

# FAIR AND EFFICIENT SECURE MULTIPARTY COMPUTATION WITH REPUTATION SYSTEMS

---

Gilad Asharov

Yehuda Lindell

**Hila Zarosim**



Bar-Ilan University  
Tradition of Excellence

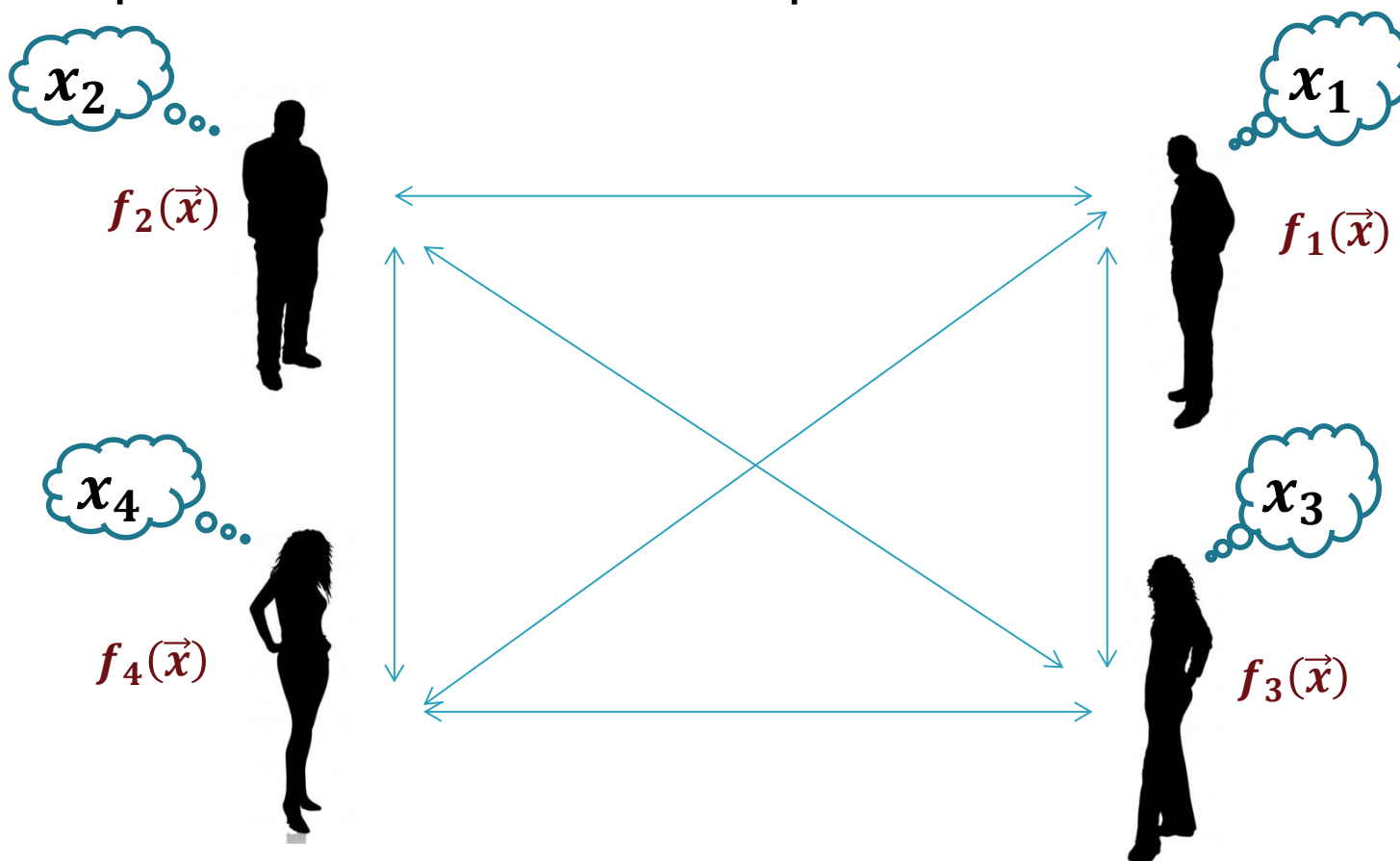
Asiacrypt 2013

# Secure Multi-Party Computation

- A set of parties who don't trust each other wish to compute a function of their inputs

# Secure Multi-Party Computation

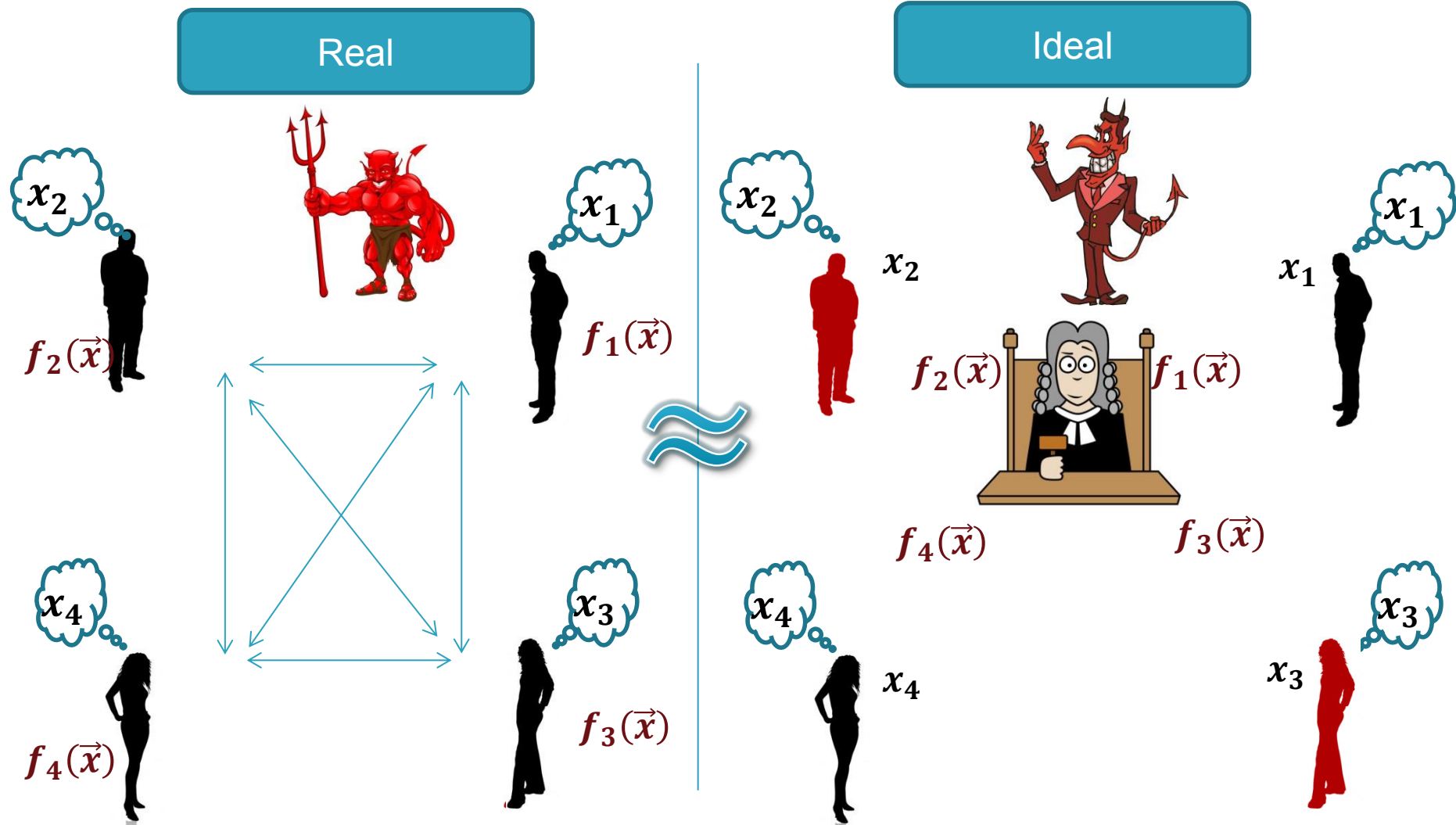
- A set of parties who don't trust each other wish to compute a function of their inputs



# Secure Multi-Party Computation

- A set of parties who don't trust each other wish to compute a function of their inputs
- Security:
  - Correctness
  - privacy
  - fairness
  - and more...

# Security Definition



# Secure Computation

Do secure protocols exist?

How many parties should remain honest to ensure the security of the protocols?

# Known Results

Honest majority is *not* guaranteed

Impossible to achieve security with *fairness* in general

There exist protocols that guarantee security except for fairness

Honest majority is guaranteed

There exist protocols with full security

- These protocols guarantee no security whatsoever when there is no honest majority

The parties have to “guess” in advance whether there is going to be honest majority

What if they are wrong?

# Really?

- Do parties really have no information about the likelihood of other parties playing honestly?
- Do you trust everyone equally?





# Reputations

- We usually do have some information about the honesty of the participants
- This information is based on their previous behavior
- We denote this by “the reputation of the party”

**Can we use the parties' reputation  
in secure computation?**

# Reputation Systems

- Systems that aim to predict the players' behavior
  - Based on the transactions history
- Formally, a reputation vector is a vector of probabilities  $(r_1, \dots, r_m)$  such that  $r_i$  represents **the probability that  $P_i$  plays honestly**
- This is a public information



# Reputation Systems

- Systems that aim to predict the players' behavior
  - Based on the transactions history
- Formally, a reputation vector is a vector of probabilities  $(r_1, \dots, r_m)$  such that  $r_i$  represents **the probability that  $P_i$  plays honestly**
- This is a public information
- There is a considerable amount of literature on how to construct and maintain these systems

# Reputation Systems and Secure Computation

We ask the following question:

**Can reputation systems be utilized in order to achieve *fair* and efficient secure multiparty computation?**

**On what conditions on the reputation system, is it possible to obtain *fair* secure multiparty computation?**

# Our Contributions

- We formally define security in this model
- We provide almost tight feasibility and infeasibility results for when it is possible to obtain *fair* secure multiparty computation

***Very informally:* There exist *fair* secure protocols for all functionalities if and only if the number of parties with  $r_i > \frac{1}{2}$  is *superlogarithmic* in  $n$**

# Our Contributions

- We consider both “independent” and “correlated” reputations
  - Does the probability that a party is corrupted depend on the probability that other parties are corrupted?
- We show that when the dependence between the reputations is limited, it is possible to obtain fair secure computation

# The Model

- Usually in secure computation the number of players is fixed. **In our model, this is a parameter of  $n$** 
  - We construct protocols that are secure as long as the probability that a subset of players plays honestly is  $1 - \text{negl}(n)$
  - This probability depends on the number of players and hence the number of players must be a parameter of  $n$ , we denote this by  $m(n)$
- We consider families of functionalities to enable a various number of players
- Security definition is almost the same as standard:
  - The choice of corrupted parties is done according to the reputation vector and it part of the real world and ideal world ensembles

# Feasibility

## Observation:

**If there exists a subset of players with honest majority, then a secure protocol exists [DY05]**

1. All parties send shares of their inputs to the subset
2. The subset carries out the computation and sends shares of the output to the parties



# Feasibility

## Observation:

If there exists a subset of players with honest majority, then a secure protocol exists [DY05]

Based on the reputation vector, what's the probability that there exists a subset with honest majority?

# Feasibility- Criteria

- We characterize the reputation system for which a subset with an honest majority exists with probability  $1 - \text{negl}(n)$
- For a subset  $T$  of players, we use the Hoeffding\* Inequality to compute the probability that the number of corrupted parties in  $T$  is  $< \frac{|T|}{2}$

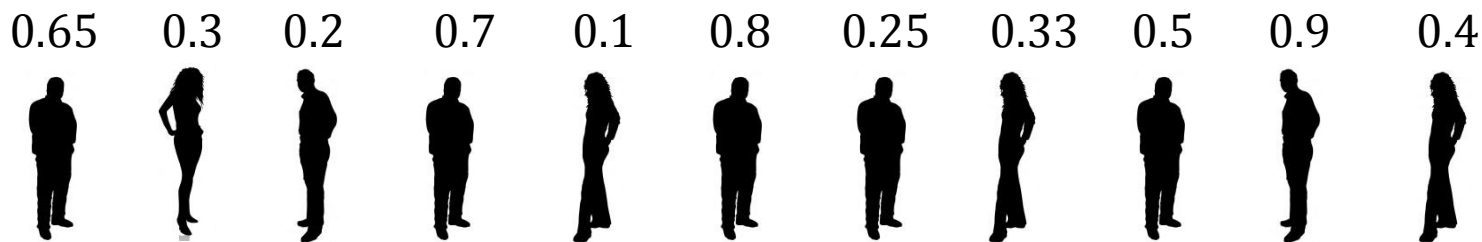
\* The Hoeffding Inequality gives an upper bound on the probability that the sum of random variables deviates from the expected sum

# Feasibility- Criteria

- For every  $n$  and a subset  $T_n$  of the players, let

$$\Delta_{T_n} = \sum_{i \in T_n} r_i - \frac{|T_n|}{2}$$

- $\Delta_{T_n}$  is the distance of the expected # of honest parties in  $T_n$  from half



# Feasibility- Criteria

- For every  $n$  and a subset  $T_n$  of the players, let

$$\Delta_{T_n} = \sum_{i \in T_n} r_i - \frac{|T_n|}{2}$$

- $\Delta_{T_n}$  is the distance of the expected # of honest parties in  $T_n$  from half



$$\sum_{i \in T_n} r_i = 0.65 + 0.2 + 0.8 + 0.5 = 2.15$$

# Feasibility- Criteria

- For every  $n$  and a subset  $T_n$  of the players, let

$$\Delta_{T_n} = \sum_{i \in T_n} r_i - \frac{|T_n|}{2}$$

- $\Delta_{T_n}$  is the distance of the expected # of honest parties in  $T_n$  from half



$$\sum_{i \in T_n} r_i = 0.65 + 0.2 + 0.8 + 0.5 = 2.15$$

$$\frac{|T_n|}{2} = 2$$

$$\Delta_{T_n} = 0.15$$

# Feasibility- Criteria

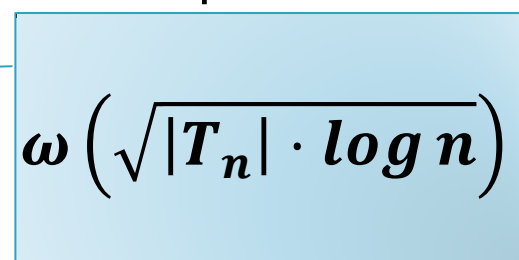
- For every  $n$  and a subset  $T_n$  of the players, let

$$\Delta_{T_n} = \sum_{i \in T_n} r_i - \frac{|T_n|}{2}$$

- $\Delta_{T_n}$  is the distance of the expected # of honest parties from half
- **Thm:** If there exists a series of subsets  $\{T_n\}_{n \in \mathbb{N}}$  such that

$$\Delta_{T_n} \geq \epsilon$$

Then there exists a secure protocol with respect to Rep.


$$\omega\left(\sqrt{|T_n| \cdot \log n}\right)$$

# Efficiently Finding The Subset

- We have a secure protocol assuming that for every  $n$ , such a subset  $T_n$  exists

How can the parties know that such a set exists?

How can the parties efficiently find the appropriate subset?

- We give an efficient algorithm for finding the subset
  - It is a greedy algorithm that sorts the reputations and finds a set with large enough ratio between  $\Delta_T$  and  $|T|$
  - See the paper for details

# Infeasibility

- We show a condition on the reputation system such that it is not possible to achieve secure computation with ***fairness***
  - Achieving security without fairness is possible with any number of corruptions
- We focus on the coin-tossing functionality:
  - Thm[Cleve86]: It is impossible to **toss a fair coin with only two-parties**
- We show how to **reduce** a multi-party coin-tossing with a reputation system that **fulfills our criteria** to a **two-party coin-tossing**



# Infeasibility – The Idea

- Fix  $n$  and let  $H_n$  be the set of parties with reputation more than  $\frac{1}{2}$ 
  - These parties are more likely to play honestly than dishonestly
- Assume that  $H_n$  is empty
  - Every party is more likely to play dishonestly
- The expected number of corrupted parties is at least  $\frac{m}{2}$
- *Intuitively*, every protocol secure with such a reputation system is secure with dishonest majority
  - We show that this implies a fair 2-party protocol for coin-tossing

# Infeasibility

- **Thm:**

Let  $Rep$  be a reputation system.

If for infinitely many  $n$ 's:

**the probability that all parties in  $H_n$  are corrupted is**

**at least  $\frac{1}{p(n)}$ ,**

then it is impossible to securely compute the coin-tossing functionality with respect to  $Rep$ .

parties that are more likely to play honestly than dishonestly

# Proof Idea

- For simplicity assume  $Rep$  s.t.  $H_n$  is empty for  $\infty$   $n$ 's

$$\Pi = \langle P_0, P_1, \dots, P_m \rangle$$

$m$ -party protocol  
with respect to  
 $Rep$



$$\pi' = \langle P'_0, P'_1 \rangle$$

2-party protocol

- We give a simplified idea of the reduction
  - The actual proof involves many technicalities
  - See the paper

# Proof Idea

$\Pi = \langle P_1, P_1, \dots, P_m \rangle$   
 $m$ -party protocol  
with respect to  $Rep$

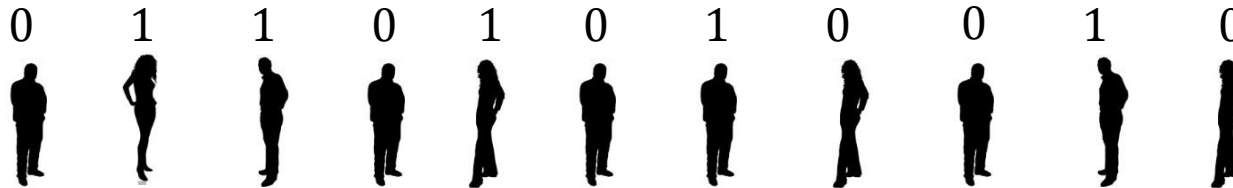


$\pi' = \langle P'_0, P'_1 \rangle$   
2-party protocol

$P'_0$

$P'_1$

Jointly toss  $m$  coins (without fairness)



# Proof Idea

$\Pi = \langle P_1, P_1, \dots, P_m \rangle$   
 $m$ -party protocol  
with respect to  $Rep$

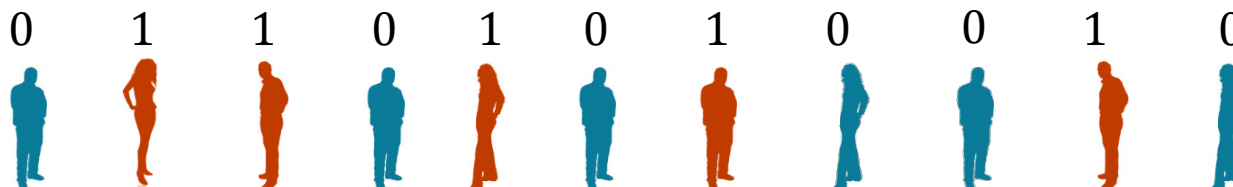


$\pi' = \langle P'_0, P'_1 \rangle$   
2-party protocol

$P'_0$

$P'_1$

Jointly toss  $m$  coins (without fairness)



Emulate  $\Pi$

$P'_0$  and  $P'_1$  determine their outputs  
according to the outputs of the virtual  
parties under their control

# Proof Idea

$\Pi = \langle P_1, P_1, \dots, P_m \rangle$   
 $m$ -party protocol  
with respect to  $Rep$



$\pi' = \langle P'_0, P'_1 \rangle$   
2-party protocol

- If  $\Pi$  is secure when  $H_n$  is empty:
  - $\Pi$  can handle  $\geq \frac{m}{2}$  corrupted parties
- Each party in  $\Pi$  goes randomly to one of the 2 parties in  $\pi'$ 
  - We expect  $\frac{m}{2}$  parties to be under the control of each party in  $\pi'$
- If one of the parties in  $\pi'$  is corrupted
  - Then all parties under its control are corrupted
  - This should be around  $\frac{m}{2}$  parties
- By the security of  $\Pi$ , we conclude that  $\pi'$  is also secure

# The Relation Between the Feasibility and the Infeasibility

- Feasibility: There exists a series of subsets  $\{T_n\}_{n \in \mathbb{N}}$  such that  $\Delta_{T_n} > \omega(\sqrt{|T_n| \cdot \log n})$
- Infeasibility: For infinitely many  $n$ 's, the probability that all parties in  $H_n$  are corrupted is at least  $\frac{1}{p(n)}$

What is the relation between the feasibility and the infeasibility results?

# Tightness of Feasibility and Infeasibility

- **Thm:** For constant reputations, the feasibility and the infeasibility results are tight

For constant reputations, there exists a protocol for securely computing any family of functionalities **if and only if**  $|H_n| = \omega(\log n)$

A secure protocol exists if and only if there exists a superlogarithmic # of players that are more likely to play honestly



# Correlated Reputations

- When we considered independent reputations:
  - We needed a subset whose expected number of honest parties is more than a half (by some factor)
  - Does this suffice also for correlated reputations?
- Example:
  - $m$  parties
  - With probability  $\frac{1}{100}$ , only 1 party is honest
  - With probability  $\frac{99}{100}$ , all parties are honest
  - What is the expected number of honest parties?
  - Is this a “secure” subset?

$$\frac{1}{100} + \frac{99m}{100} = \frac{99m + 1}{100}$$

# Correlated Reputations

## Our Contributions

- We define security of protocol with respect to reputation systems with correlated reputations
- We define the notion of “limited dependence”
- We show that when the amount of dependence is small, it is possible to obtain fair secure computation
- See the paper for details

# Summary and Open Questions

- We define a new model for secure computation with reputation systems
- We give feasibility and infeasibility results for independent reputations
- We initiate the study of correlated reputations
  - There is still much to understand in this model
- We assume that such systems exist and maintained
  - An interesting open question is to use secure computation for constructing and maintaining reputation systems

**T H A N K**

**Y O U**