc(q,[nB,mA-1]) entry is generated by B.1 (because the attacker does not have q and A does not generate such an encryption). Furthermore, it is the mesage sent by B.1 at t_1 (because nB is chosen randomly by B.1). Hence mA = xA and q = p.

Hence [A,p] was added to hst at t_2.

[2 pts] A cannot update hst between t_2 and t_0 (because that would require B.1 to generate a response to the new A.nL value, which does not happen because B is idle during t_1 to t_0.

So [A,p] remains the last entry in hst just before t_0.

Problem 3 [20 points]

Part b [8 pts]

Does *Inv* A_2 hold, where

A_2 : ((j,k in hst.keys) and j $\neq$ k and hst[j][0] = hst[k][0]) $\Rightarrow$ hst[j][1] $\neq$ hst[k][1]

Solution

No.

This does not hold because A does not include a nonce in its [A,Z,B] message. So when A sends [A,Z,B] to request a session key, the attacker replays the [Z,A,.] message from an earlier request, which makes A use the session key from the earlier request again.

Counter-example evolution:

1. Initial step, Z.1: [Z,A,enc(kAZ,[.,B,jAB,jkt])] in channel and α; jkt = enc(kBZ,[jAB,A]).
2. A.1, B.1, A.2, B.2: receive msg 1 and connect to B with session key jAB; send [A,Z,B].
   After: hst = [[A,jAB], [B,jAB]].
3. Attacker:
   remove msg 2; send [Z,A,enc(kAZ,[.,B,jAB,jkt])] (from step 1).
4. A.1, B.1, A.2, B.2: receive msg 3 and connect to B with session key jAB.
   After: hst = [[A,jAB], [B,jAB], [A,jAB], [B,jAB]].
   A_2 does not hold.