# CS 65500
# Advanced Cryptography

## Lecture 14: Maliciously Secure MPC

Instructor: Aarushi Goel

Spring 2025

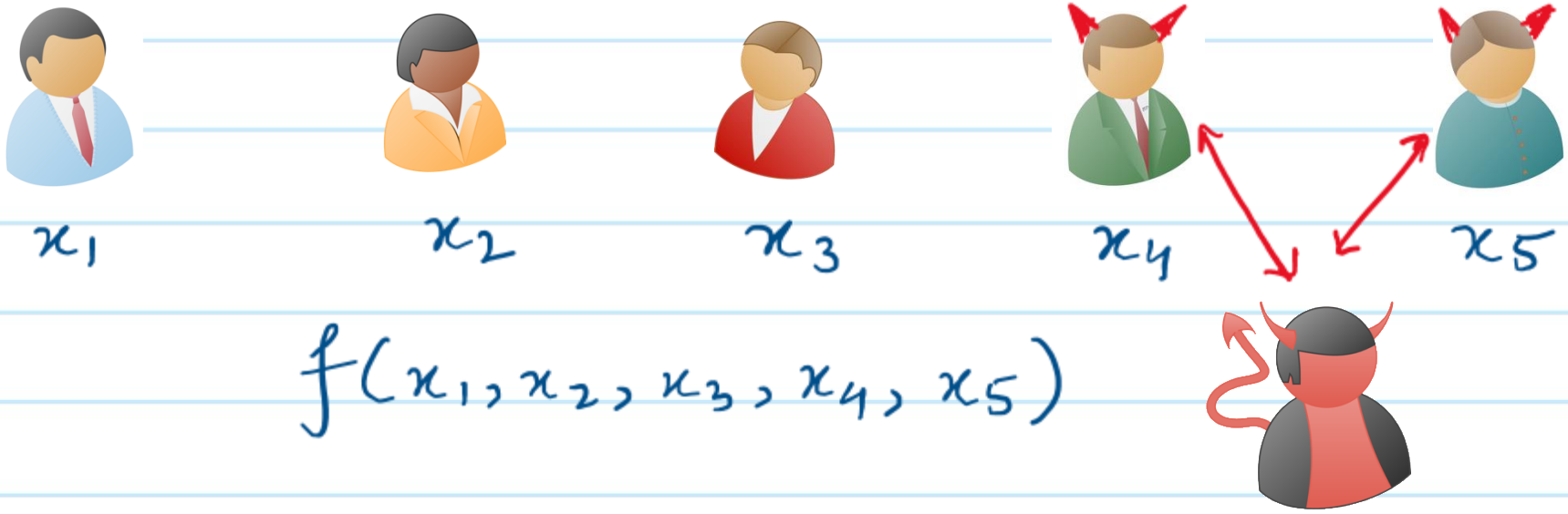
# Agenda

→ Defining maliciously secure multiparty computation

Announcement: HW4 will be released on March 14 and will be due on March 26

# Secure Multi-Party Computation : Setting

Parties: $P_1$ $P_2$ $P_3$ $P_4$ $P_5$



Inputs: $x_1$ $x_2$ $x_3$ $x_4$ $x_5$

$$f(x_1, x_2, x_3, x_4, x_5)$$

→ Suppose t out of n parties are corrupted by a monolithic *adversary*

→ Parties want to securely compute f on their joint inputs

So far we have only considered semi-honest adversaries

# Difference Between Semi-Honest and Malicious Adversaries

## Semi-Honest

- Passive/Honest-but-curious adversary
- Follows the protocol honestly

## Malicious

- Active adversary
- May arbitrarily deviate from the protocol description

# Example: What could go wrong in the presence of a Malicious Adversary?

(Semi-Honest BGW/ GMW)

* Input Sharing: The corrupt parties may send garbage values instead of honestly computed shares of their inputs.

* Circuit Evaluation: The corrupt parties may arbitrarily deviate from the protcol to violate privacy or lead to incorrect evaluation of the circuit.

* Output Reconstruction: The corrupt parties may wait to see the output shares sent by honest parties and then send garbage values or not send anything at all.

# Difference Between Semi-Honest and Malicious Adversaries

## Semi-Honest

→ Passive/Honest-but-curious adversary

→ Follows the protocol honestly

* Uses its input to honestly participate in the protocol
* The output of the protocol is the correct output of the function.

## Malicious

→ Active adversary

→ May arbitrarily deviate from the protocol description

* May not use any specific input to participate and use an arbitrary strategy.
* The protocol may not be able to output anything if the adversary mis-behaves. or the adversary could learn the output but prevent honest parties from learning the output.

# Shortcomings of the Definition of Semi-Honest MPC when dealing with a malicious adversary.

**Definition:** A protocol $\pi$ securely computes a function $f$ in the semi-honest model, if $\exists$ a PPT simulator algorithm $S$, s.t., $\forall$ $t$-sized subset $C \subset [n]$ of corrupt parties, for any security parameter $\lambda$ & $\forall$ inputs $x_1, \ldots, x_n$, it holds that:

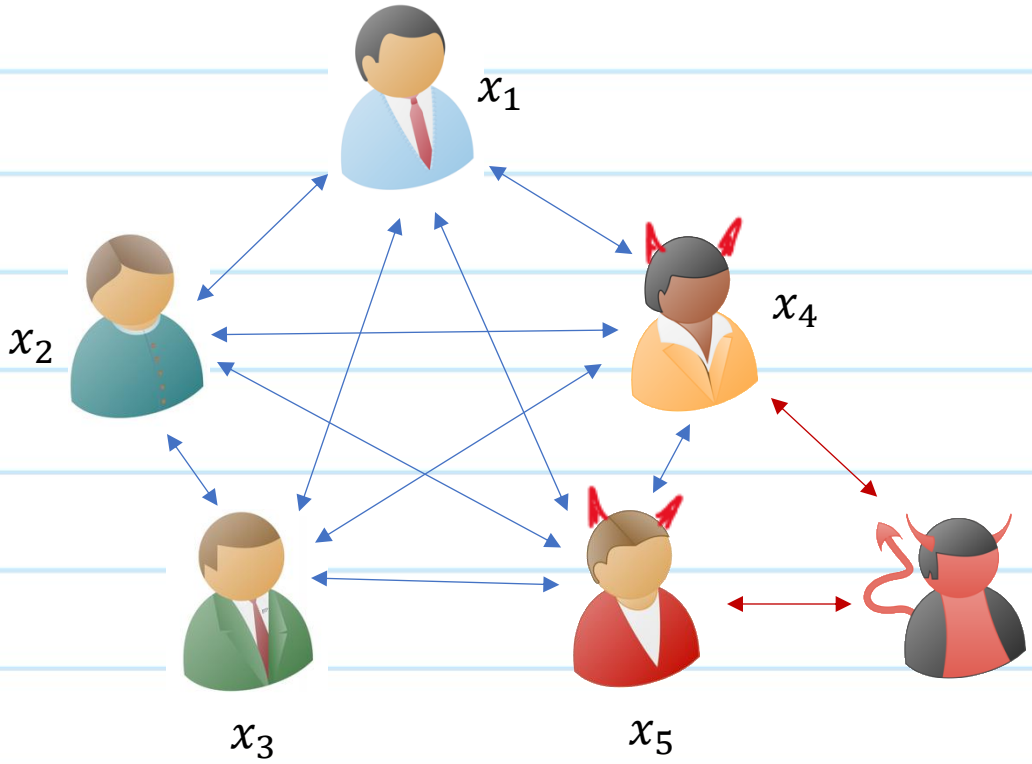
$$\{ S(\{x_i\}_{i \in C}, f(x_1, \ldots, x_n), f(x_1, \ldots, x_n)) \} \approx_c \{ \underbrace{\text{View}_C(\pi)}_{\text{View of Adv}}, \underbrace{\text{Out}_{[n] \setminus C}(\pi)}_{\text{output of honest parties}} \}$$

if the adv misbehaves, its inputs may not be well-defined

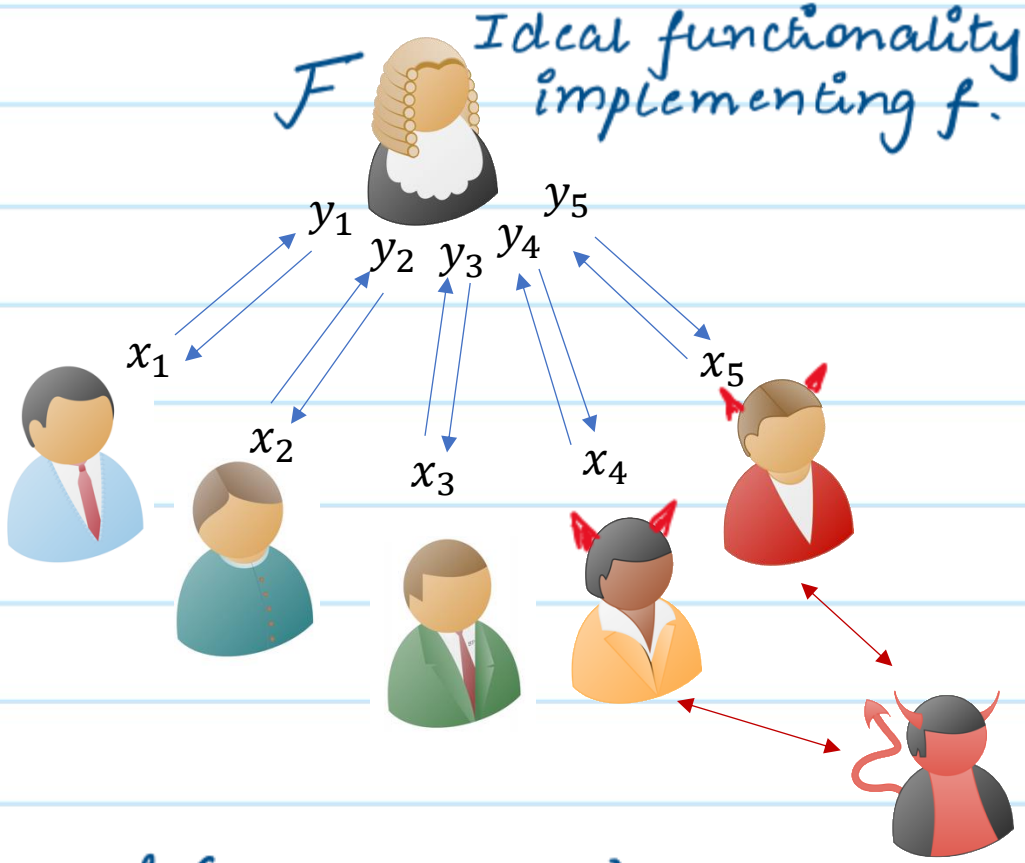
How is this output defined?

# Security for MPC: A new perspective

## Protocol in the Real world




## Ideal World




$$(y_1, \cdots y_5) = f(x_1, \cdots, x_5)$$

# Security for MPC: A new perspective

→ A malicious adversary corrupting at most t out of the n parties learns nothing about the inputs of the honest parties beyond what it learns from interacting with the ideal functionality.

→ Honest parties parties follow the protocol according to a well-defined input, which can also be given to the ideal functionality in the ideal world.

→ Input of the corrupt parties is not well-defined in the in the real world

What input should be given to the ideal functionality in the ideal world?

# Input Extraction

→ Intuitively, in a secure protocol, whatever an adversary can do in the real world should also be achievable in the ideal world by some suitable choice of inputs for the corrupt parties.

→ Therefore, we leave it to the simulator to choose inputs for the corrupt parties.

→ This aspect of simulation is called extraction, since the simulator extracts an effective ideal-world input from the real-world adversary that explains the input's real-world effect.

→ In most constructions, it is sufficient to consider black-box simulation, where the simulator is given access only to the oracle implementing the real-world adversary, not its code.

# Formalizing the Requirements of a Maliciously Secure MPC in the Real-Ideal World Paradigm

Let $C \subset [n]$ be $t$-sized subset of corrupt parties. We define two distributions:

1 $\text{Real}_{\pi,A}(\lambda, \{x_i\}_{i \notin C})$: Run the protocol using $\lambda$ as the security parameter & $\{x_i\}_{i \in C}$ as the inputs of the honest parties. The messages of corrupt parties are chosen based on $A$. Let $y_i$ denote the output of each honest party $P_i$ & $\text{View}_i$ denote the view of each $P_i$ in this protocol.

Output: $\{\{\text{view}_i\}_{i \in C}, \{y_i\}_{i \notin C}\}$

Let $Sim^A$ be a PPT algorithm given oracle access to A.

2. $Ideal_{F, Sim^A}(\lambda, \{x_i\}_{i \notin C})$: Run $Sim^A$ until it outputs $\{x_i\}_{i \in C}$, compute $(y_1, \ldots, y_n) \leftarrow F(x_1, \ldots, x_n)$. Then, give $\{y_i\}_{i \in C}$ to $Sim^A$. Let $\{view^*_i\}_{i \in C}$ denote the final output of $Sim^A$.

Output: $\{\{view^*_i\}_{i \in C}, \{y_i\}_{i \notin C}\}$

# Defining Maliciously Secure MPC

**Definition:** A protocol $\pi$ securely realizes $\mathcal{F}$ in the presence of malicious adversaries, if $\exists$ a PPT simulator algorithm Sim, such that $\forall$ PPT malicious adversaries $\mathcal{A}$ corrupting any $t$-sized subset $C \subset [n]$ of the parties, $\forall \lambda \in \mathbb{N}$ and $\forall \{x_i\}_{i \notin C}$, the following two distributions are computationally / statistically / perfectly indistinguishable:

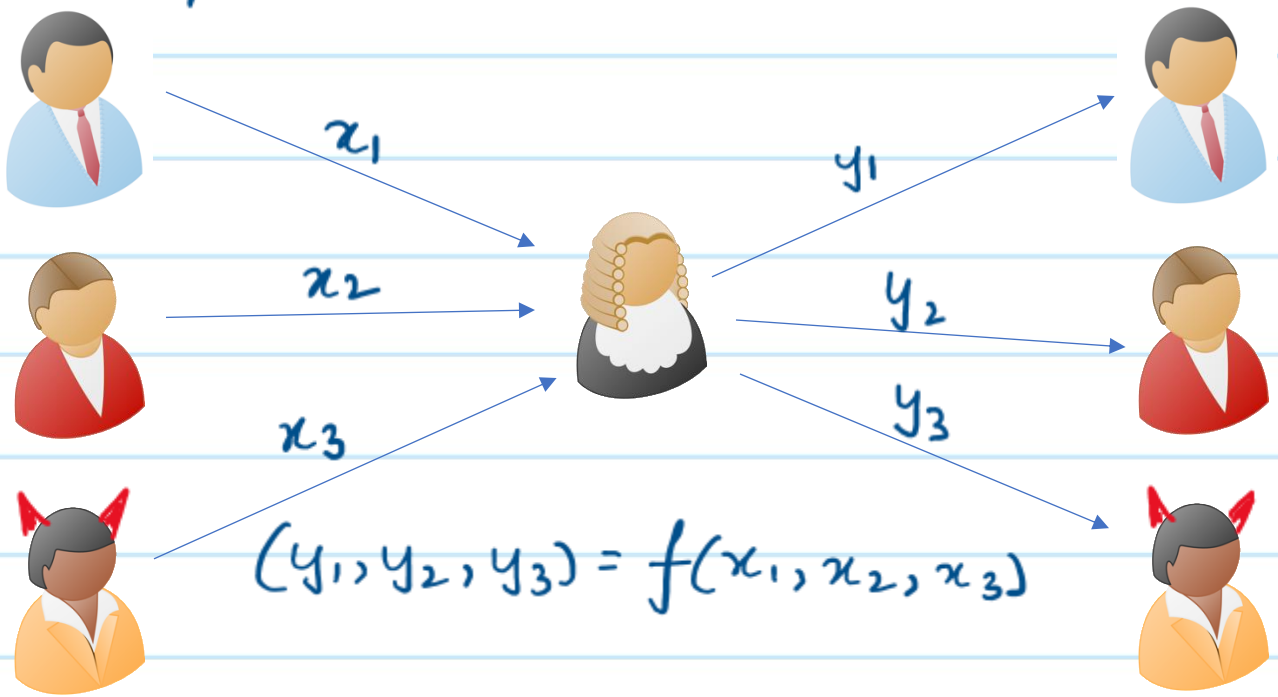
$$\text{Real}_{\pi, \mathcal{A}}(\lambda, \{x_i\}_{i \notin C})$$

$$\text{Ideal}_{\mathcal{F}, \text{Sim}^{\mathcal{A}}}(\lambda, \{x_i\}_{i \notin C})$$

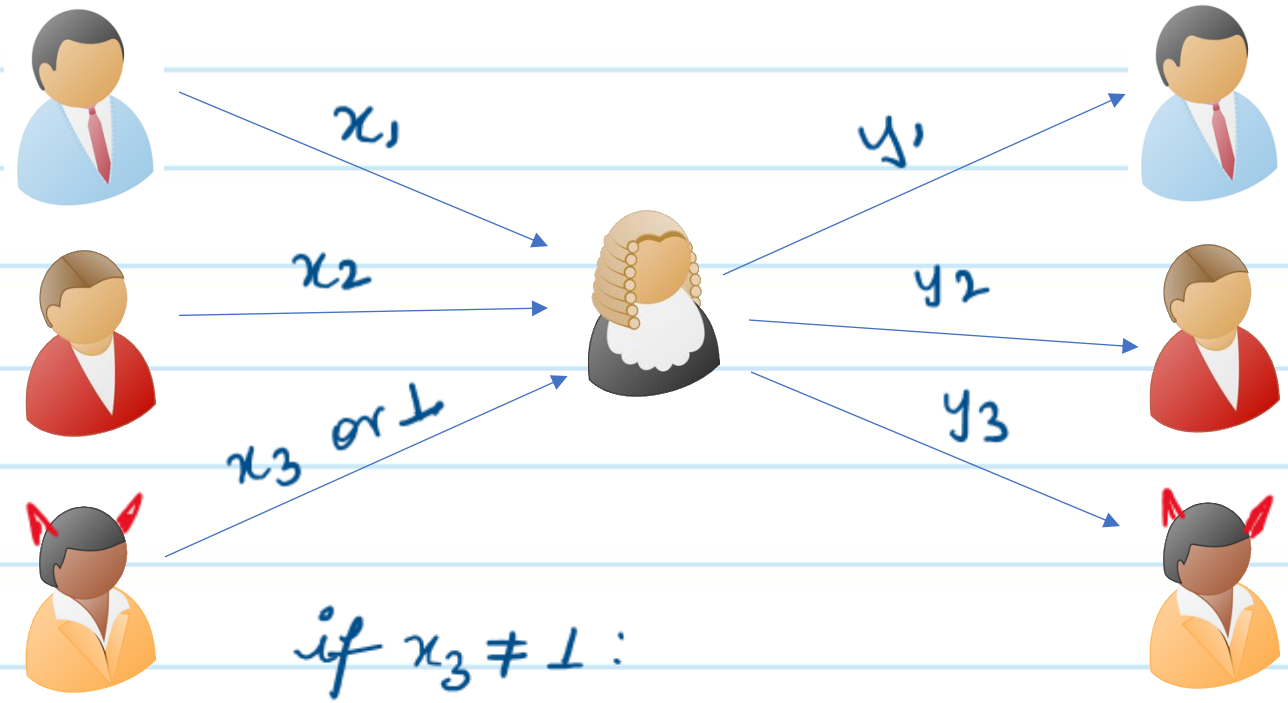
# Variants of Ideal Functionality

Different types of ideal functionalities can capture different privacy / robustness guarantees.

I **Guaranteed Output Delivery**: All parties are guaranteed to get the correct output.



II **Fairness**: Either all parties get the correct output or no one gets any output




if $x_3 \neq \perp$ :

$$(y_1, y_2, y_3) = f(x_1, x_2, x_3)$$

else: $y_1 = y_2 = y_3 = \perp$

III **Security with abort**: The adversary can learn the output & then decide whether or not the honest parties should get the output.