Multi Party Computation and Distributing Trust

MIT PROFESSIONAL EDUCATION

CSAIL
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

Multi Party Computation and Distributing Trust

Shafi Goldwasser

RSA Professor of EECS

Computer Science and Artificial Intelligence Laboratory (CSAIL)

Massachusetts Institute of Technology

MIT PROFESSIONAL EDUCATION

CSAIL
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

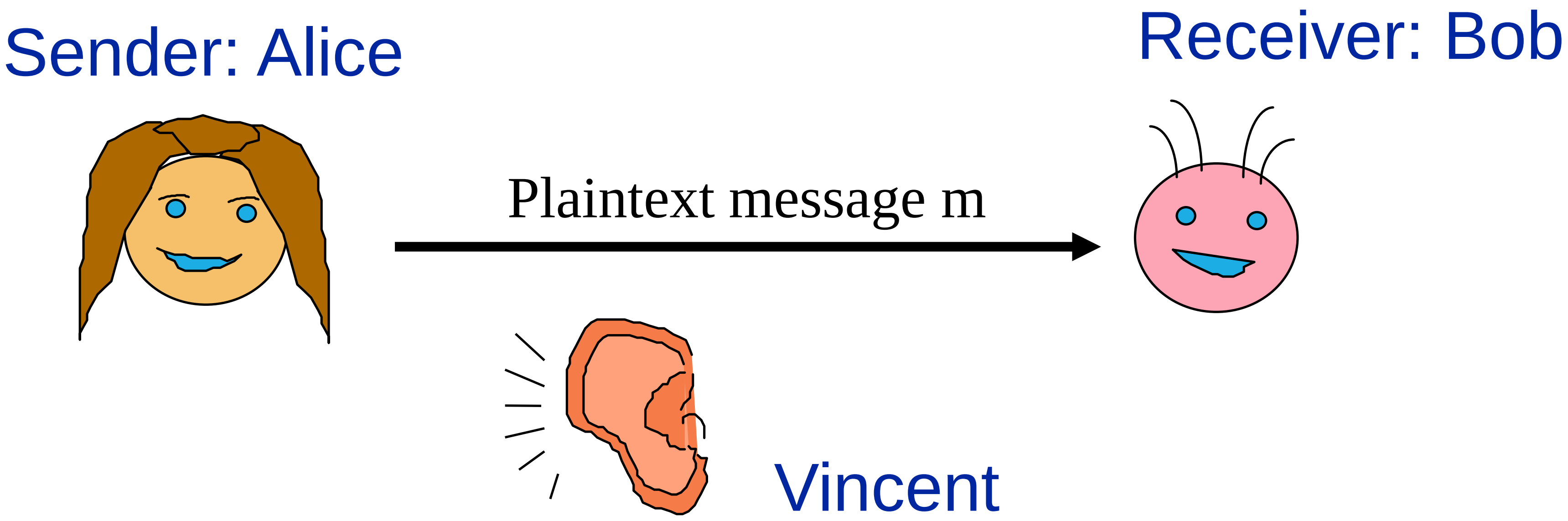
Unit 1

- From Communication to Computation
- Multi Party Computation
- Correctness, Privacy, Fairness
- Who is the adversary
- Definition of Privacy

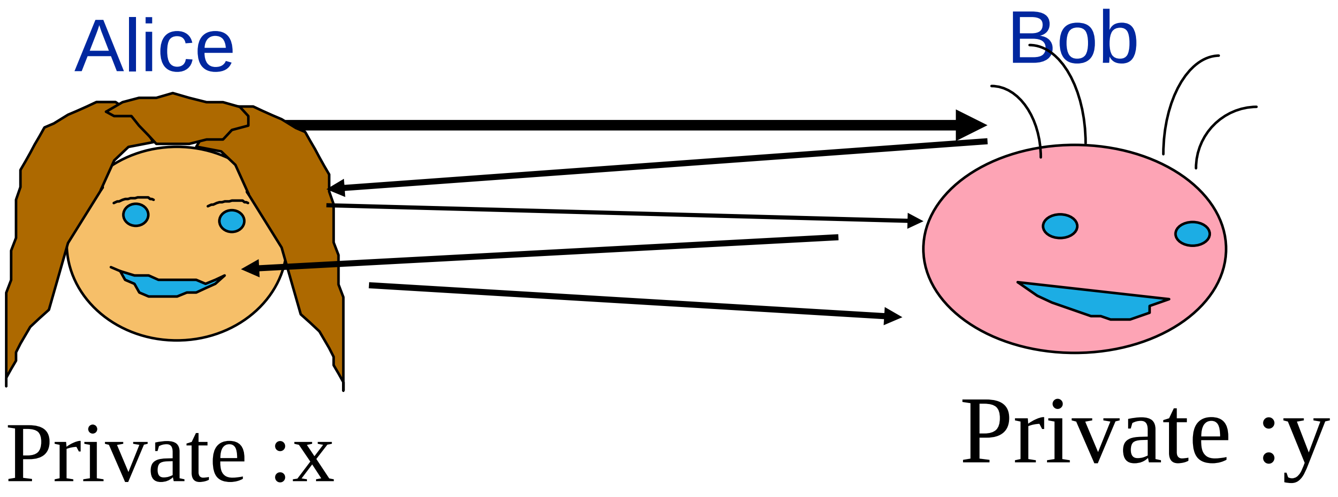
Classically: Secure Communication

Classical Cryptography addresses

privacy and authenticity of *communication*

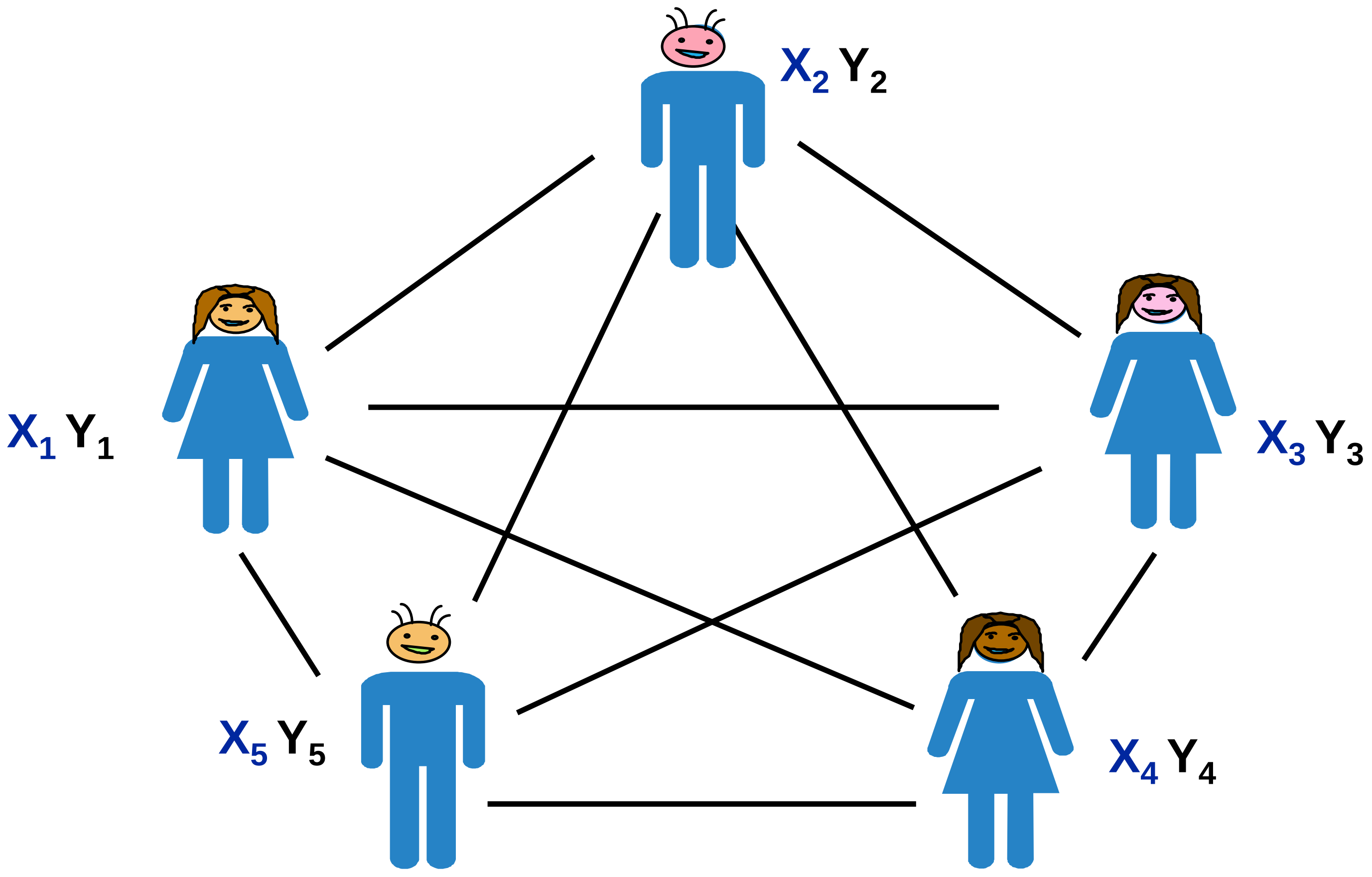


From Communication to Computation



New Goal: Compute arbitrary functions g(x,y) without revealing x or y to each other

Multi Party Secure Computation



N users

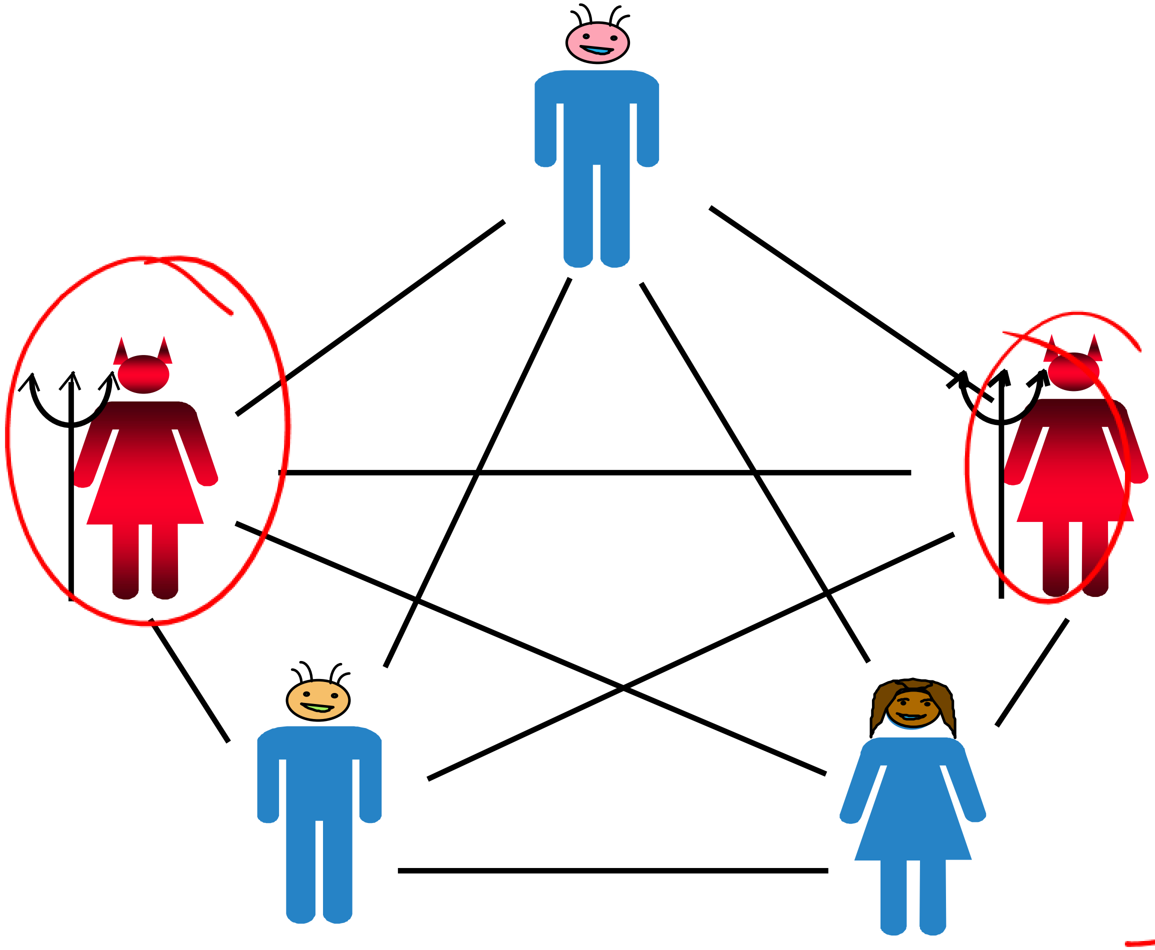
Each user/player
has private information X_i

New Goal: correctly compute

$$(Y_1 \ldots Y_n) = g\,(X_1 \ldots . X_n)$$

keeping X_i private from everyone else

Who is the Adversary?



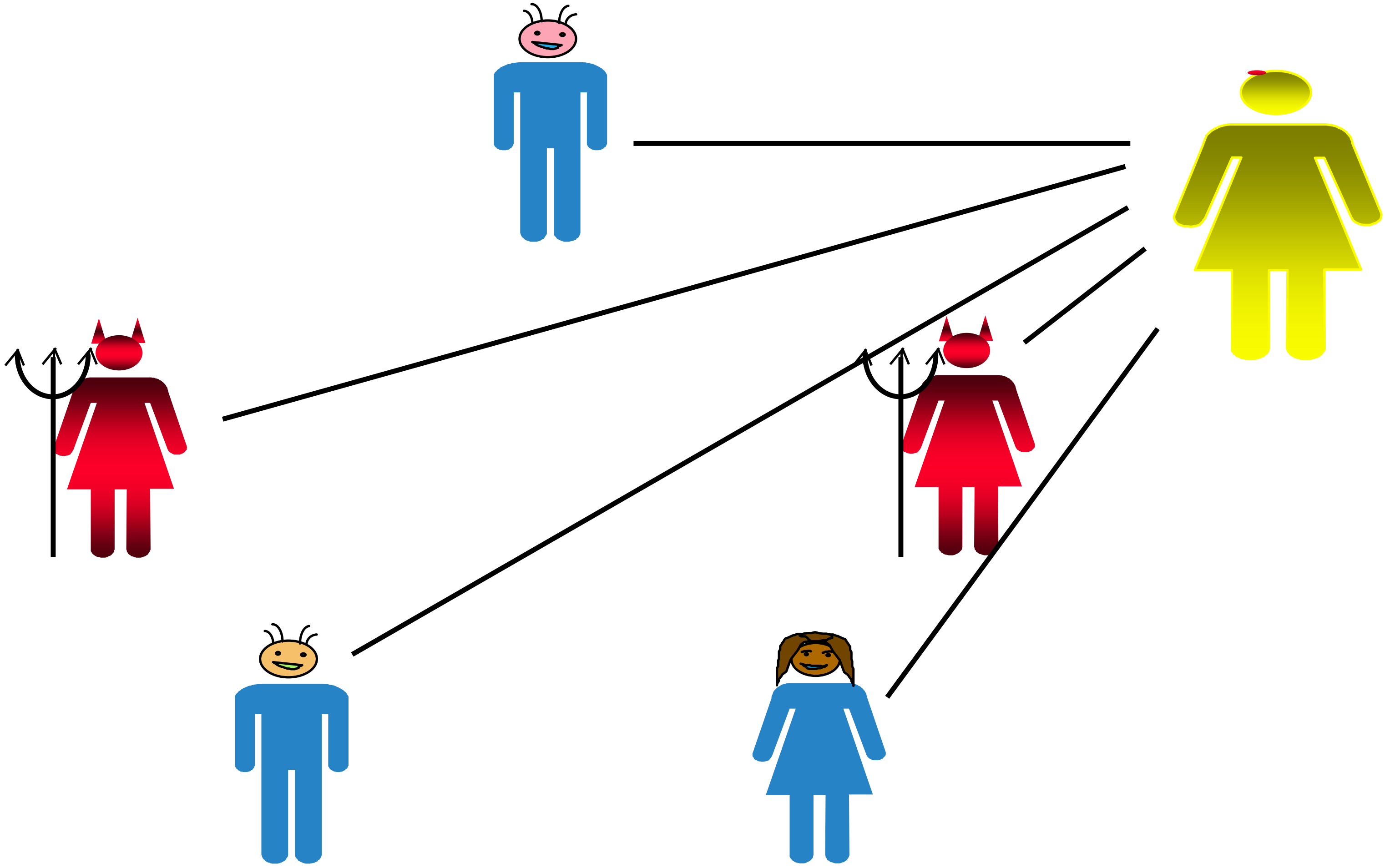
Subsets of colluding players:

- Honest but Curious
- Malicious
- Mobile

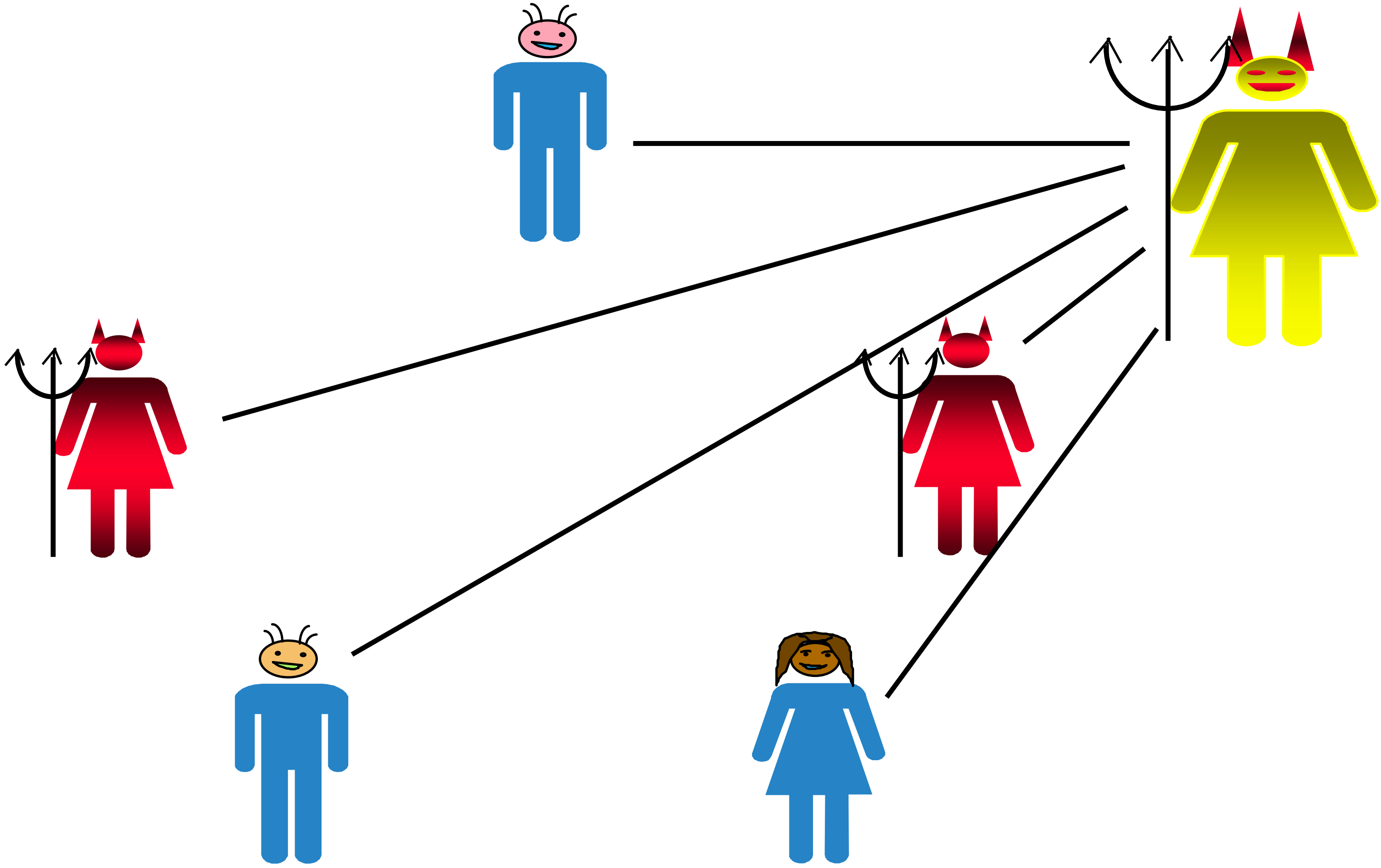
Globally coordinated

Zero Knowledge Proofs

Trusted Center Solution



Problem: Trusted Center May be Faulty

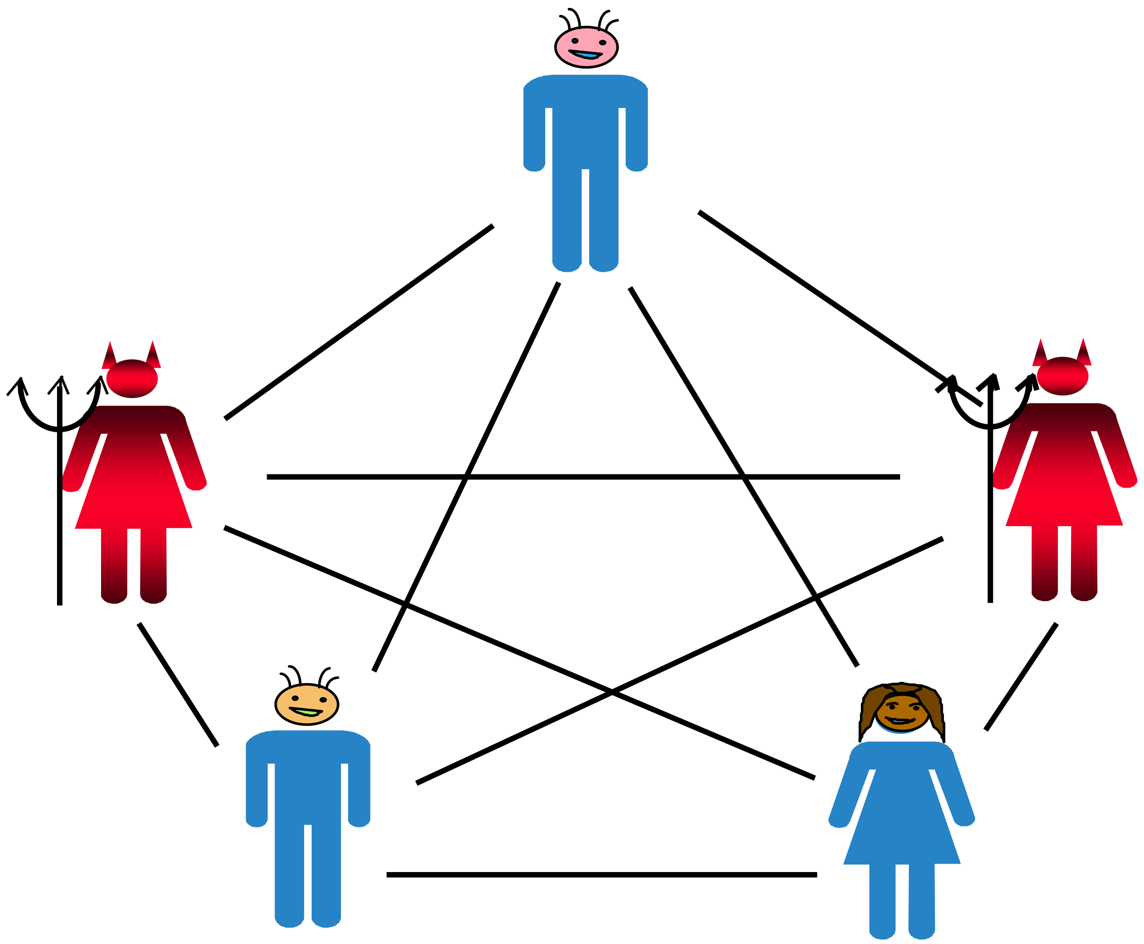


Centralized Authorities and Secrets Are Dangerous

Any single computer can be broken into and information can be spied on

Any single hardware component cannot be trusted to keep secrets

Secure Multi Party Computation



Compute

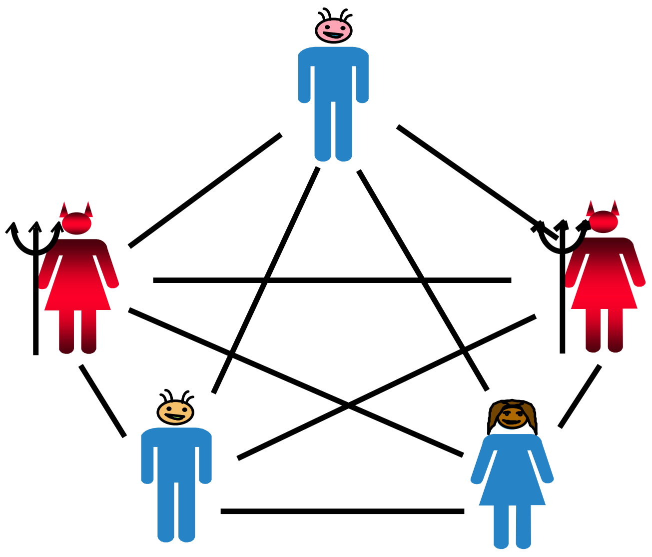
$$(Y_1 \ldots Y_n) = g\,(X_1 \ldots . X_n)$$

by a decentralized protocol *emulating the properties of the trusted center solution*

- correctness
- privacy
- Independence of inputs

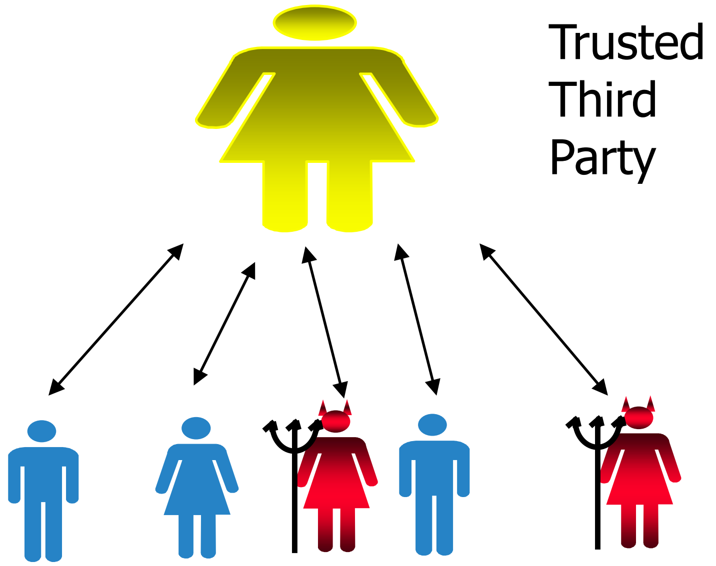
How to Define Privacy: Real versus Ideal Simulation Paradigm

Real World



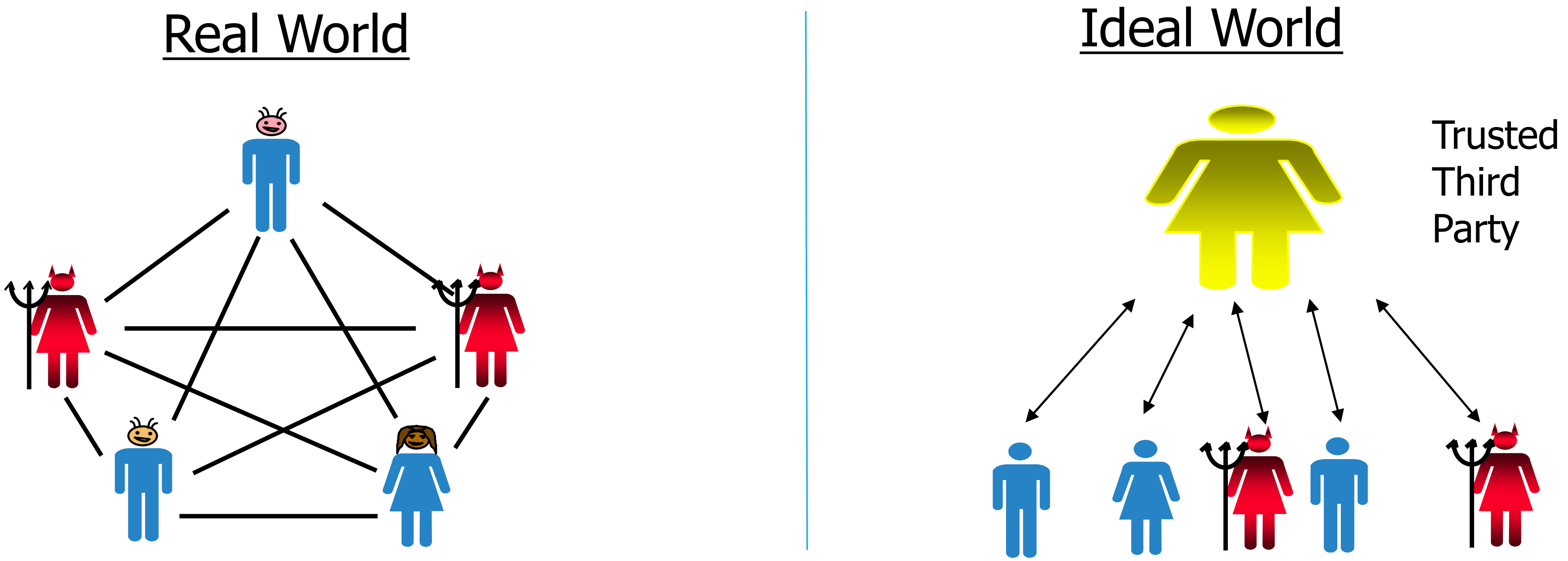
Adversary can corrupt parties, and Interact with honest parties via the protocol

Ideal World



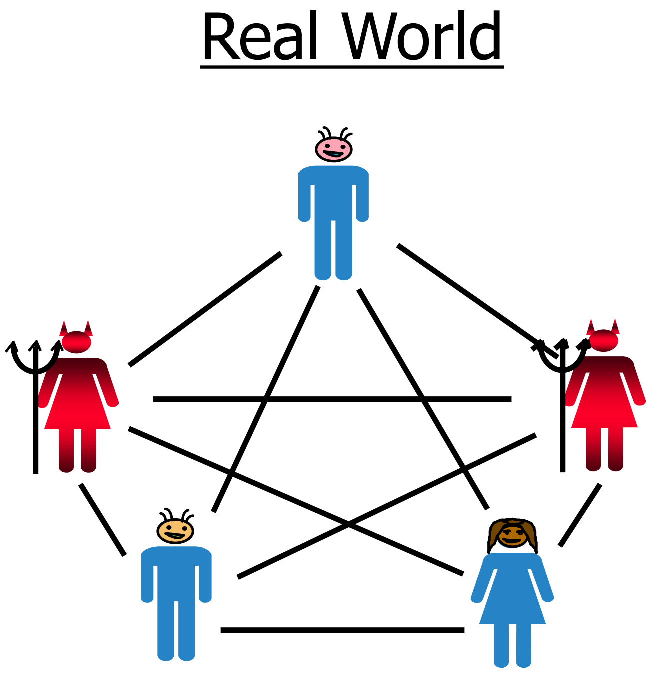
Adversary can only send input x_i, and receive output f(x1.,…,xn) From trusted center party

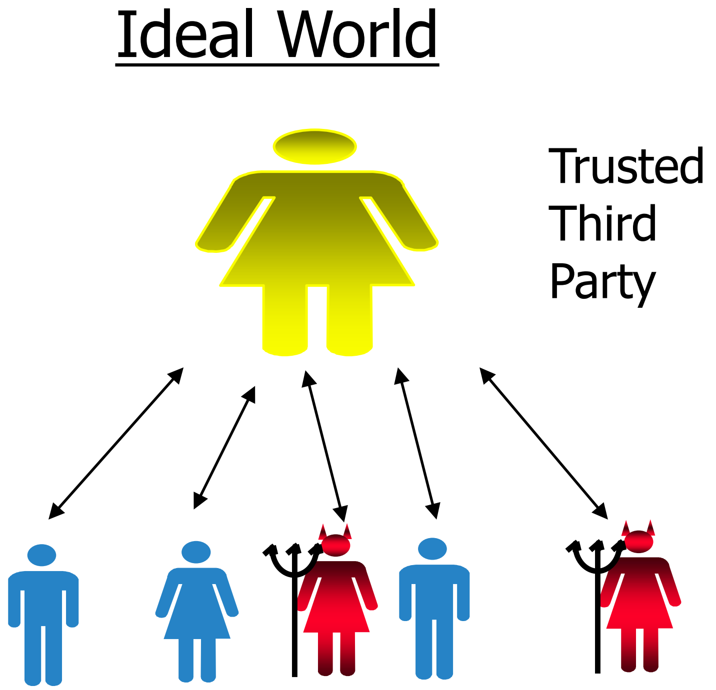
How to Define Privacy: Real versus Ideal Simulation Paradigm



We say that protocol $P=(P_1 \ldots P_n)$ is private if: for any adversary A in "real" world there exists an adversary A' in the "ideal" world s.t. whatever A can compute after the protocol, A' can as well.

How to Define Privacy: Real versus Ideal Simulation Paradigm



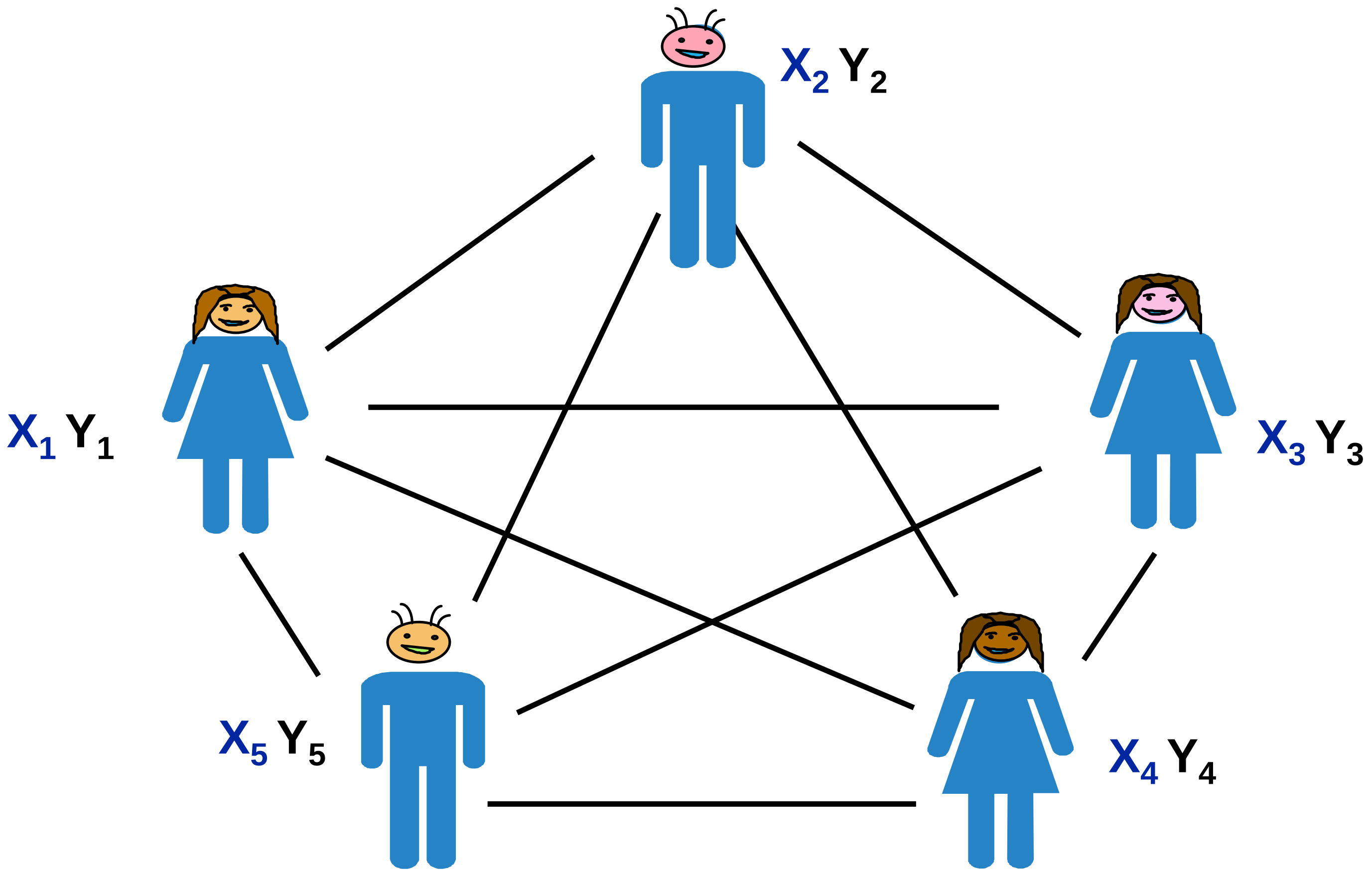


This implies that, A cannot compute anything more about the inputs of honest players than implied by the inputs and outputs of corrupted players

Unit 2

- Potential Applications of MPC
- Medical, Financial, Data Base intersection
- Usages in Practice

MPC is an All Encompassing Paradigm



$(y_1 \ldots y_n) = g(x_1 \ldots x_n)$

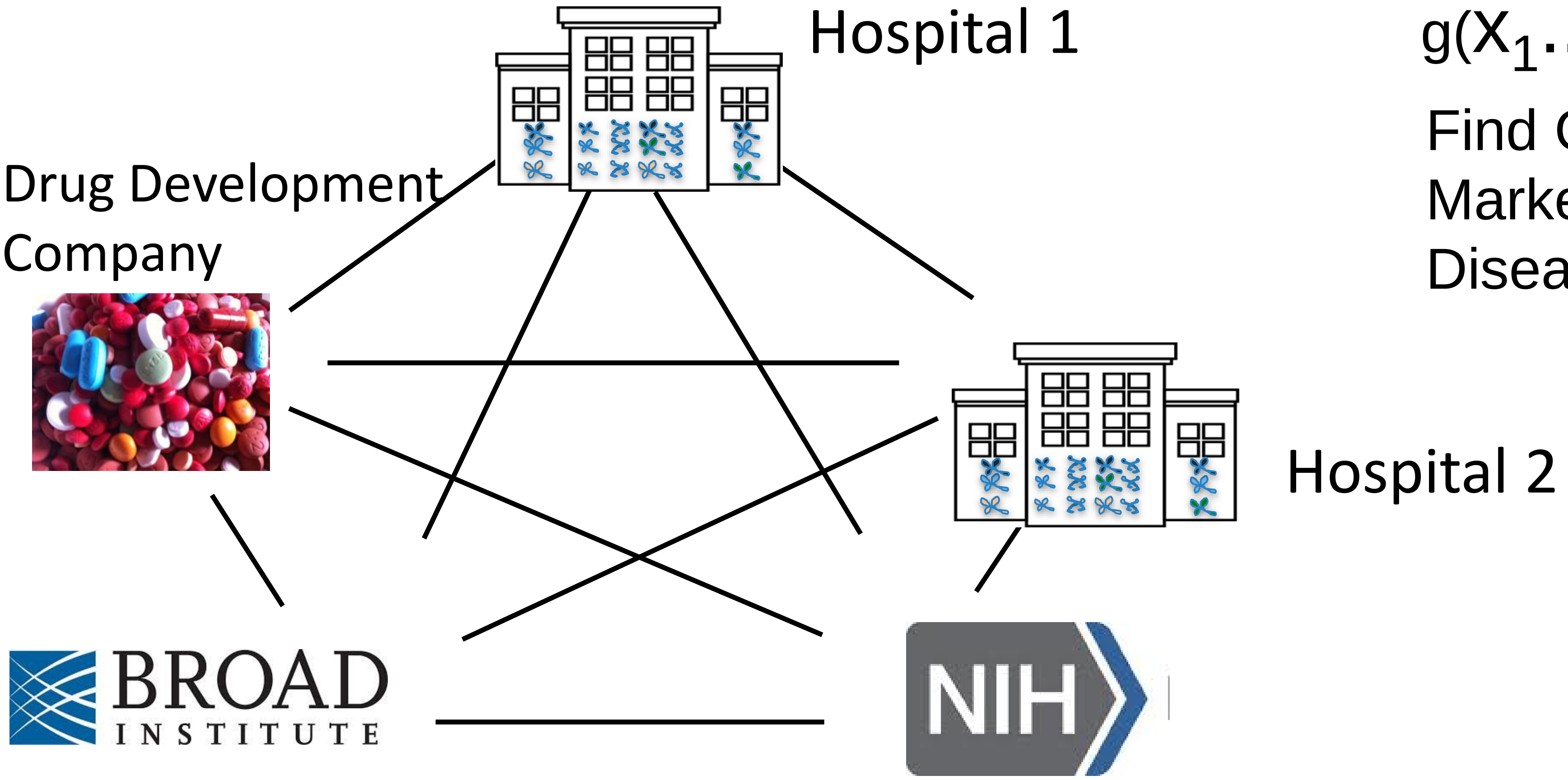
<u>g can be any computation:</u>

Electronic Election,

Auctions,

E-commerce applications.

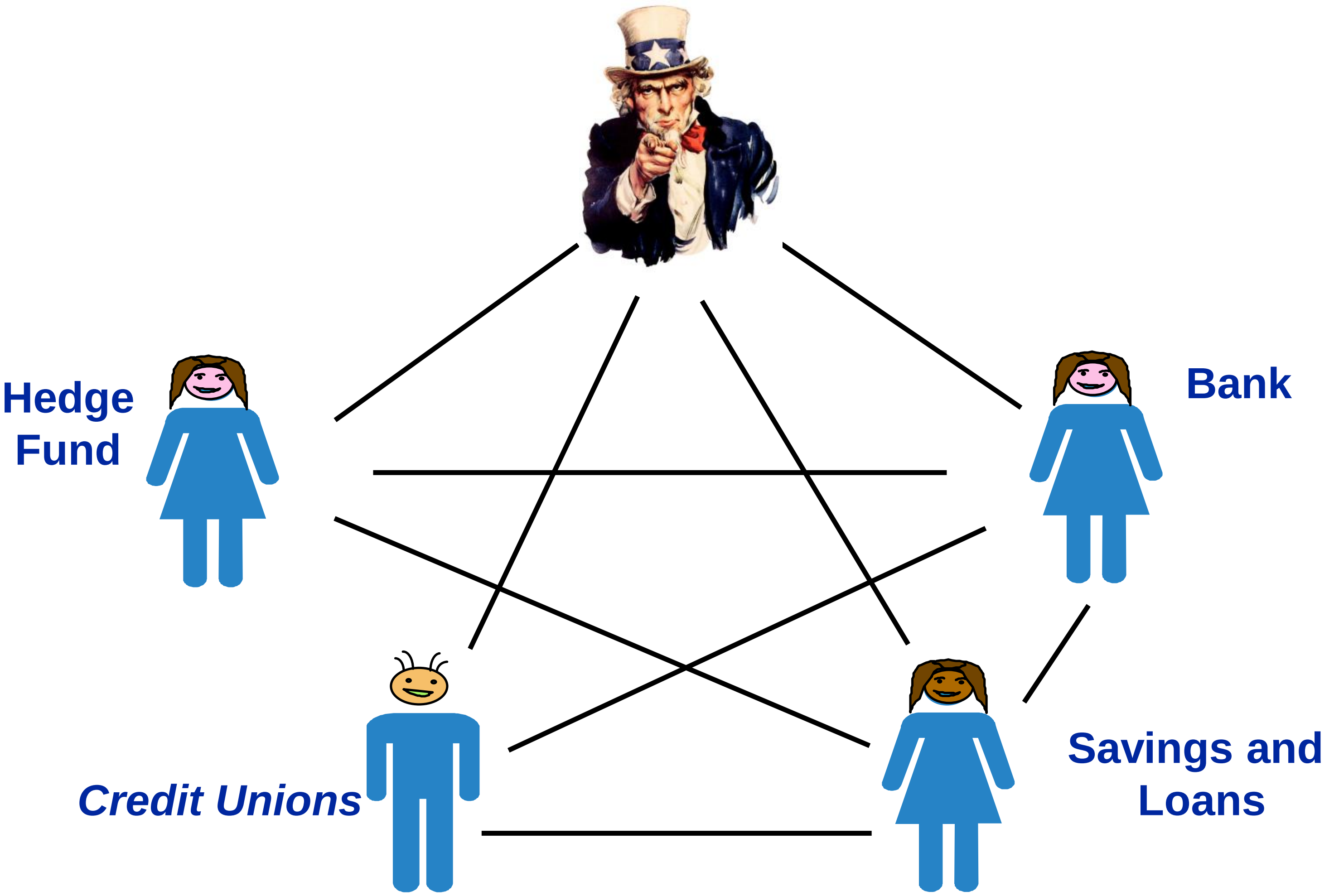
Medical Research



$g(X_1 .... X_n)=$

Find Genetic
Markers for
Diseases

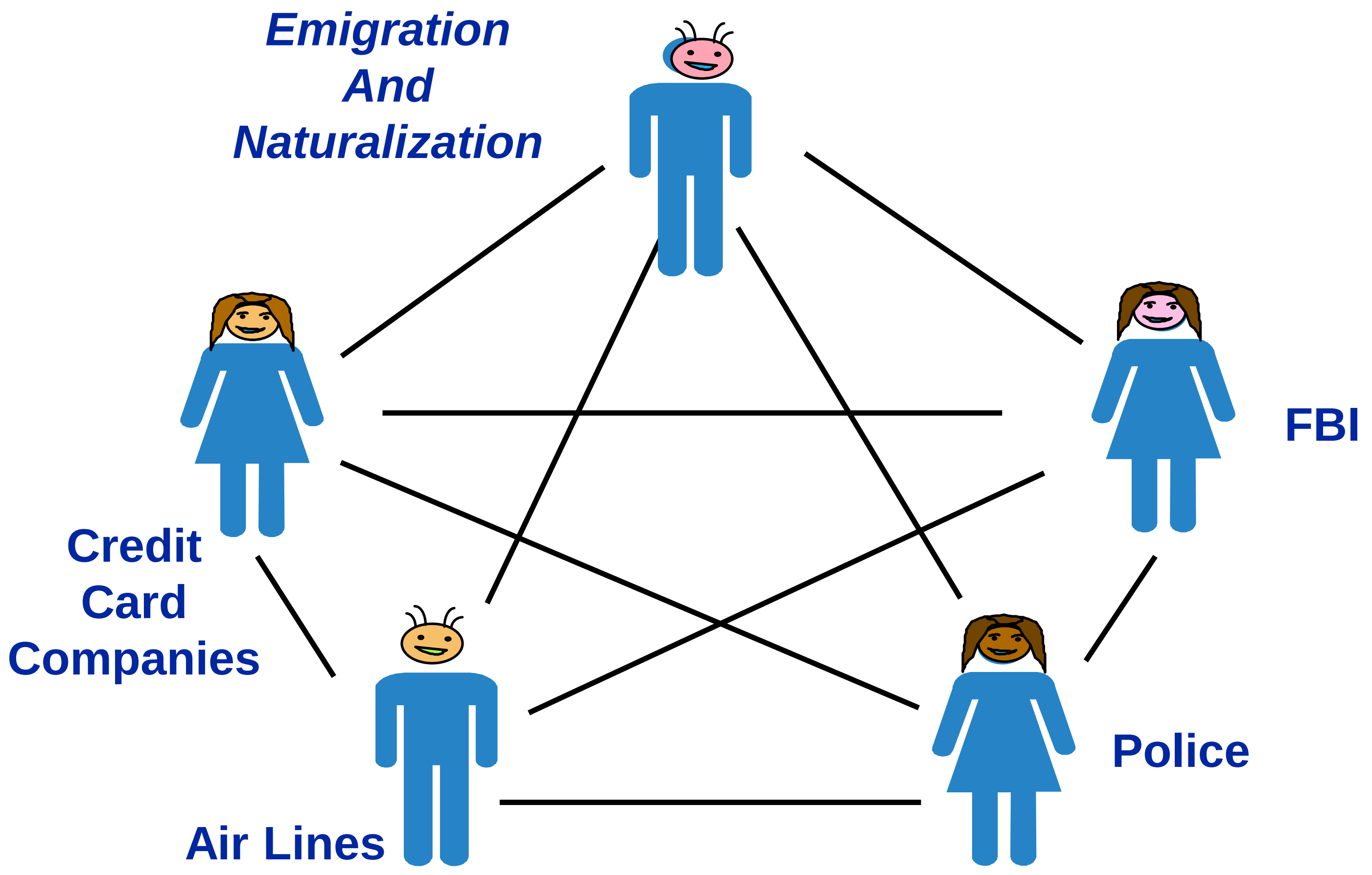
Risk Prediction for Global Economy



$g(X_1 \ldots X_n) =$

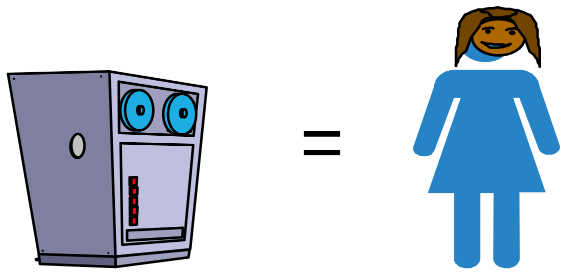
Will the Banks become insolvent

Data Mining



$g(X_1 .... X_n)=$

Intersection
without
revealing
individual Lists



Benchmarking (is used)

Companies want to know how well they are doing

- Compare performance to that of competitors

Actual performance is a trade secret

Can be translated to solving a linear program

- Other LP examples: Multi-attribute auctions, bilateral negotiations

Avoid Responsibility of dealing with private data

Denmark: Sugar Beet Auction (is used)

- 1200 Danish Farmers trade production rights for sugarbeets via an electronic auction.

- As a result, 25.000 tons worth of production rights change owner

- The market price at which trading occurs is computed by 3 servers, based on encrypted bids that are never decrypted.

First large-scale application of Secure Multiparty Computation.

Estonian Government Analysis Projects (is used)

2011: financial reporting for the Estonian Association of ICT companies

2014: linking income tax records and educational records to analyze if working during studies causes students to fail their studies

Future planned applications

2015: employee satisfaction survey in an Estonian City government

Unit 3: How to do MPC?

- Secret Sharing
- Sum Sharing
- Simple Example of Computing Salary with sum sharing

- Question: general computation
- Threshold Secret Sharing
- Polynomial Math
- Computing on Shares
- Pulling it all together: computing any function
- Completeness Theorems for n>=2t+1

Key Tool: Secret Sharing

A method to share secret information S among n parties so that

- Each of the n parties receives a share of the secret
- No player can recover the secret on his own
- Together all players can recover the secret

Sum Sharing

Let S be the secret to be shared

- Choose prime $p > S$
- Choose at random $x_1,...,x_{n-1}$ from $[1,...,p]$
- Set $x_n = S – (x_1 +....+x_{n-1}) \bmod p$
- Give player i share x_i

Claim: Any n-1 shares gives no information on S.

Application: Computing AVERAGE SALARY

Let X_a, X_b, X_c denote salaries and prime p>max salary

1. Each player chooses at random 3 numbers whose sum mod p is his salary

   $X_a = 8 = 5 + 2 + 1 \mod p$
   $X_b = 10 = 3 + 4 + 3 \mod p$
   $X_c = 12 = 6 + 5 + 1 \mod p$

2. Each player sends green to A, purple to B, blue to C
   The recipient sums the shares received.

   $Y_a = 5 + 3 + 6 = 14 \mod p$
   $Y_b = 2 + 4 + 5 = 11 \mod p$
   $Y_c = 1 + 3 + 1 = 5 \mod p$

3. Player i sends Y_i to all others:

   Average = $(Y_a + Y_b + Y_c \mod p)/3$

claim: Players cannot deduce more information about another player's salary than computable from the **Average** and his own salary.

claim: Can scale up to (n-1) colluding players out of n

Due to M. Rabin

Structure of the protocol

1. Secret Share your salary

2. Compute on the shares received: the AVERAGE is now secret-shared

3. Reconstruct the secret AVERAGE

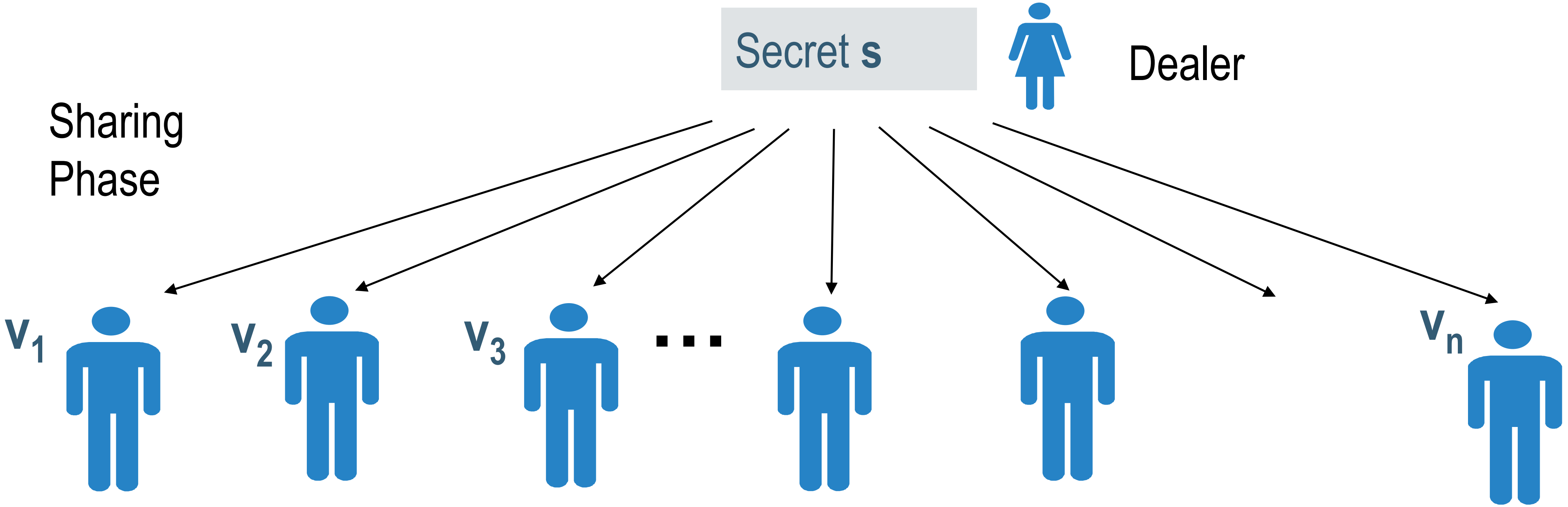
This can be done for general g, not only for AVERAGE

Key Tool: Threshold Secret Sharing

A method to share secret information S among n parties so that

- Each of the n parties receives a share of the secret
- Any t+1 parties can cooperate to recover the secret
- Any ≦t parties have no information about S
- t is a parameter

SECRET SHARING : SHARING PHASE

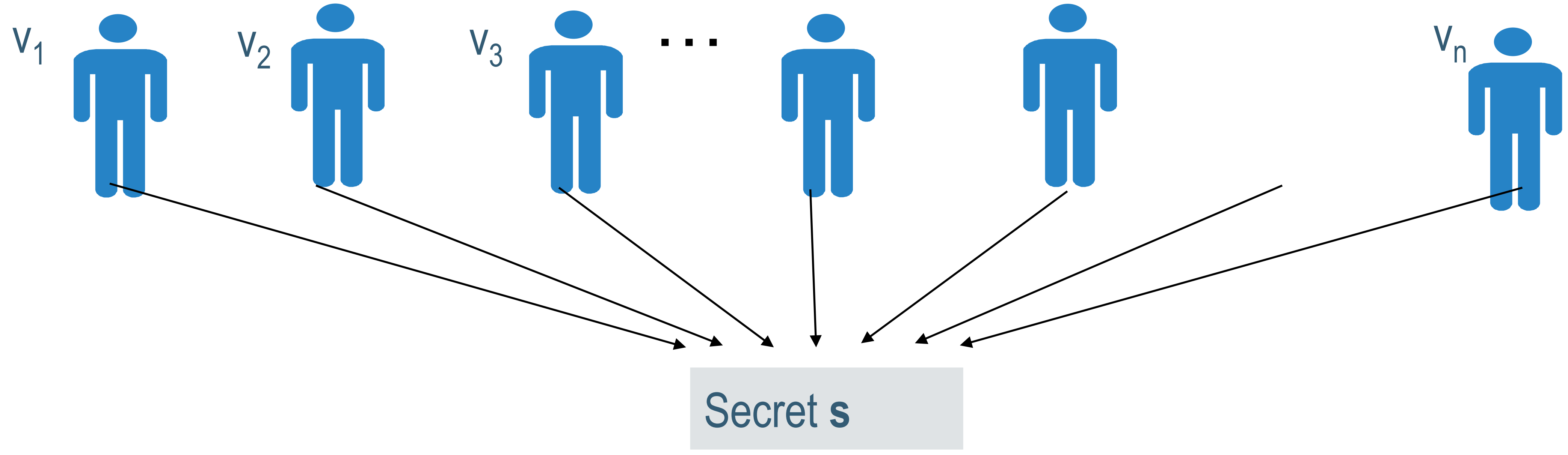


< t +1 players cannot reconstruct the secret

SECRET SHARING: RECONSTRUCTION PHASE

Reconstruction Phase

$\geq$ t +1 players can reconstruct the secret



Using Polynomials to Represent information

Let $c_i \in [1,\ldots,p]$ and $P(x) = c_0+c_1x +c_2x^2 +\ldots+c_tx^t \bmod p$ degree is t

Definition: r is a “root” of P(x) if P(r) = 0, E.g. $P(x) = x^2 - 2x + 1$, root = 1

Theorem: Every degree t polynomial has at most t roots.

Proof: By induction on the degree

Theorem: Any t+1 pairs (a_i,b_i) s.t. $b_i = P(a_i)$ and such that $a_i \neq a_j$ for all $i \neq j$ **uniquely determine** a polynomial of degree t

Proof: Let P and Q be two different t-degree polynomials. R(x)=P(x)-Q(x) is a t-degree polynomial. $P(a)=Q(a) \Rightarrow R(a) = 0$, but R has at most t roots. Contradiction.

How to Interpolate a Polynomial

Let $c_i \in [1,\ldots,p]$ and $P(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_t x^t \bmod p$

Theorem: The coefficients c_i's of P can be computed efficiently from t+1 pairs (a_i,b_i) such that . $a_i \neq a_j$ and for i=1,...,t+1

Lagrange Interpolation: Let $\lambda_i(x) = \prod_{j:i\neq j}(x-a_j)(a_i-a_j)^{-1} \bmod p$.

Note: $\lambda_i(x) = 1$ iff $x=a_i$. λ_i are called the Lagrange coefficients for t-degree polynomials

Set $P(x) = \sum_{i=1}^{t+1} b_i \lambda_i(x)$ [check that $P(a_i)=b_i$ for i=1…t+1

Claim: P is the unique t-degree polynomial such that $b_i=P(a_i)$ for all i=1,...,t+1

Secret Sharing Using Polynomials

Let S denote the secret from [1,..,p], let n>2t

$P(0) = S$

Sharing Phase

Dealer

- Picks t random coefficients $R_1, R_2, \ldots, R_t$ from [1,…,p]
- Let $P(x) = R_t x^t + R_{t-1} x^{t-1} + \ldots + R_1 x^1 + S \bmod p$
- For all j in [1,…,.n] the dealer gives player j's his share $v_j = P(j)$

Due to A. Shamir

Reconstruction Phase

Given t shares (i, v_i), interpolate the polynomial P and compute $P(0) = S$

t+1

Fact: Any t shares give no information on P(0)

Call P(0) : the secret of polynomial P

Computing with Shares of Polynomials: Almost

Let P(x), Q(x) be two polynomials of degree t where P(0) and Q(0) are secrets

S_1 S_2

and player i has shares P(i) and Q(i).

The sum of the shares is a share of the sum

The sum of the polynomials R(x)= P(x)+Q(x) is a degree-t polynomial

- R(0) =P(0) + Q(0)
- R(i) = P(i) + Q(i)
- Small caveat: R is not random

$S_1 \times S_2$

The product of the shares is "almost" a share of the product

The product polynomial R(x)=P(x)Q(x) is a degree-2t polynomial

- R(0)=P(0)Q(0)
- R(i) =P(i)Q(i)
- Small Caveat: R is not random

Due to
Benor Goldwasser Wigderson

MIT PROFESSIONAL EDUCATION

CSAIL

Computing with Shares of Polynomials: Almost

Let $P(x)$, $Q(x)$ be two polynomials of degree t where P(0) and Q(0) are secrets and player i has shares P(i) and Q(i).

The sum of the shares is a share of the sum: each player can compute it

The sum of the polynomials $R(x)= P(x)+Q(x)$ is a degree-t polynomial

- R(0) =P(0) + Q(0)
- R(i) = P(i) + Q(i)
- ~~Small caveat~~: R is not random. Can Fix

The product of the shares is "almost" a share of the product: how to fix this

The product polynomial R(x)=P(x)Q(x) is a degree-2t polynomial

- R(0)=P(0)Q(0)
- R(i) =P(i)Q(i)
- Small Caveat: R is not random. Can Fix

How to reduce the degree: need the help of other players

Product $R(x)=P(x)Q(x)$ is a degree 2t polynomial. Player j knows share $R(j)$

Goal: Protocol at the end of which, each player should know a share of a new polynomial R' of degree t such that $R(0)=R'(0)$ $= S_1 * S_2$

$R'(i)$

Let $R(0)= \Sigma_{i=1 \text{ to } n} \lambda_i(0)R(i)$ for $\lambda_i=\lambda_i(0)$ the Lagrange coefficients for 2t degree polynomials

Protocol: Each player i re-shares $R(i)$ using new random t-degree polynomials:

1. Each player i choose random polynomial R_i of degree t such that $R_i(0)=R(i)$ and send $R_i(j)$ to player j.

2. Define $R'(x)=\Sigma \lambda_i R_i(x)$. Player j computes new share of R' as $R'(j)= \Sigma \lambda_i R_i(j)$

How to reduce the degree: need the help of other players

Claim: (1) $\deg(R')=t$,

(2) $R'(0)=\Sigma\, \lambda_i R_i(0)=\Sigma\, \lambda_i R(i) = R(0)$

(3) Each player j can compute $R'(j)= \Sigma\, \lambda_i R_i(j)$

How to randomize polynomials

To randomize polynomial R of deg t maintaining its secret:

choose a random polynomial R' of degree t s.t. $R'(0)=0$

and set $R''(x)=R(x)+R'(x)$.

Claim: R'' is a random polynomial of degree t such that $R''(0)=R(0)$.

In terms of secret sharing:

1. Each player chooses a random degree t polynomial R'_i s.t. $R'_i(0)=0$ and sends to player j share $R_i'(j)$.
2. Player j sets his share of R'' to be $R''(j)=\text{old-share}+\Sigma_i R_i'(j)$

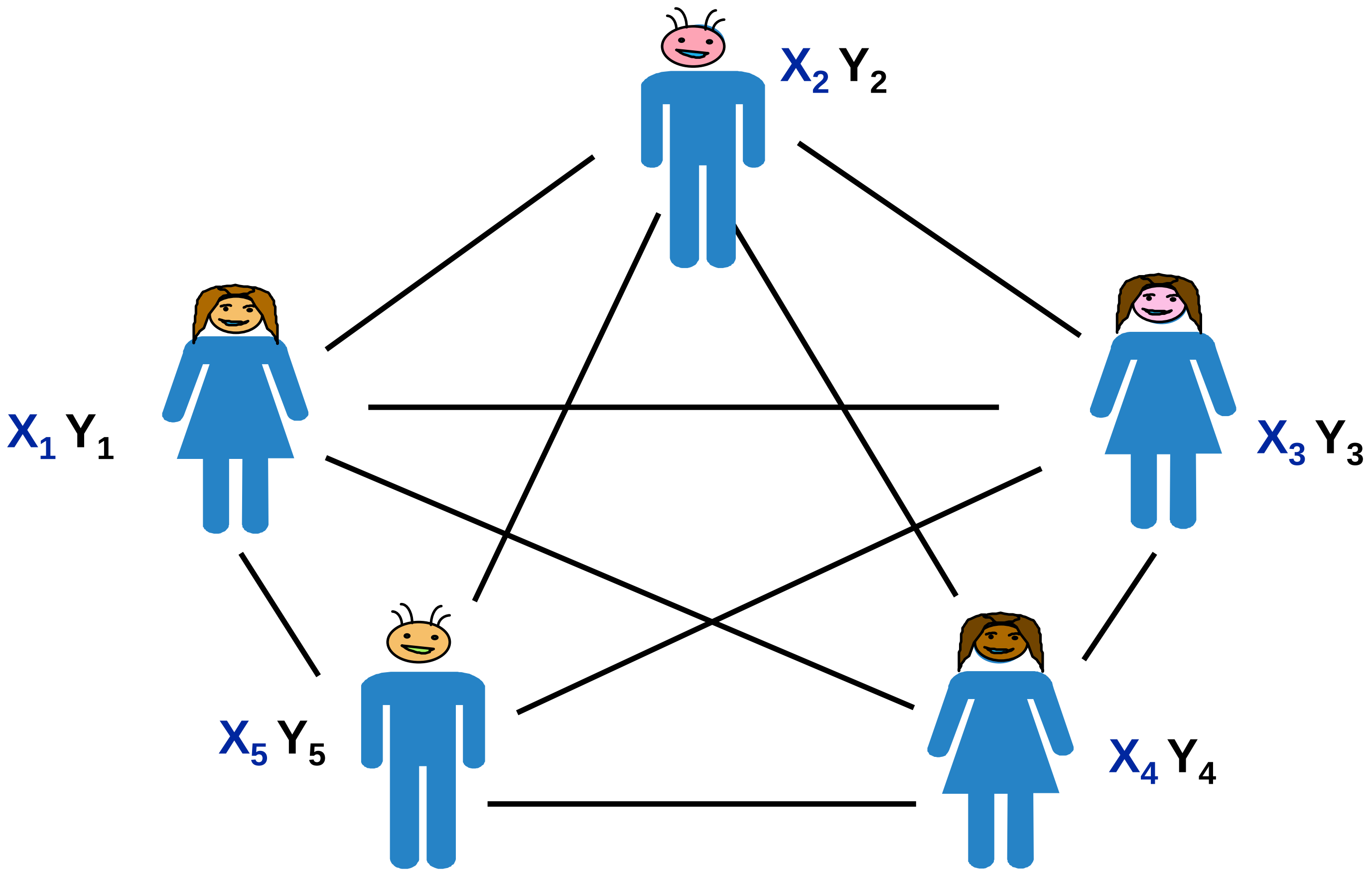
Overview: We can compute on shares of polynomials

Given shares of secrets s_1 and s_2, one can compute a share of s_1+s_2

Given shares of secrets s_1 and s_2 one can compute shares of $s_1 * s_2$

(with the help of other players via a protocol)

Back to Multi Party Secure Computation



N users

Each user/player
has private information X_i

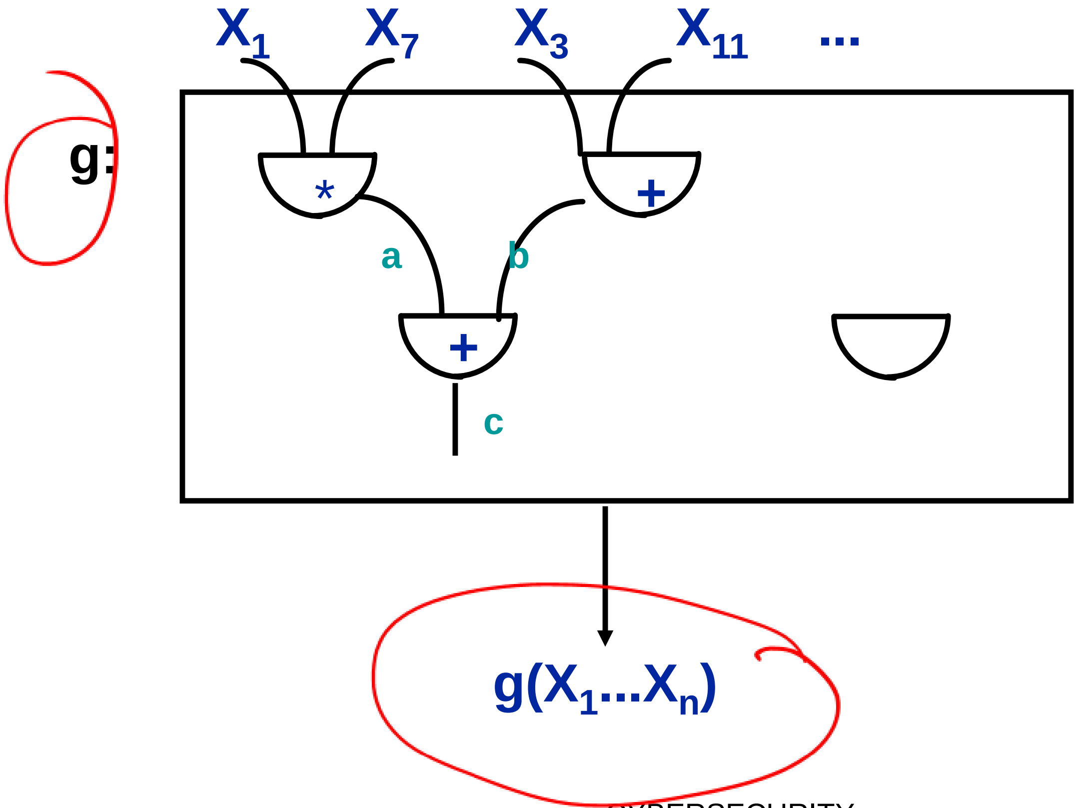
New Goal: correctly compute

$$(Y_1 \ldots Y_n) = g\,(X_1 \ldots X_n)$$

keeping X_i private from everyone else

MPC for general function g

Computing any function g can be done by performing a sequence of simple computations: sums and products.



MPC for general function g in 3 steps

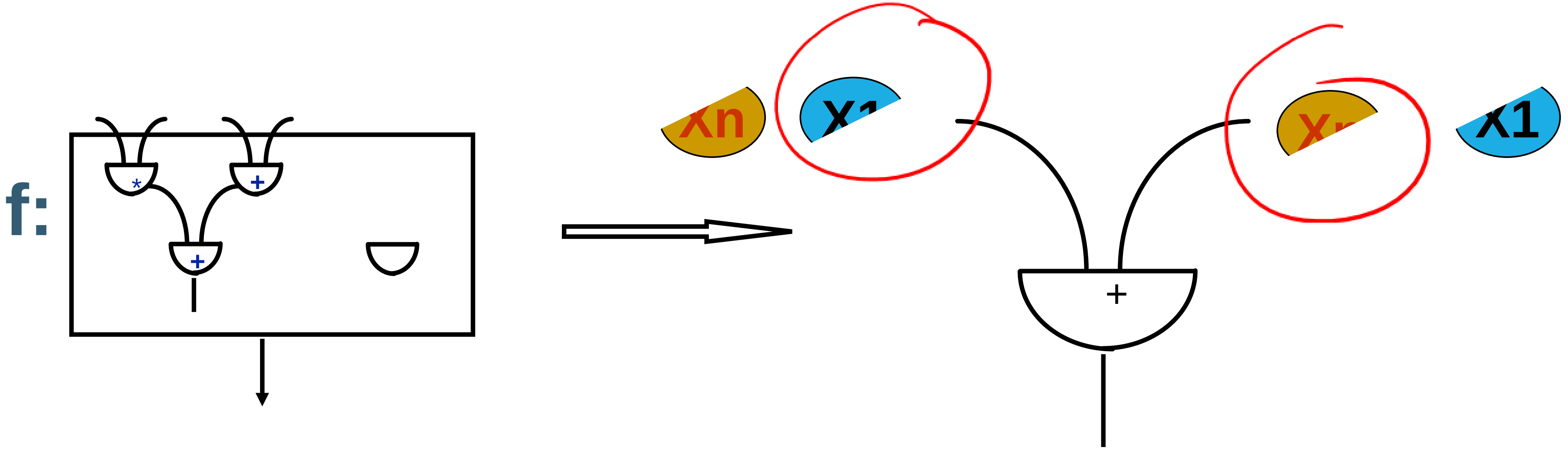
Input step:
each user secret shares his secret input x_i

Computation Step:
For each intermediate computation or gate, each player will have a share of the input values and compute a share of the output wire value.

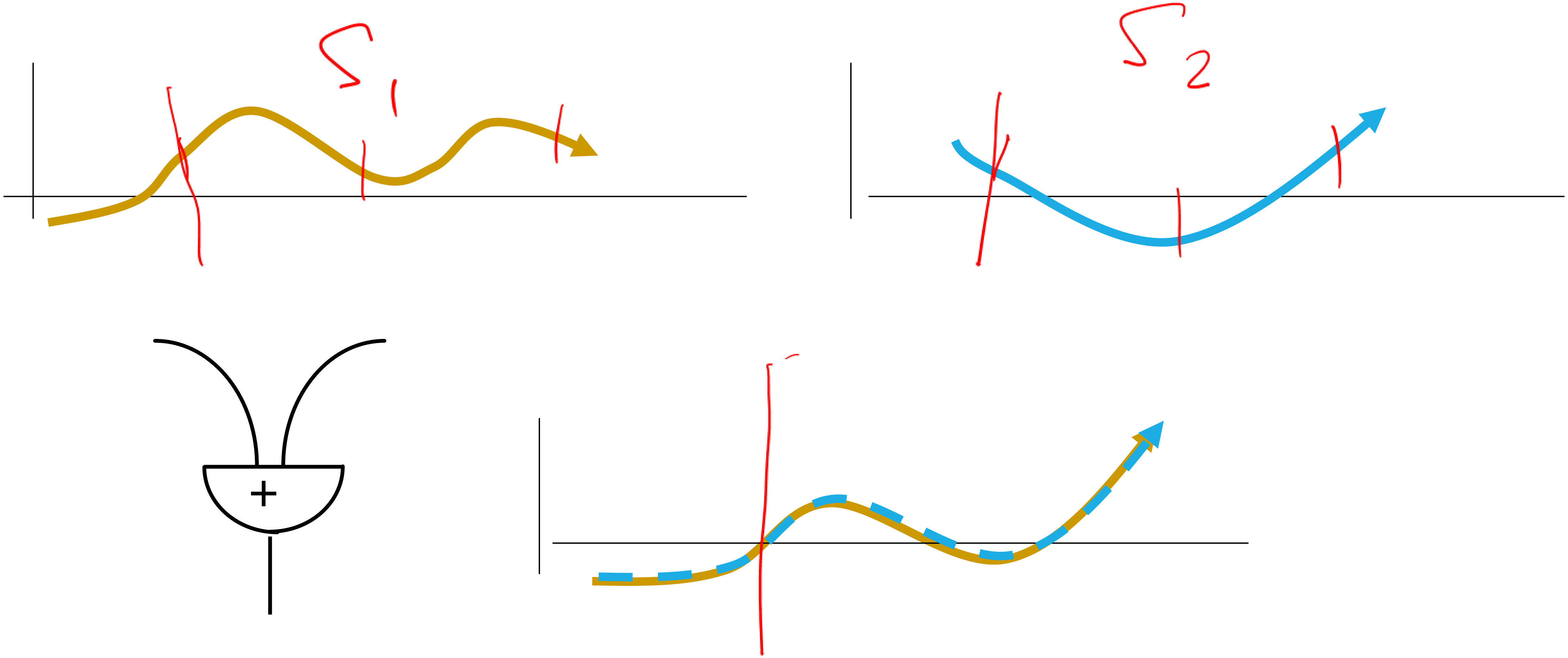
Output Step:
Everyone reveals shares of output of the entire computation g,
Everyone can reconstruct output from shares $g(x_1 \ldots x_n)$

MPC for general function g: Pictorially

Each player has a share of others secret inputs and can compute on shares alone



THE MAGIC OF ALGEBRA AND RANDOMNESS



COMPLETENESS THEOREMS

Theorem

Assuming honest majority, any multiparty function can be securely computed when the colluding faulty players are curious but non-deviating

Assuming honest 2/3 majority, any multiparty function can be securely computed when the colluding faulty players maliciously deviate from the protocol

Assuming honest majority, and *ability to broadcast messages* any multiparty function can be securely computed when the faulty players maliciously deviate

No Complexity Assumptions on the power of adversary

Unconditional security

VERY POWERFUL BUT EXPENSIVE BECAUSE OF INTERACTION PER GATE FOR ARBITRARY CIRCUITS

Good News: typical values are small. n= 3, p<64 digits

Many Efficiency Optimization

Less interaction, Less computation, batching of operations

Special g are often simple:

Linear Regression, Statistics, Comparisons

Has been implemented for particular tasks

Sugar beet auction in Denmark, Statistical Analysis in Estonia

ISSUES

How to reduce interaction by using more cryptographic operations?

Fully Homomorphic Encryption Lecture

Mapping algorithms to circuits: Algorithm → circuits → protocol

- Multiplications provide measure of complexity

Unit 4: Distributing Trust and Power

- Don’t store keys in 1 place:
- Don’t do Computation in 1 place
- Key escrow
- Cloud Application: remote storage and computation

DISTRIBUTE INFORMATION AND POWER

Distribute Information

- No single computer should have all the information
- Never store entire key in one place

Distribute Power

- No single computer should be able to perform cryptographic computations on its own such as sign, decrypt, etc.

SIMPLE AND ESSENTIAL FIX

Never store your secret keys in one place

Secret Share them and store shares in different computers

When its time to use them, reconstruct and then delete

Interesting Question:

Can you sign and decrypt never reconstructing secret keys?

THRESHOLD PUBLIC KEY CRYPTOGRAPHY

n parties share the ability to perform cryptographic operations

any t+1 out of n can perform operation jointly

Any t cannot perform operation

Example Cryptographic operations:

- Digital Signatures
- Decryption

APPLICATIONS OF THRESHOLD PKC APPLICATIONS

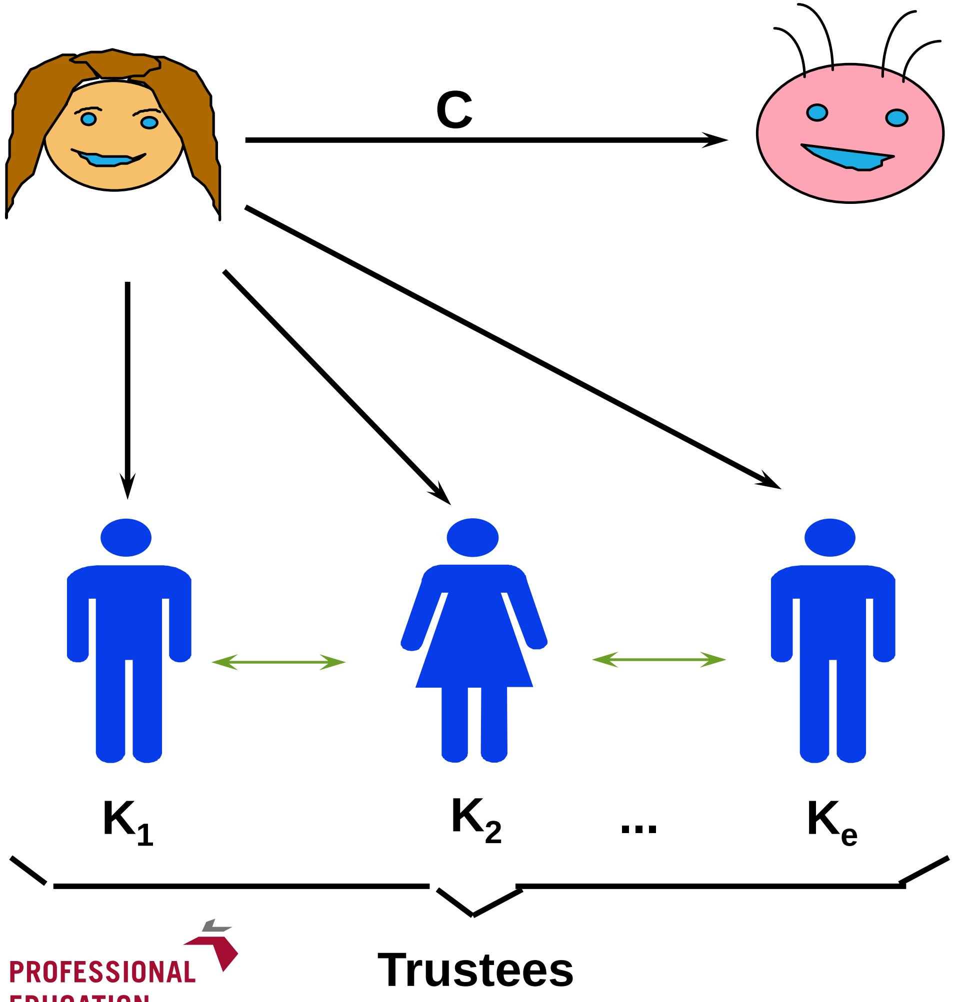
Remote Keys due to resource constraint of local machine (smart-card).

Public keys Associated with organizations.

Decryption Service “sits” on the Net for people with certificates

Key Escrow.

Key Escrow Reduces to General Multi-Party Computation



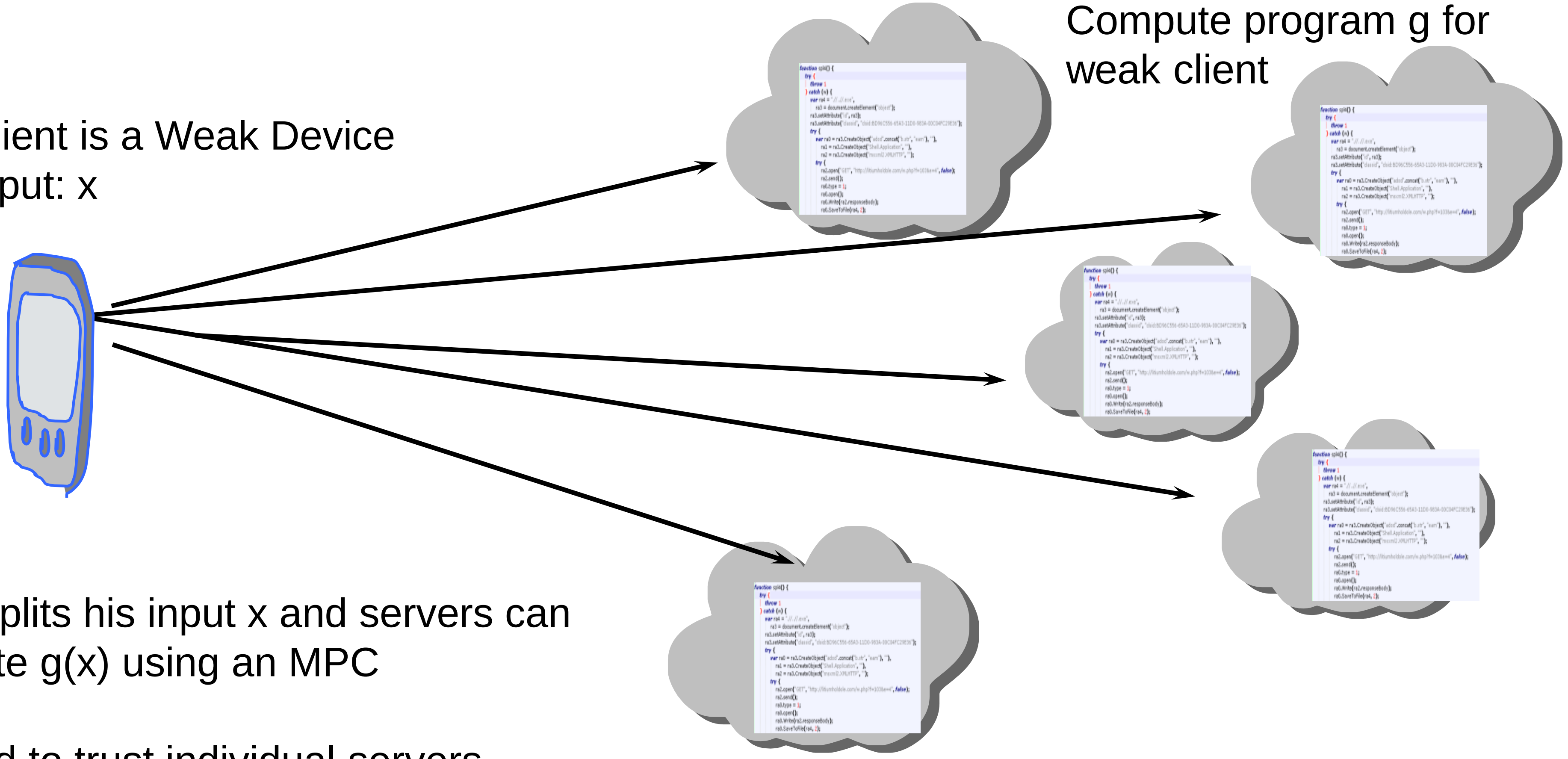
Key could be generated distributively by general multi-party computation.

In Practice:
Efficient Solutions
for RSA, Diffie-Hellman key exchange
using special purpose properties.

Outsource Computation to the cloud

Strong servers
Compute program g for
weak client

Client is a Weak Device
Input: x



Client splits his input x and servers can
Compute g(x) using an MPC

No need to trust individual servers

Unit5: Two Party Computation

- Two Party computation: certified mail, PIR, secret exchange
- Oblivious Transfer
- Circuit Garbling
- Zero Knowledge

TWO PARTY COMPUTATION AND COMPUTATION WITH FAULTY MAJORITY

The MPC we described work as long as the number of potentially faulty colluding untrusted players t are in minority

What if n=2t, or n<2t

Most interesting case is n=2 and neither party trusts the other

TWO PARTY COMPUTATION APPLICATIONS

Certified e-mail

Secret Exchange

Private Information Retrieval

TWO PARTY COMPUTATION

Different set of techniques

Zero Knowledge Proofs

Garbled Circuits

Oblivious Transfer

Require Computational Assumptions

on the power of

the adversary: computational security

THANK YOU

Shafi Goldwasser

RSA Professor of EECS