4 problems over 5 pages. 60 points. Closed book. Closed notes. No calculator or electronic device.

1. [20 points]

The following program of the Needham-Schroeder protocol has a KDC Z, client A, server B, and attacker.

```
ProtocolNS(Z, A, B) { // kdc, client, server
   chan ← [];
   hst ← [];   // connection history
   kAZ ← random();
   kBZ ← random();
   startSystem(Z, KdcNS(Z,A,B,kAZ,kBZ));
   startSystem(A, ClientNS(A,Z,B,kAZ));
   startSystem(B, ServerNS(B,Z,A,kBZ));
   startSystem(AttackerNS1());
}
```

```
KdcNS(Z, A, B, kAZ, kBZ) {
   // atomicity points: 1
   mKeyA ← kAZ;
   mKeyB ← kBZ;
   while (true) {
 1:  msg ← rx([A,Z,B,.]);
      n ← msg[3];
      k ← random();
      tkt ← enc(mKeyB,[k,A]);
      tx([Z,A,enc(mKeyA,[n,B,k,tkt])]);
   }
}
```

```
ClientNS(A, Z, B, kAZ) {
   // atomicity points: 1, 2
   mKey ← kAZ;
   while (true) {
      n1 ← random();
      tx([A,Z,B,n1]);
 1:   msg ← rx([Z,A,.]);
      z ← dec(mKey,msg[2]);
      if (z.size = 4 and z[0,1] = [n1,B]) {
         kAB ← z[2];
         tkt ← z[3];
         n2 ← random();
         tx([A,B,tkt,enc(kAB,n2)]);
 2:      msg ← rx([B,A,.]);
         z ← dec(kAB,msg[2]);
         if (z.size = 2 and z[0] = n2-1) {
            n3 ← z[1];
            hst.append([A,kAB]);
            tx([A,B,enc(kAB,n3-1)]);
         }
      }
   }
}
```

```
ServerNS(B, Z, A, kBZ) {
   // atomicity points: 1,2
   mKey ← kBZ;
   while (true) {
 1:  msg ← rx([A,B,.,.]);
      tkt ← msg[2];
      z ← dec(mKey,tkt);
      if (z.size = 2 and z[1] = A) {
         kAB ← z[0];
         n2 ← dec(kAB,msg[3]);
         n3 ← random();
         tx([B,A,enc(kAB,[n2-1,n3])]);
 2:      msg ← rx([A,B,.]);
         if (msg[2] = enc(kAB,n3-1))
            hst.append([B,kAB]);
       }
      }
   }
}
```

```
AttackerNS1() {
   read and write chan
}
```

1 (cont). [20 points]

For each assertion below, prove or disprove whether the assertion holds for `ProtocolNS`.

a. *Inv* `(A.kAB` *ncf* α`)`

b. `((i in hst.keys) and i` $\neq$ `0 and hst[i] = [B,k])` $\Rightarrow$ `hst[i-1] = [A,k]`

If you prove, present a list of predicates, say A_0, A_1, $\cdots$, whose conjunction should be invariantly complete and implies the assertion's predicate. (A_0 could be the assertion's predicate.) No need for informal argument or justification table.

If you disprove, present a counter-example evolution. Start each step execution on a new line. No need for informal argument.

2. [20 points]

Repeat problem 1 but with program AttackerNS1 replaced by program AttackerNS2 given below:

```
AttackerNS2() {
   read and write chan

   // get A.mKey and set A.mKey/Z.mKeyA to new random value
   getPwdA() {
      α.append(A.mKey);
      A.mKey ← Z.mKeyA ← random();
   }
}
```

3. [10 points]

In the program below: `g` and `p` are public Diffie-Hellman parameters; and `W` is a secret password-derived key initially unknown to the attacker.

```
Protocol(A, B, W, g, p) {
   chan ← [];
   startSystem(A, Client(A,B,W,g,p));
   startSystem(B, Server(B,A,W,g,p));
   startSystem(Attacker());
}
```

```
Attacker() {
    <read and write chan>
}
```

```
Client(A, B, W, g , p) {
   // atomicity points: 1
      nL ← random();
      tL ← g^nL mod p;
      tx([A,B,1,tL]);
  1:  msg ← rx([B,A,.,.]);
      tR ← msg[2];
      keyDH ← tR^nL mod p;
      cB ← dec(W, dec(keyDH,msg[3]));
      tx([A,B,2,enc(keyDH, enc(W,cB+1))]);
}
```

```
Server(B, A, W, g, p) {
   // atomicity points: 1,2
  1:  msg ← rx([A,B,1,.]);
      tR ← msg[3];
      nL ← random();
      tL ← g^nL mod p;
      keyDH ← tR^nL mod p;
      cB ← random();
      tx([B,A,tL,enc(keyDH, enc(W,cB))]);
  2:  msg ← rx([A,B,2,.]);
}
```

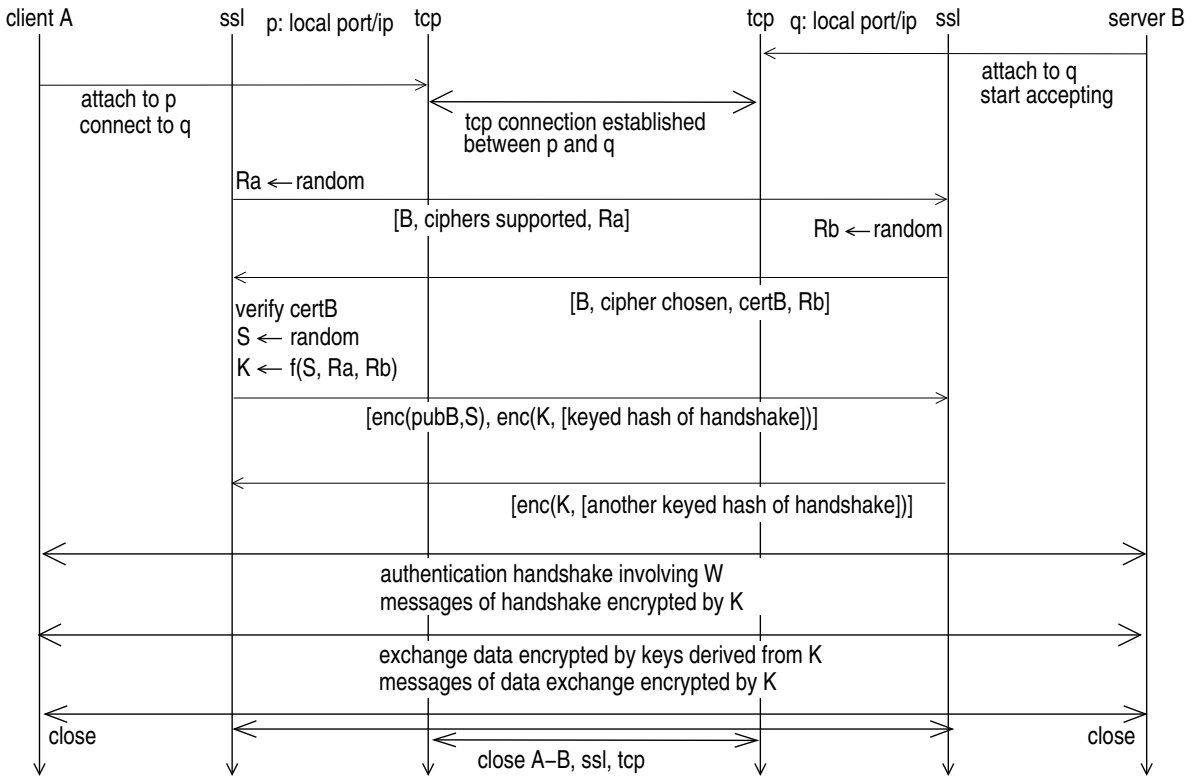
a. Can an attacker who can only read `chan` obtain `W` by offline-dictionary attack?

b. Can an attacker who can read and write `chan` obtain `W` by offline-dictionary attack?

In each part above, answer yes or no. If you answer yes, give the dictionary attack and say how the attacker obtains the inputs for the dictionary attack. If you answer no, explain briefly (no need for predicates).

4. [10 points]

Outlined below are the handshakes invovled in an SSL session between client `A` and server `B`. The client attaches at tcp-port/ip-address `p`. The server attaches at tcp-port/ip-address `q`. The server has a certificate, `certB`. The client has the public key of the CA that issued `certB`. The client and server also share a key, `W`, derived from a password.



Answer the following questions.

a. Indicate on the figure the point at which `A` authenticates `B`.
Indicate on the figure the point at which `B` authenticates `A`.

b. At the client node, are the messages sent between client `A` and its ssl encrypted.

c. Outline the structure of a tcp message (between tcp `p` and tcp `q`) in the data exchange phase. Indicate which parts are encrypted and which are not.

d. Can an attacker that can read/write messages between ssl and tcp in the client node obtain the data exchanged between client `A` and server `B`. Explain briefly.