Mid Semester - Fall, 2025
M. Tech. Cryptology & Security - 2^{nd} Year
Advanced Cryptology
(with Answers)

September 12, 2024

Maximum Marks: 25
Maximum Time: 2 hr
Open note/book exam

Instructions

- Maximum marks is 25. Total marks provided in the paper: 30.
- Be short and precise. For example, a 5 (respectively 3) mark question's answer should not be ideally more than a page (respectively half a page) with standard hand-writing. Partial marking will be provided for a partially correct answer/attempt.
- All parts of the same questions must be done at the same place, and in order. Marks will be deducted for violating this – this will be followed strictly.
- Verbosity (e.g. unnecessary/irrelevant sentences) is highly discouraged, and may lead to deduction of marks.
- You can use any book/notes during the exam, but *not internet*.
- The seating arrangements and other regulations circulated from Dean's office must be followed adequately.

Questions

1. Alice and Bob are going for a private auction, where they want to bid in any order, but their goal is to get the item at the minimum price. The auctioneer plans to use a CPA secure encryption to hide their bids. Auctioneer has the decryption key.

   (a) Then what would be the best strategy for Alice/Bob such that they can get the item at the minimum price?

   (b) How the auctioneer can change/strengthen the scheme so that this strategy does not work any more? Explain briefly.

   (3+3 = 6)

   Answer:

   (a) Since CPA secure encryptions are malleable, the best strategy which supports obtaining the item in minimal price is to wait for other's encrypted bid, say $\mathsf{Enc}(B)$, and then compute own's encrypted bid $\mathsf{Enc}(B+1)$ by just mauling blindly. As result, for the mauler always obtains the item at a price which is exactly above the bid set by the other party.

(b) The auctioneer can strengthen the scheme to make it CCA-secure. We know that CCA-secure encryptions are not susceptible to malleability attacks. This will ensure that the above strategy does not work anymore.

2. Consider a simple private voting scheme with the following steps

   - There are 3 options to vote. Each voter's vote is a binary vector of dimension 3, where 1 denotes a "yes" vote, and 0 means a "no" vote. For example, a vote for the first and third option shall be denoted by a vector $(1, 0, 1)$.
   - The voting authority releases a public key for Exp-El-Gamal Encryption, for which only the authority holds the secret decryption key.
   - Each party sends their votes (the vector) to the authority.

   Now answer the following questions:

   (a) Consider a setting, where all voters are honest, and the authority is semi-honest. So, the honest voters do not want their individual votes to get leaked to the authority. However, the authority will not diverge from the protocol steps it is supposed to execute (as it is semi-honest). Describe a protocol.

   (b) Now, in addition to semi-honest authority, consider malicious voters. Assume that a voter can vote "yes" to multiple options. In that case, how the authority can verify that each encrypted votes are legitimate? (e.g. $(1, 0, 1)$ is a legitimate voting vector, whereas $(2, 3, 8)$ is not.)

   (c) Finally consider that in addition to above, each voter can vote "yes" to only one option (e.g. $(0, 0, 1)$ is a legitimate voting vector, whereas $(1, 0, 1)$ is not.). Then how would the authority verify the legitimacy of each encrypted vote?

   (5+5+5 = 15)

   Answer:

   (a) Each player's vote is a binary vector. Let us assume that player-i's vote is a binary vector $\vec{v}_i$. Then player-i encrypts $\vec{v}_i$ component-wise using the given Exp-El-Gamal public key to get a ciphertext vector $\vec{c}_i = (c_{i1}, c_{i2}, c_{i3})$ where each c_{ij} is an Exp-El-Gamal ciphertext $(g^{r_{ij}}, pk^{r_{ij}} g^{v_{ij}})$ for player-i's vote v_{ij} for candidate-j. The authority gathers the ciphertexts and computes homomorphic component-wise additions $a_j = ADD(c_{1j}, c_{2j}, \ldots)$. Then it decrypts each a_j to get the total votes for each candidate-j.

   (b) Each voter-i is asked to provide a non-interactive zero-knowledge proof for the following statement:

   – Each ciphertext c_{ij} contains an encryption of either 0 or 1.

   This can be easily achieved by using a OR proof for two Chaudm-Pedersen statements for Exp-El-Gamal:

   – If c_{ij} is parsed as (R_{ij}, E_{ij})), then either R_{ij} and E_{ij} has equal exponents with respect to bases g and pk; or R_{ij} and E_{ij}/g has equal exponents with respect to bases g and pk.

   This will convey that the encrypted value is a bit, without revealing the exact value.

   (c) To ensure that each voter can only vote one candidate, the authority further asks for another NIZK proof:

   – If $\vec{c}_i = (c_{i1}, c_{i2}, c_{i3})$ is player-i's ciiphertext vector, and let $s_i = ADD(c_{i1}, c_{i2}, c_{i3})$, then the proof attests to the fact that s_i encrypts exactly 1 – this can be proven by applying another single Chaum-Pederson proof of the second type from above.

3. Let π be a generic two party computation (2PC) protocol which computes any boolean function $\{0,1\} \times \{0,1\} \to \{0,1\}$ between any two parties P_1 and P_2 when P_1 holds the first input x_1 and P_2 holds the second input x_2 and at the end of the protocol both of them get the output y. Now

consider a *specific* function $y = f(x_1, x_2)$ which is computed by P_1 and P_2 using π. At the end of the protocol party P_1 is able to compute x_2 and party P_2 is able to compute x_1. Then answer each of the following questions with arguments.

(a) Is it possible to conclude whether π is secure or insecure? Argue with a concrete example.

(b) Now consider that the same protocol is used to compute the AND function $g(x_1, x_2) = x_1 \wedge x_2$. If P_1's input is 1 then it is able to recover P_2's input x_2. Does the conclusion about the security of π change or not?

(c) Now consider the same protocol is used to compute the OR function $h(x_1, x_2) = x_1 \vee x_2$. If P_1's input is 1 then in the end it recovers P_2's input x_2. How does this fact change your conclusion about π, if at all?

(3+3+3 = 9)

Answer:

(a) No, because the functionality may leaks the inputs. So, even if a secure MPC protocol is executed, the same can happen. Formally, this is easy to see via a simulation argument for XOR functionality: if one party is corrupt, then simulator can just use that party's input x_1, and output y to compute the exact input $x_2 := y \oplus x_1$ of the other party.

(b) No. Again, if P_1's input is 1, then the output becomes $y = 1 \wedge x_2 = x_2$. Hence, again, given P_1's input $x_1 = 1$, the simulator can derive $x_2 := y$. In oyther words, if a party's input is 1, then it bis supposed to learn other party's input anyway. That continues to be hold even if the protocol is secure

(c) This time we can conclude that the protocol is insecure. This is because, if $x_1 = 1$, the $y = 1$ regardless of x_2. However, if the protocol is leaking x_2, then that will not be computed / simulated from x_1 and y only.