1. [20 points]
An organization has four departments. Each department has a CA (certification authority) that issues certificates for employees in its department. Let P, Q, R, S be these CAs. There is also a “root” CA, named X, that issues certificates for P, Q, R and S. X does not issue certificates for employees.

a. Give the steps taken when a new employee joins P’s department.

Solution [8 pts]
- New employee, say A, generates public-key pair, say [priA, pubA]  [3 points]
- Gives pubA to P. [Optional] gets back a certificate signed by P, say certA.  [2 points]
  Note: certA = [A id, pubA, serial #, expiry date, P’s signature on certA]
- Gets X’s public key, say pubX  [3 points]

End of solution

b. A and B are two employees of P’s department. Supply an authentication handshake by which A connects to B and establishes a session key $n_A \oplus n_B$, where $n_A$ and $n_B$ are random numbers generated by A and B, respectively, during the authentication handshake. Your protocol must be secure against an attacker that can eavesdrop, intercept and send messages. Give only the messages exchanged and the actions taken at A and B; do not give explanations or motivations.

Solution [12 pts]
Below: certP is P’s certificate (signed by X); crlX is a recent CRL of X; crlP is a recent CRL of P.

<table>
<thead>
<tr>
<th>client A</th>
<th>server B</th>
</tr>
</thead>
<tbody>
<tr>
<td>has pubX; gets certA, certP, crlP, crlX (from DS)</td>
<td>has pubX; gets certP, crlX, crlP (from DS)</td>
</tr>
</tbody>
</table>
| send msg1: [A, B, certA] | receive msg1
verify certA (using pubX, certP, crlX, crlP), get pubA
generate nB
send msg2: [B, A, enc(nB, pubA)] | [2 points] |
| receive msg2
verify certB (using pubX, certP, crlX, crlP), get pubB
generate nA
send msg3: [A, B, enc(nA, pubB)] | [2 points] |
| extract nB from xB (using priA); session key ← nA\oplus nB | extract nA from xA (using priB); session key ← nA\oplus nB |

Grading
1 point: Showing how A gets certB (from DS)
2 points: B gets certA from A (in msg1). Alternative: msg1 does not have certA; B gets it from DS after receiving msg1
2 points: A sends enc(nA, pubB) to B.
2 points: B sends enc(nB, pubA) to A.
3 points: Using certP, crlX, crlP (at A and B). Note: Not ok for A (or B) to get pubB when they join and use it always.
2 points: Using pubX (at A and B)

End of solution
2. [20 points]

Client A and server B share a weak secret key J (e.g., obtained from a password dictionary). They also share Diffie-Hellman parameters p and g. Supply an authentication handshake by which A connects to B and establishes a session key. Your protocol must be secure against an attacker that can eavesdrop, intercept and send messages, and do dictionary attacks. 

*Give only the messages exchanged and the actions taken at A and B; do not give explanations or motivations.*

**Solution [20 pts]**

<table>
<thead>
<tr>
<th>Client A (has J) g, p</th>
<th>Server B (has file with entry [A:J]) g, p</th>
</tr>
</thead>
<tbody>
<tr>
<td>generate random sA</td>
<td>receive msg1</td>
</tr>
<tr>
<td>tA ← g^{sA} mod p</td>
<td>extract tA // using J</td>
</tr>
<tr>
<td>send msg1: [A, B, enc(tA, J)]</td>
<td>generate random sB</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>receive msg2</td>
<td></td>
</tr>
<tr>
<td>extract tB // using J</td>
<td></td>
</tr>
<tr>
<td>session key ← tB^{sA} mod p</td>
<td></td>
</tr>
</tbody>
</table>

**Grading**

5 pts: not using Diffie-Hellman (DH). Don’t see how to solve it without DH.
15 pts: for regular (unauthenticated) DH.
15-17 pts: for an (incorrect) authenticated DH that exposes J to dictionary attack.

**Fyi: Examples** of incorrect “authenticated” DH that exposes J to dictionary attack:

- A sends \(\text{enc}([nA, tA], J)\) B responds with \(\text{enc}([nA+1, tB], J)\).
  
  **Attack:** Eavesdropper has nA and nA+1 encrypted by J. So can do dictionary attack
  
  (Note: \(\text{enc}([tA, nA], J)\) and \(\text{enc}([nB, tB], J)\) may be ok)

- A and B do regular DH. Establish session key \(K = g^{sAsB} \pmod p\).
  
  Then A sends msg1 containing \(\text{enc}(nA, J)\) and \(\text{enc}(nB, J)\). B responds with msg2 containing \(\text{enc}(nA+1, J)\).

  **Attack:** Do man-in-middle attack during regular DH, establishing DH keys, say K1 with A and K2 with B.
  
  When A sends msg1, attacker relays it (via K1, K2) to B, and obtains \(\text{enc}(nA, J)\).
  
  When B sends msg2, attacker relays it to A (via K2, K1) and obtains \(\text{enc}(nA+1, J)\).
  
  Attacker can now do dictionary attack on J.

- A and B do regular DH. Choose session key as \(L = \text{enc}(K, J)\) (i.e., \(L = \text{enc}(g^{sAsB} \pmod p, J)\)).

  **Attack:** Do man-in-middle attack during regular DH, establishing DH keys, say K1 with A and K2 with B.
  
  So A’s session key is, say \(L1 = \text{enc}(K1, J)\). And B’s session key is, say \(L2 = \text{enc}(K2, J)\)
  
  Suppose A sends recognizable plaintext encrypted by L1, say msg3 = \(\text{enc}("Hello", L1)\).
  
  Do dictionary attack: \(cL1 \leftarrow \text{enc}(K1, J)\); check for decrypt(msg3, cL1) = “Hello”.

**End of solution**