1. [20 points]

Part a. [6 points]

Inv $A_1$ holds, where $A_1 : \psi(K)$

**Solution**

True.

At the start, only A and B have K; it is not in $\alpha$. The only expressions involving K seen by the attacker have the form $\text{enc}(K,x)$, where x is received from the channel. So x can be a value generated by the attacker, but the attacker cannot set x to be a simple function of K or to $\text{dec}(K,K)$. So $\text{enc}(K,B.nA)$ does not expose K.

Part b. [8 points]

Inv $A_2$ holds, where $A_2 : ((i \in \text{hst.keys}) \text{ and } \text{hst}[i] = [B,p]) \implies ([A,p] \text{ in } \text{hst}[0..i-1])$

**Solution**

False.

Counter-example evolution.

1. **Initial step**
   - After: $[A,B,1,2]$ in chan and in $\alpha$; $\text{hst} = []$.
2. Attacker changes msg 1 (i.e., message in step 1) to $[A,B,1,3]$. (Attacker knows that B’s challenge will be 3.)
3. B.1: receive msg 2
   - After: $[B,A,1,3,\text{enc}(K,3)]$ in chan and in $\alpha$; B at 2; B.nL = B.nR = 3; $\text{hst} = []$.
4. Attacker, using $\text{enc}(K,3)$ field in msg 3, sets chan to $[[A,B,2,\text{enc}(K,3)]]$.
5. B.2: receive msg 4
   - After: $\text{hst} = [[B,\text{enc}(-K,6)]]$. $A_1$ not satisfied.

Part c. [6 points]

Attacker cannot learn K by dictionary attack, assuming that K is a weak key.

**Solution**

False

Consider the evolution consisting of the initial step and B.1 (and attacker only eavesdrops).

Then attacker gets:

- 2 // from msg $[A,B,1,2]$ sent in initial step
- $\text{enc}(K,2)$; refer to it as z. // from B’s response msg $[B,A,1,3,\text{enc}(K,2)]$

Attacker can then do the following off-line dictionary attack:

```java
for (cPw in Dictionary) {
    cK = pwToKeyFunction(cPw); // cK: candidate password key
    if ($\text{enc}(cK,2) = z$) // cK equals K;
}
```
2. [10 points]

Can the attacker obtain data by dictionary attack, assuming \( K \) is a weak key?

Solution

Yes.

Here is an evolution ending with the attacker obtaining data.

1. Initial step
   After: let \( x_A \) be \( A.nL \)'s value; let \( t_A = A.tL = g^{x_A} \mod p \). \([A,B,1,\text{enc}(K,t_A)]\) in \( \alpha \);

2. Attacker:
   - remove msg 1; generate random \( x_B \);
   - let \( t_B = g^{x_B} \mod p \); send msg \( [B,A,1,t_B] \).

3. \( A.1 \) receive msg 2
   After: \([A,B,2,\text{enc}(L,\text{'HELLO',data})]\) in \( \alpha \), where \( L = t_B \cdot x_A \mod p = g^{x_A \cdot x_B} \mod p \).

Attacker now has
   - \( \text{enc}(K,t_A) \); let this be \( z_1 \). // step 2
   - \( x_B \); \( t_B = g^{x_B} \mod p \) // step 2
   - \( \text{enc}(L,\text{'HELLO',data}) \); let this be \( z_2 \). // step 3

Attacker can then do the following off-line dictionary attack:

```plaintext
for (cPw in Dictionary) { // cPw: candidate password
cK ← pwToKeyFunction(cPw); // cK: candidate password key
cTA ← dec(cK,z1); // cTA: candidate tA
cKeyDH ← cTA \cdot x_A \mod p;
cMsg ← dec(cKeyDH,z2);
if (cMsg.size = 2 and cMsg[0] = 'HELLO')
data ← cMsg[1]; // and K = cK and pw = cPw
}
```
3. [20 points]

Part a. [10 points]

Inv $A_1$ holds, where

$A_1 : \psi(A,\text{Key})$  // attacker does not learn $A$’s current master key

Solution

True.

At the start, only $A$ and $Z$ have $A$.key; attacker does not. $\text{getPwdA}$ does not give the attacker $A$.key, because it sets $A$.key to a new value (unknown to attacker) after reading it. Thus the only expressions involving $A$.key seen by the attacker are sent by $Z$; they have the form $\text{enc}(A$.key, $x$), where $x$ involves a random component ($k_{AB}$) and a value received from the channel (msg[3]). The attacker can set msg[3], but it cannot set $x$ to a simple function of $K$ or to $\text{dec}(A$.key, $A$.key). So the attacker cannot compute $A$.key.

Part b. [10 points]

Inv $A_2$ holds, where

$A_2 : ((j \in \text{hst.keys}) \land j > 0 \land \text{hst}[j] = [B,p] \Rightarrow \text{hst}[j-1] = [A,p]$  

Solution

False.

Counter-example evolution.

- Attacker eavesdrops an $A$-$B$ connection, say with session key $j_{AB}$ and ticket $j_{kt}$.
  
  After this: $\text{hst} = [[A,j_{AB}], [B,j_{AB}]]$.

- Attacker calls $\text{getPwdA}$ to get $A$.key’s value, i.e., $k_{AZ}$, and set $A$.key to new value.
  
  Attacker uses $k_{AZ}$ to get key $j_{AB}$ from messages in previous step.

- Attacker connects to $B$ using ticket $j_{kt}$ and session key $j_{AB}$.
  
  After this: $\text{hst} = [[A,j_{AB}], [B,j_{AB}], [B,j_{AB}]]$. $A_2$ does not hold.

In more detail:

1. Initial step
   $[A,Z,B,\_] \text{ in channel}.$
2. $Z.1$: receive msg 1
   $[Z,A,\text{enc}(k_{AZ},[\_B,j_{AB},j_{kt}])] \text{ in channel and }\alpha; \ j_{kt} = \text{enc}(k_{BZ},[j_{AB},A]).$
3. $A.1$: receive msg 2; send msg $[A,B,1,j_{kt},\text{enc}(j_{AB},\_)].$
4. $B.1$: receive msg 3; send msg $[B,A,\text{enc}(j_{AB},\_)].$
5. $A.2$: receive msg 4; send msg $[A,B,\text{enc}(j_{AB},\_)]$; update hst to $[[A,j_{AB}]]$; send msg $[A,Z,B,\_].$
6. $B.2$: receive msg 5; update hst to $[[A,j_{AB}], [B,j_{AB}]].$
7. Attacker: call $\text{getPwdA}$ to get $Tk_{AZ}$; decrypt field 2 of msg 2 using $Tk_{AZ}$ to get $j_{AB}$; send msg 3.
8. $B.1$: receive msg 7; send msg $[B,A,\text{enc}(k_{AB},[\_,y_B])]$, where $y_B = B.nL$.
9. Attacker: receive msg 8; send msg $[A,B,\text{enc}(j_{AB},y_B-1)].$
10. $B.2$: receive msg 9; update hst to $[[A,j_{AB}], [B,j_{AB}], [B,j_{AB}]]$.

At this point, $A_2$ does not hold.