Solution to problem 1a [10 points]

We prove *Inv* (mKey *ncf* α).

Informal argument [4 points]

It suffices to show that any mKey-term in α is a secure encryption using mKey (because of axiom 2). The messages sent by A or B contain three kinds of fields: (1) ids (A, B); (2) challenges (nL values); and (3) responses (enc(mKey,nR) values). Only the response field involve mKey, and these are secure encryptions using mKey because the attacker cannot write to chan.

Proof [6 points]

The conjunction of C_1–C_4 is invariantly complete, and C_1 implies (mKey *ncf* α).

C_1 : (y in α.*inpts*(mKey)) $\Rightarrow$ (y *seu* mKey) [2 points]

C_2 : (y in chan.*inpts*(mKey)) $\Rightarrow$ (y *seu* mKey) [2 points]

C_3 : A.nL.*inpts*(mKey) = [] [1 point]

C_4 : B.nL.*inpts*(mKey) = [] [1 point]

Details:

	C_1	C_2	C_3	C_4
initial step	true	true	true	true
A.1	C_2, C_3, C_1	C_2, C_3	true	C_4
B.1	C_2, C_4, C_1	C_2, C_4	C_3	$true$
B.2	C_1	C_2	C_3	C_4

Solution to problem 1b [10 points]

We prove *Inv* `forall(i in hst.keys: hst[i] = [B,nB,nA]` $\Rightarrow$ `[A,nA,nB] in hst[0..i-1])`

Informal argument [4 points]

Because the attacker cannot write `chan`, the protocol's behavior is simple to describe:

0. `A` sends `[A,B,1]`
1. `B` receives `[A,B,1]` and sends `[B,A,1,enc(mKey,1)]`.
2. `A` receives above msg, adds `[A,1,1]` to `hst`, and sends `[A,B,enc(mKey,1)]` and `[A,B,2]`.
3. `B` receives first msg above, adds `[B,1,1]` to `hst`.
4. Repeat steps 1, 2, 3 with the `nR` and `nL` values in the messages increased by 1.

So just before `[B,nB,nA]` is added to `hst`, the last entry in `hst` is `[A,nA,nB]`.

Proof [6 points]

The conjunction of D_0 and D_1 is invariantly complete. Hence *Inv* D_0 holds, and this is what we want to establish.

D_0 : `forall(i in hst.keys: hst[i] = [B,nB,nA]` $\Rightarrow$ `[A,nA,nB] in hst[0..i-1])`

D_1 : (E_1 `or` E_2 `or` E_3), where

E_1 : `(B at 1) and chan = [[A,B,A.nL]]` [2 points]

E_2 : `(B at 2) and B.nR = A.nL and chan = [[B,A,B.nL,enc(mKey,B.nR)]]` [2 points]

E_3 : `(B at 2) and B.nR < A.nL and ([A,B.nR,B.nL] in hst)`
`and chan = [[A,B,B.nR,enc(mKey,B.nL)], [A,B,A.nL]]` [2 points]

Details:

	D_0	D_1
initial step	true	true
`A.1`	D_0	D_1 (E_2 before ensures E_3 after)
`B.1`	D_0	D_1 (E_1 before ensures E_2 after)
`B.2`	D_0, D_1 (E_3 before)	D_1 (E_3 before ensures E_1 after)

Solution to problem 2a [10 points]

We prove *Inv* (mKey *ncf* α).

Informal argument [4 points]

(Almost the same as for problem 1a.)

It suffices to show that any mKey-term in α is a secure encryption using mKey (because of axiom 2). The messages sent by A or B contain three kinds of fields: (1) ids (A, B); (2) challenges (nL values); and (3) responses (enc(mKey,nR) values). Only the response field involve mKey, and these are secure encryptions using mKey because even though the attacker can write to chan, the mKey values it can write are themselves secure encryptions using mKey.

Proof [6 points]

(Predicates are the same as for problem 1a.)

The conjunction of C_1–C_4 is invariantly complete, and C_1 implies (mKey *ncf* α).

C_1 : (y in α.*inpts*(mKey)) $\Rightarrow$ (y *seu* mKey) [2 points]

C_2 : (y in chan.*inpts*(mKey)) $\Rightarrow$ (y *seu* mKey) [2 points]

C_3 : A.nL.*inpts*(mKey) = [] [1 point]

C_4 : B.nL.*inpts*(mKey) = [] [1 point]

Details: (The only difference from the table in 1a is the "attacker write" row.)

	C_1	C_2	C_3	C_4
initial step	true	true	true	true
A.1	C_2, C_3, C_1	C_2, C_3	true	C_4
B.1	C_2, C_4, C_1	C_2, C_4	C_3	$true$
B.2	C_1	C_2	C_3	C_4
attacker write	C_1	C_1, C_2	C_3	C_4

Solution to problem 2b [10 points]

We disprove *Inv* forall(i in hst.keys: hst[i] = [B,nB,nA] $\Rightarrow$ [A,nA,nB] in hst[0..i-1])

Informal argument [4 points]

In B's message, [B,A,nB,enc(mKey,nA)], nA equals nB (assuming the attacker does nothing). So the attacker can make up the response to B without A receiving the message.

Proof [6 points]

Counter-example evolution:

- Initial step
  After: [A,B,1] in chan; hst = [].
- B.1
  After: [B,A,1,enc(mKey,1)] in chan; B.nL = B.nR = 1; hst = [].
- Attacker, using enc(mKey,1) field in above message, sets chan to [[B,A,enc(mKey,1)]].
- B.2
  After: [B,A,1,enc(mKey,1)] in chan; hst = [[B,1,1]]. Assertion's predicate not satisfied.

Solution to problem 3 [15 points]

(Taken from my 414 spring 2010 exam 1.)

Part a. [5 points]

- A interacts with CA offline. [2 points]
- A generates its public-key pair [`pubA`, `priA`] and gives CA its `pubA`. [2 points]
  A gets CA's public key `pubCA` and (optionally) certificate for A issued by CA, `certA`. [2 points]

Part b. [3 points]

- C can impersonate A to B until A's certificate expires (1 year at worst) [3 points]

Part c. [6 points]

- A interacts offline with CA
- A generates a new public key pair (as in part a) [2 points]
- CA adds A's old certificate's serial number in the next CRL it issues [2 points]
- Assume B uses latest CRL. Then C can impersonate A to B until A's old certificate's expiry time or until next CRL is issued, which is within 1 hour of contacting CA, whichever is earlier. [2 points]

Part a
–2 point for missing CA's public key.
–1 point for missing certificate.

Part b
–2 point for not referring explicitly to expiry time

Part c
–3 point for not using CRL

Solution to problem 4a [5 points]

No, an attacker who can only read cannot obtain data because A and B establish a Diffie-Hellman key.

Solution to problem 4b [10 points]

Yes, an attacker who can read and write chan can obtain data, by doing the classic man-in-middle attack.

- Initial step: [A,B,A.tL] in chan.
- Attacker removes message. Generates random ZnL, sets ZtL to g^{ZnL} mod p, sends [A,B,ZtL].
- B.1: sets B.keyDH to $g^{ZnL \cdot B.nL}$ mod p, sends [B,A,B.tL].
- Attacker removes message. Sends [B,A,ZtL].
  Sets ZBkeyDH to $g^{ZnL \cdot B.nL}$ mod p (DH key shared with B).
  Sets ZAkeyDH to $g^{ZnL \cdot A.nL}$ mod p (DH key shared with A).
- A.1 sends [A,B,'DATA',enc(A.keyDH,data)].
- Attacker reads message. Uses ZAkeyDH to decrypt field 4.