Indian Institute of Technology, Kanpur
Computer Science and Engineering
Quiz 1
CS670: Cryptographic Techniques for Privacy Preservation

Instructor: Adithya Vadapalli

28/08/2025

Name: ____

Roll Number: ____

This quiz contains 9 pages (including this cover page) and 9 questions. Total of marks is 25.

Distribution of Marks

Question	Points	Score
1	2	
2	1	
3	1	
4	2	
5	2	
6	2	
7	2	
8	5	
9	8	
Total:	25	

1. Consider two parties, Alice and Bob, who would like to compute the function

$$f(x_1, x_2, y_1, y_2) = (x_1 \oplus y_1) \wedge (x_2 \oplus y_2),$$

where x_1, x_2 are Alice's private inputs and y_1, y_2 are Bob's private inputs. Assume they use a naive Garbled Circuits protocol (without any optimizations).

(a) (1 point) How many Oblivious Transfer (OT) instances are required to compute this function securely?

Solution: 2; because the number inputs of the evaluator is two.

(b) (1 point) How many ciphertexts in total must the Garbler send to the Evaluator?

Solution: 12; because there are three gates and we need four per gate.

2. (1 point) In the GRR3 optimization (Garbled Row Reduction with 3 ciphertexts), the Garbler sends only three ciphertexts per gate. The fourth ciphertext is not transmitted explicitly. What value does the Evaluator compute as the fourth ciphertext?

Solution: 0^n

3. (1 point) After applying the Free-XOR optimization in garbled circuits, how many ciphertexts need to be sent to evaluate an XOR gate?

Solution: 0

4. (2 points) In Yao's Garbled Circuits protocol, the *point-and-permute* technique allows the Evaluator to efficiently identify the correct row of the garbled table. Suppose Alice garbles a single AND gate (i.e., the circuit consists of only this one gate). How many ciphertexts must she send to Bob in order for him to evaluate the gate? Briefly explain in a single sentence.

Solution: 2; Because the garbler knows his input. Therefore, he knows the ciphertexts that are not relevant.

5. (a) (1 point) What is Random OT (ROT)? Clearly state the inputs and outputs.

Solution: The receiver gets a random bit b and a random message m_b. The sender gets two random messages m_0 and m_1.

(b) (1 point) What is the role of ROT in secure computation protocols?

Solution: Serves as a preprocessing step before the real OT

6. (a) (1 point) What is OT Extension?

Solution: Method to do several OTs at the cost of a few

(b) (1 point) What purpose does OT Extension serve in secure computation?

Solution: OTs are expensive – so, they allow us to reduce the cost in scenariou where we needy many OTs

7. (2 points) Suppose two values x and y are secret-shared among $n = 3$ parties using *Replicated Secret Sharing* (RSS) with threshold $t = 2$ (i.e., $(3, 2)$-RSS). We would like the parties to obtain additive shares **(not RSS!!)** of the product $z = x \cdot y$. How much communication (in terms of the number of field elements sent per party) is required to perform this multiplication? Briefly justify your answer.

Solution: 0 The three parties can locally compute the additive shares.

8. Recall the protocol for secure multiplication of secret-shared values in Multiparty Computation (MPC). Parties $\mathcal{S}_0$ and $\mathcal{S}_1$ hold (x_0, y_0) and (x_1, y_1) respectively.

   The Du–Atallah protocol proceeds as follows: A trusted dealer samples random values $(X_0, Y_0, X_1, Y_1, \alpha)$. The dealer sends $(X_0, Y_0, X_0 \cdot Y_1 + \alpha)$ to $\mathcal{S}_0$ and $(X_1, Y_1, X_1 \cdot Y_0 - \alpha)$ to $\mathcal{S}_1$. Next, $\mathcal{S}_0$ and $\mathcal{S}_1$ exchange $x_b + X_b$ and $y_b + Y_b$ for $b \in \{0, 1\}$.

Then the servers compute:

$$z_0 \leftarrow x_0 \cdot (y_0 + (y_1 + Y_1)) - Y_0 \cdot (x_1 + X_1) + (X_0 \cdot Y_1 + \alpha),$$

$$z_1 \leftarrow x_1 \cdot (y_1 + (y_0 + Y_0)) - Y_1 \cdot (x_0 + X_0) + (X_1 \cdot Y_0 - \alpha).$$

(a) (2 points) Explain the purpose of the random term α introduced in the multiplication protocol. What will happen to the correctness if the dealer does not use the term α? What will happen to the privacy if the dealer does not use the term α?

Solution: The correctness is not affected. The security is. Without α, P_0 learns, X_0 and $X_0 \cdot Y_1$, thus learning Y_1. But, the party also learns $y_1 + Y_1$. Therefore, it can learn y_1 the secret input of P_1.

(b) (3 points) Describe the most efficient way to evaluate the dot product of two secret-shared vectors using MPC. Assume that $(\vec{x}_0, \vec{y}_0)$ is held by $\mathcal{S}_0$ and $(\vec{x}_1, \vec{y}_1)$ is held by $\mathcal{S}_1$. They should obtain z_0 and z_1 respectively, such that: $z_0 + z_1 = \langle \vec{x}, \vec{y} \rangle$.

Solution: A trusted dealer samples random values $(\vec{X}_0, \vec{Y}_0, \vec{X}_1, \vec{Y}_1, \alpha)$. The dealer sends $(\vec{X}_0, \vec{Y}_0, \vec{X}_0 \cdot \vec{Y}_1 + \alpha)$ to $\mathcal{S}_0$ and $(\vec{X}_1, \vec{Y}_1, \vec{X}_1 \cdot \vec{Y}_0 - \alpha)$ to $\mathcal{S}_1$. Next, $\mathcal{S}_0$ and $\mathcal{S}_1$ exchange $\vec{x}_b + \vec{X}_b$ and $\vec{y}_b + \vec{Y}_b$ for $b \in \{0, 1\}$.

Then the servers compute:

$$z_0 \leftarrow \vec{x}_0 \cdot (\vec{y}_0 + (\vec{y}_1 + \vec{Y}_1)) - \vec{Y}_0 \cdot (\vec{x}_1 + \vec{X}_1) + (\langle \vec{X}_0, \vec{Y}_1 \rangle + \alpha),$$

$$z_1 \leftarrow \vec{x}_1 \cdot (\vec{y}_1 + (\vec{y}_0 + \vec{Y}_0)) - \vec{Y}_1 \cdot (\vec{x}_0 + \vec{X}_0) + (\langle \vec{X}_1, \vec{Y}_0 \rangle - \alpha).$$

.

9. Design $(2+1)$-party MPC protocols for the following operations on secret-shared values:

   (Hint: Write these functions as expressions that contain only those operations that you know how to do in MPC!)

   (a) (4 points) Bitwise OR $(a \vee b)$. P_0 and P_1 should obtain shares of $a \vee b$.

Solution: We can express bitwise OR as

$$a \vee b = (a \oplus b) \oplus (a \wedge b).$$

This requires one AND operation and two XOR operations.

1. XOR operations: These involve no communication and are executed locally.
2. AND operation: This is performed using a Du-Atallah-style multiplication, where multiplication is replaced by bitwise AND and addition by bitwise XOR.

The AND requires one round in the preprocessing phase (40 ms) and one round in the online phase (40 ms). In the preprocessing phase, $\mathcal{S}_2$ provides correlated randomness for the AND. In the online phase, the parties exchange blinded shares and complete the multiplication. The XORs are executed locally during the online phase.

   (b) (4 points) Conditional selection (IF statement: $(c?a : b)$). In other words, if $c = 1$, output a. And if $c = 0$: output b. P_0 and P_1 should get shares of either a or b (depending on the value of c).

Here P_0 holds (a_0, b_0, c_0) and P_1 holds (a_1, b_1, c_1) such that: $a_0 \oplus a_1 = a$, $b_0 \oplus b_1 = b$, and $c_0 \oplus c_1 = c$. We have $a, b \in \{0,1\}^\lambda$ and $c \in \{0,1\}$.

Solution: We can express conditional if:

$$c \wedge a \oplus ((1-c) \wedge b) = b \oplus (c \wedge (a \oplus b))$$

This requires one AND operation and two XOR operations.

1. XOR operations: These involve no communication and are executed locally.
2. AND operation: This is performed using a Du-Atallah-style multiplication; however, there is a subtle difference between this and the previous part. Previously, it was a bitwise AND operation between two *lambda*-bit strings. We now have c, which is a bit. There are several ways to get around it. The simplest is to extend it to *lambda* bits.

The AND requires one round in the preprocessing phase (40 ms) and one round in the online phase (40 ms). In the preprocessing phase, $\mathcal{S}_2$ provides correlated randomness for the AND. In the online phase, the parties exchange blinded shares and complete the multiplication. The XORs are executed locally during the online phase.

Clearly indicate which steps can be moved to the preprocessing phase.

Assume it takes 40 ms to send a message between any two parties. How much communication time does your protocol incur across the Internet (in the *preprocessing* and the *online* phase)?

This page is intentionally left blank to accommodate work that wouldn't fit elsewhere and/or scratch work.

This page is intentionally left blank to accommodate work that wouldn't fit elsewhere and/or scratch work.