# Secure Multiparty Computation

Lucian-Dan Neștian

Supervisor: Lect. Dr. Sorin Iftene

"Alexandru Ioan Cuza" University, Iași

July 2019

# Outline

# Secure Multiparty Computation

## Context

- $n$ parties
- each party $i$ has a private input, $x_i$
- collaboratively compute a function, $f(x_1, ..., x_n)$

# SMC Proprieties

- Privacy
- Correctness

# Main SMC primitives

- Homomorphic Encryption
- Oblivious transfer
- Garbled Circuits

# Homomorphic Encryption

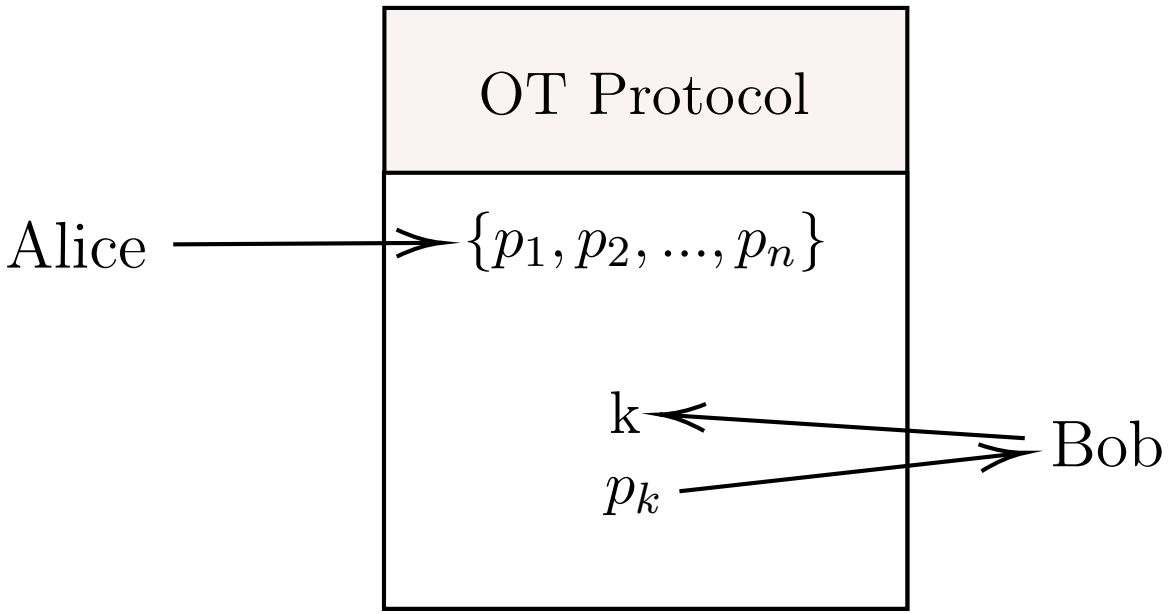
$$E(x_1) \bullet E(x_2) = E(x_1 \bullet x_2)$$

RSA :

$$E(x_1) \bullet E(x_2) = x_1^e x_2^e \bmod n = (x_1 x_2)^e \bmod n = E(x_1 x_2)$$

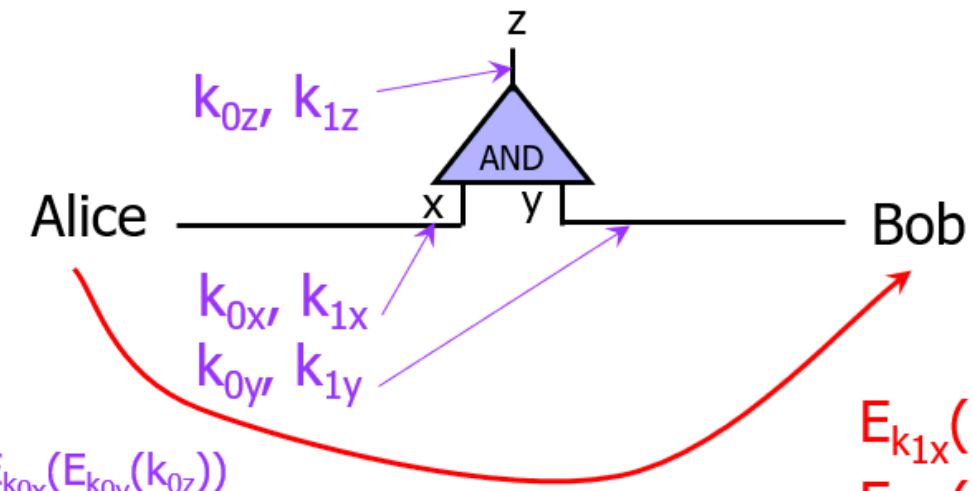
# Oblivious Transfer

FIGURE – 1-out-of-n Oblivious Transfer




- Alice should not know the value of $k$
- Bob should not know more than the value he requested, $p_k$

# Garbled Circuits Construction




AND Table

| x | y | z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

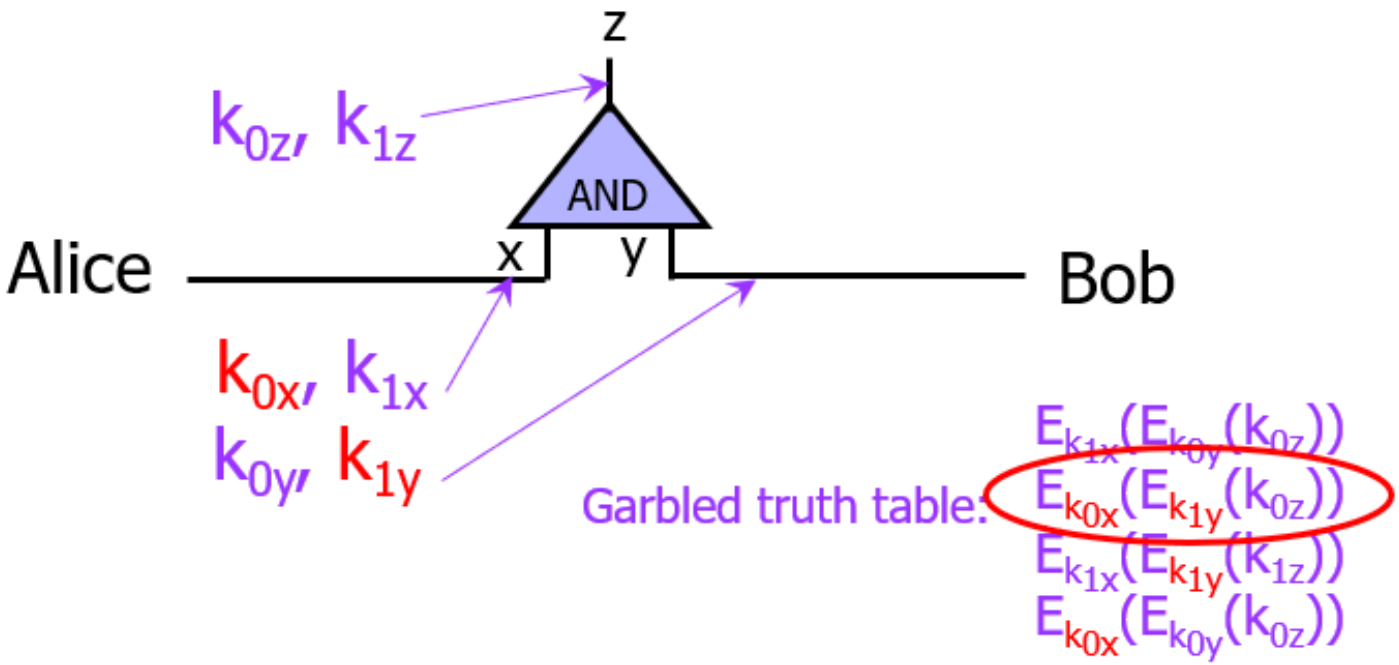
→

$E_{k0x}(E_{k0y}(k_{0z}))$
$E_{k0x}(E_{k1y}(k_{0z}))$
$E_{k1x}(E_{k0y}(k_{0z}))$
$E_{k1x}(E_{k1y}(k_{1z}))$

Garbled truth table:

$E_{k1x}(E_{k0y}(k_{0z}))$
$E_{k0x}(E_{k1y}(k_{0z}))$
$E_{k1x}(E_{k1y}(k_{1z}))$
$E_{k0x}(E_{k0y}(k_{0z}))$

# Garbled Circuits Construction




Vitaly Shmatikov, University of Texas, Austin, CS 380S
`www.cs.utexas.edu/~shmat/courses/cs380s_fall09/17yao.ppt`

# Privacy-Preserving KMeans Clustering

## classic approach

- Initialize the $k$ means $\mu_1 ... \mu_k$ to 0.
- Arbitrarily select $k$ starting points $\mu'_1 ... \mu'_k$
- **repeat**
  - Assign $\mu'_1 ... \mu'_k$ to $\mu_1 ... \mu_k$ respectively
  - **for all** *points* $i$
    - put point $i$ in the *closest cluster* (given a distance function)
  - **end for**
  - Calculate new means $\mu_1 ... \mu_k$
- **until** the difference between $\mu'_1 ... \mu'_k$ and $\mu_1 ... \mu_k <$ an arbitrary threshold

# Privacy-Preserving KMeans Clustering

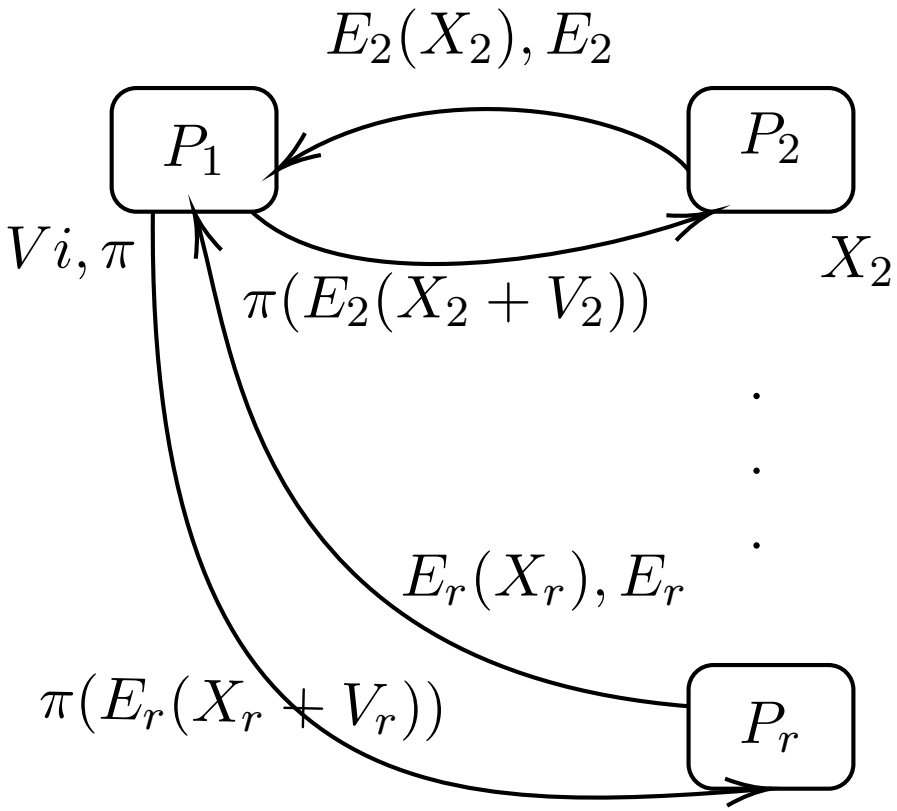
## secure approach

$$P_1 \text{ has } X_1 = \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{k1} \end{bmatrix}, P_2 \text{ has } X_2 = \begin{bmatrix} x_{12} \\ x_{22} \\ \vdots \\ x_{k2} \end{bmatrix} \ldots P_r \text{ has } X_r = \begin{bmatrix} x_{1r} \\ x_{2r} \\ \vdots \\ x_{kr} \end{bmatrix}$$

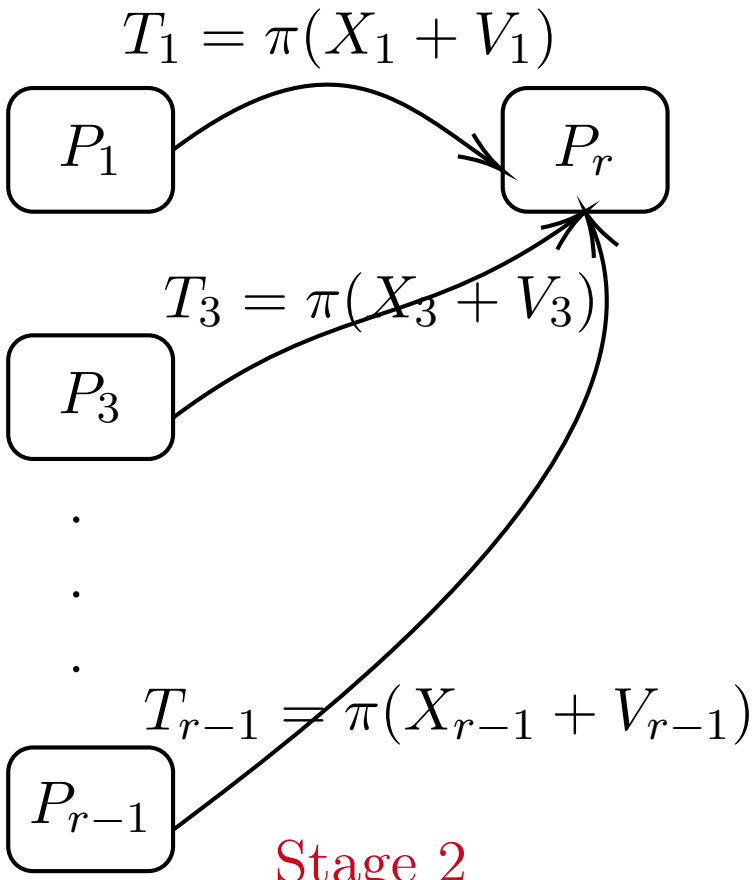
$$\underset{i=1..k}{\operatorname{argmin}}\Big(\sum_{j=1..r} x_{ij}\Big)$$

- Disguise the components of the distance with random values that cancel out when combined.
- Compare distances so only the comparison result is learned.
- Permute the order of clusters.

# closest cluster



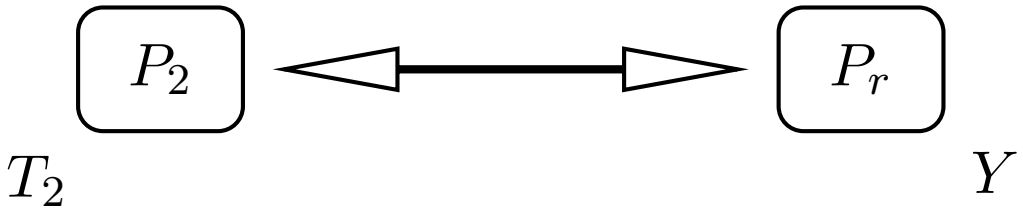

Stage 1




Stage 2

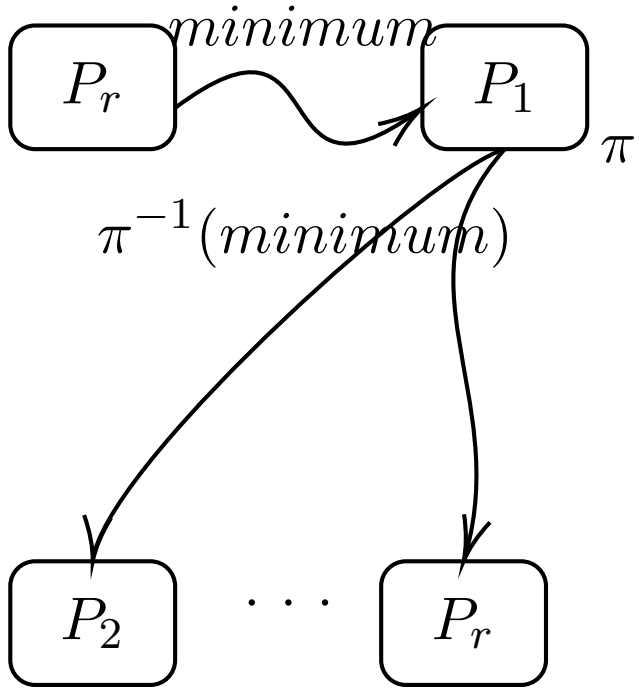
with $V_{k \times r}$ such that $\sum_{i=1}^{r} \vec{V_i} = \vec{0}$

# closest cluster
## retrieving the minimum




*secureAddAndCompare*
$(\mathbf{Y_j} + T_{2j}\ , \mathbf{Y_{minimum}} + T_{2minimum})$

*update minimum accordingly*

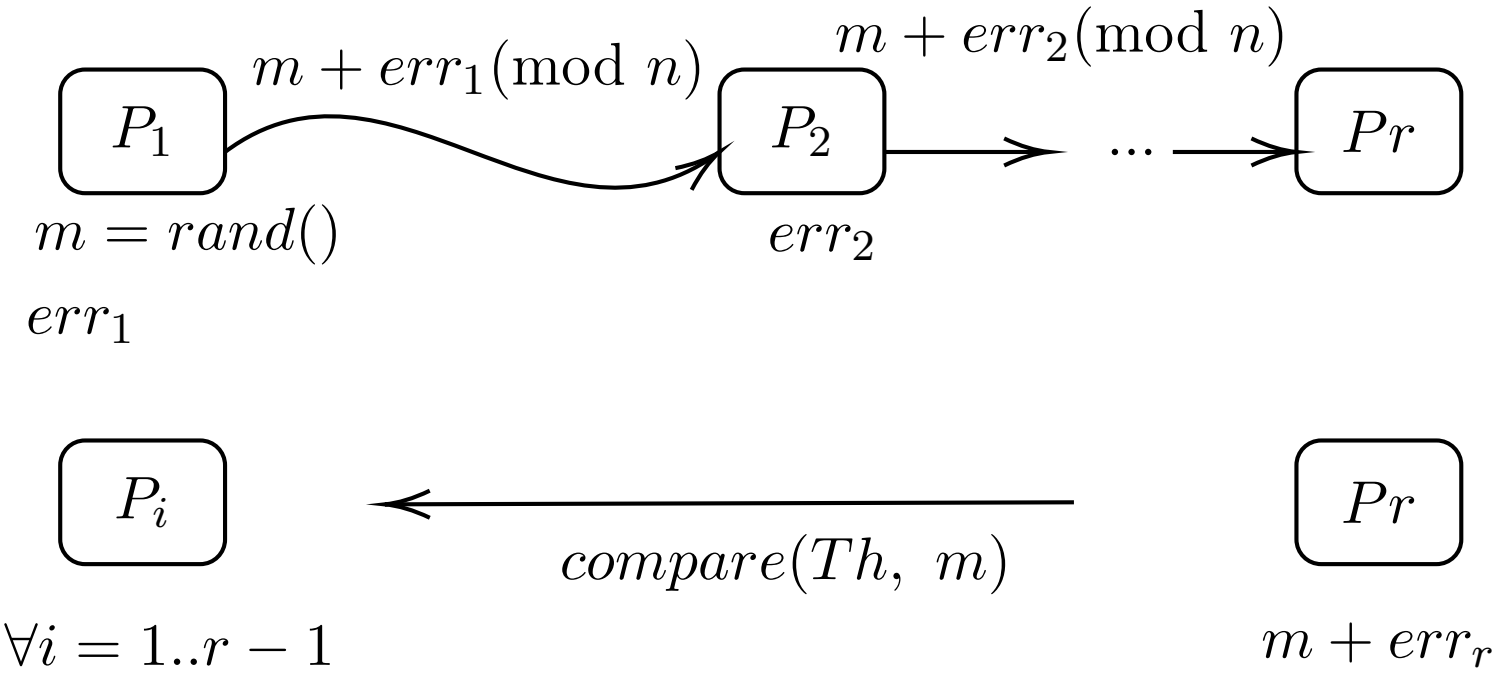



Stage 3

Stage 4

$$Y = T_1 + \sum_{i=3}^{r} T_i$$

# check termination threshold




$$err_i = 0$$
$$for\ each\ j\ in\ \{1...k\}:$$
$$err_i = err_i + dist(\mu_j,\ \mu'_j)$$

Jaideep Vaidya, Privacy − preserving kmeans clustering over vertically partitioned data

# Future work

Directions for further study :

- Reduce computational and communication cost
- Other clustering solutions with the privacy-preserving property

# Conclusions

Secure multiparty computation :

- aims to solve the distributed privacy problem
- is generic
- has great practical use