CS 598DH Secure Computation – Homework 1

Professor David Heath

Due: February 15, 2024

Problem 1. To formalize the security of a protocol, it is essential to first specify the *functionality* of that protocol. The functionality specifies *ideal-world behavior*. Write out functionalities for each of the following tasks (an example functionality presented in Figure 1):

1. Compute set intersection, where each party's input is a set.
2. Voting. You choose parameters of the voting problem.

Hint: The tasks are not formally specified; choose your own formalization. The goal of this problem is to practice notation and to get used to formalizing intuitive notions.

Answer 1.

Problem 2. Sometimes it is not clear whether a certain behavior is an "attack" against a protocol. Our definition of security roughly states that a protocol is insecure if there is an adversarial behavior that is possible in the real world, but that same behavior is impossible in the ideal world.

For each of the following, consider a semi-honest adversary that successfully carries out the described behavior *as part of a real-world protocol*. Specify whether this behavior indicates that the real-world protocol is *insecure* (yes or no). If the procotol is insecure, give a brief explanation.

1. Alice holds x and Bob holds y, where $x, y \in \{0, ..., N-1\}$. They wish to compute:

$$(x + y) \bmod N$$

   A semi-honest adversary corrupts Alice and learns Bob's input in its entirety.

2. Alice holds x and Bob holds y, where $x, y \in \{0, ..., 7\}$. They wish to compute:

$$\begin{cases} 1 & \text{if } x < y \\ 0 & \text{otherwise} \end{cases}$$

   A semi-honest adversary corrupts Alice and learns the *most* significant bit of y (assume y is written as a three bit value, e.g. $4 = 100_2$).

PARAMETERS:

1. Let $P_0, ..., P_{n-1}$ be n parties. Suppose $n > 2$.
2. Each party P_i has input $x_i \in [0..k)$.

FUNCTIONALITY:

1. Compute the lowest and highest elements:

$$x_{\text{lo}} = \min_{i=0}^{n-1}(x_i) \qquad x_{\text{hi}} = \max_{i=0}^{n-1}(x_i)$$

2. Compute the set $X = \{x_i\} \setminus \{x_{\text{lo}}, x_{\text{hi}}\}$.
3. Each party P_i outputs the average of X:

$$\frac{\sum_i x_i}{n-2} \text{ where } x_i \in X$$

Figure 1: Example functionality that computes the average, omitting the single highest and the single lowest value. Each party provides an input between 0 and k

3. Alice holds x and Bob holds y, where $x, y \in \{0, ..., 7\}$. They wish to compute:

$$\begin{cases} 1 & \text{if } x < y \\ 0 & \text{otherwise} \end{cases}$$

A semi-honest adversary corrupts Alice and learns the *least* significant bit of y (assume y is written as a three bit value, e.g. $4 = 100_2$).

4. Alice holds x and Bob holds y, where $x, y \in \{0,1\}^n$. They wish to compute the inner product of those strings modulo 2:

$$\left(\sum_{i=0}^{n-1} x_i \cdot y_i\right) \bmod 2$$

A semi-honest adversary corrupts Alice and learns whether the y string has a majority of 0s or a majority of 1s. Assume that n is odd.

5. Alice and Bob each hold a string: Alice's string k is used as an encryption key, and Bob's string x is plaintext. Parties wish to compute $\text{Enc}(k, x)$ and deliver the output to both parties. Assume the encryption scheme is correct. I.e., the following holds:

$$\text{Dec}(k, \text{Enc}(k, x)) = x$$

A semi-honest adversary corrupts Alice and learns Bob's input in its entirety. *Hint: You might be concerned that I didn't define security of the encryption scheme. Does it matter?*

Answer 2.

Problem 3. Suppose Alice has an input $x \in \{0, 2, 4, ..., 8\}$ and Bob has an input $y \in \{1, 3, 5, ..., 9\}$. Here is a protocol that computes the function $\max(x, y)$:

- If Bob has input $y = 9$, he announces "yes" and both parties output 9 and halt. Otherwise he announces "no" and the protocol continues.
- If Alice has input $x = 8$, she announces "yes" and both parties output 8 and halt. Otherwise, she announces "no" and the protocol continues.
- If Bob has input $y = 7$, he announces "yes" and both parties output 7 and halt. Otherwise he announces "no" and the protocol continues.
- If Alice has input $x = 6$, she announces "yes" and both parties output 6 and halt. Otherwise, she announces "no" and the protocol continues.
- ...
- The protocol continues until some party says "yes", at which point the output is determined and the protocol is finished.

Construct simulators that demonstrate this protocol is secure in the presence of a semi-honest adversary.

Answer 3.

Problem 4. Consider Problem 3, modified as follows: Alice has an input $x \in \{0, 1, ..., 9\}$ and Bob has an input $y \in \{0, 1, ..., 9\}$. The protocol is amended as follows:

- If Bob has input $y = 9$, he announces "yes" and both parties output 9 and halt. Otherwise he announces "no" and the protocol continues.
- If Alice has input $x = 9$, she announces "yes" and both parties output 9 and halt. Otherwise, she announces "no" and the protocol continues.
- If Bob has input $y = 8$, he announces "yes" and both parties output 8 and halt. Otherwise he announces "no" and the protocol continues.
- ...
- The protocol continues until some party says "yes", at which point the output is determined and the protocol is finished.

Is the modified protocol still secure in the semi-honest model? If so, prove security; if not, explain the vulnerability.

Answer 4.

Problem 5. We often assume that in 2PC, each party outputs the same value. Let's instead consider the case where the ideal functionality delivers separate outputs to each party.

Suppose there are two functions f_A and f_B and that in the ideal world, the functionality receives x from Alice and y from Bob, then delivers only $f_A(x, y)$ to Alice and only $f_B(x, y)$ to Bob.

1. Give an example f_A and f_B where it is demonstrably insecure (i.e., less secure than the ideal world described above) if in the real world *both* parties learn $f_A(x, y)$ and $f_B(x, y)$. *Hint: there are extremely simple choices of f_A and f_B that meet the criteria.*

2. Suppose we have access to a semi-honest secure protocol that computes any function $f(x, y)$ and delivers this output to both Alice and Bob.

   (a) Formalize a new protocol that uses the above protocol as a black-box. This new protocol should deliver $f_A(x, y)$ to Alice and $f_B(x, y)$ to Bob.

   (b) Prove your new protocol secure in the semi-honest model by constructing simulators.

   You may assume that x, y, $f_A(x, y)$, and $f_B(x, y)$ are each n-bit strings.

Answer 5.

Problem 6. Some MPC techniques can be made *very* efficient when there are many parties and few corruptions. For instance, in class we discussed Oblivious Transfer (OT). With few corruptions, we can implement OT very efficiently. In this problem, consider the following setting:

- The adversary is *semi-honest*.
- There are *three parties*.
- There is an *honest majority*. I.e., the adversary *corrupts at most one party*.

Consider the following "assisted OT" functionality:

PARAMETERS:

1. Let P_0, P_1, P_2 be three parties.
2. The sender P_0 inputs two secrets $x_0, x_1 \in \{0,1\}^n$.
3. The receiver P_1 inputs a selection bit $s \in \{0,1\}$.
4. The helper P_2 inputs $\perp$.

FUNCTIONALITY:

1. P_0 outputs $\perp$.
2. P_1 outputs x_s.
3. P_2 outputs $\perp$.

1. Construct a protocol that securely achieves the above functionality in the considered setting. Your protocol should require *no cryptographic assumptions* (e.g., you do not need to assume something like DDH) and the total number of transmitted bits should be $O(n)$.

2. Construct simulators for each party that demonstrate your protocol is secure.

Answer 6.