# Trial lecture

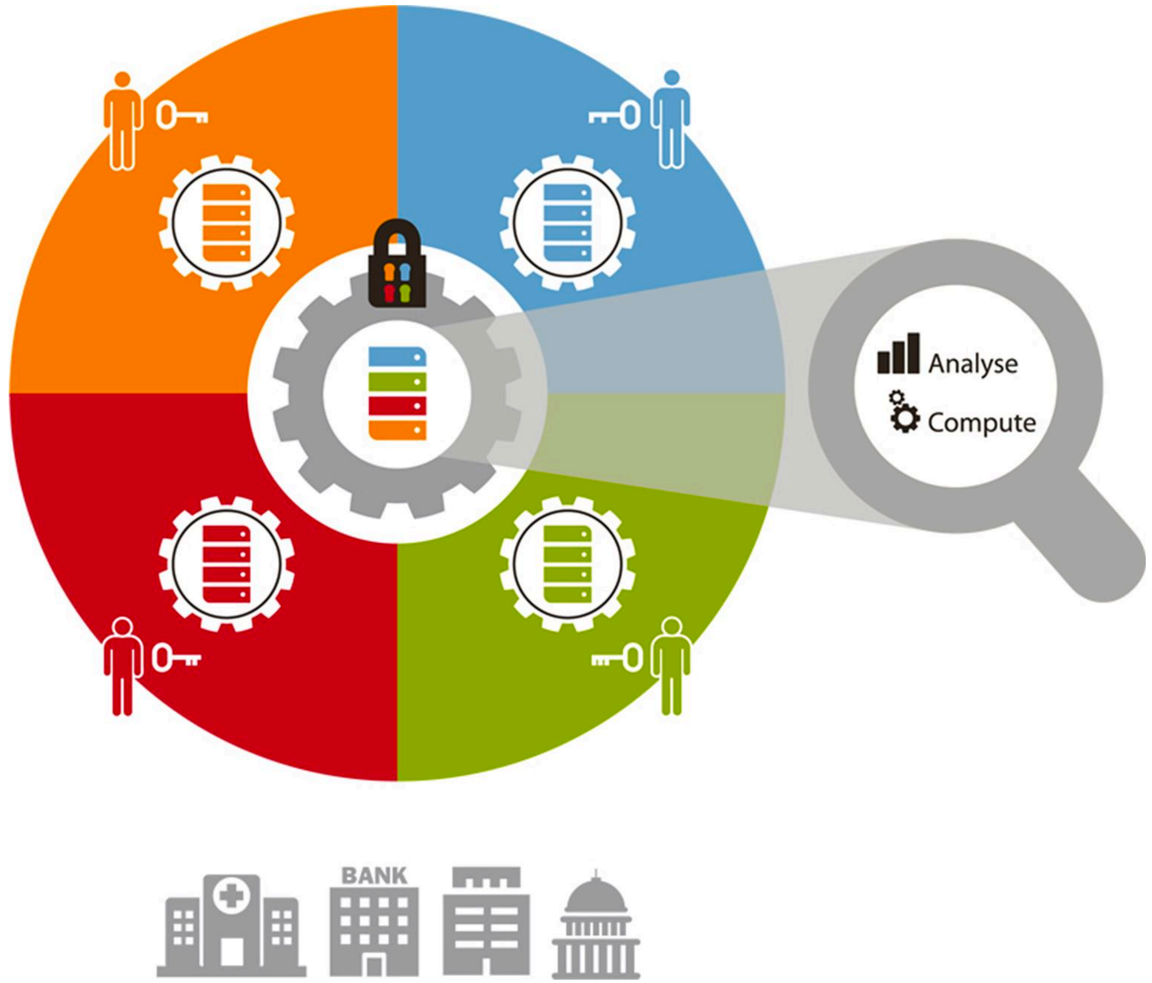
# Secure Multi-Party Computations. From two to many millionaires - in the presence of faithful and unfaithful computing power

Suresh Kumar Mukhiya
skmu@hvl.no

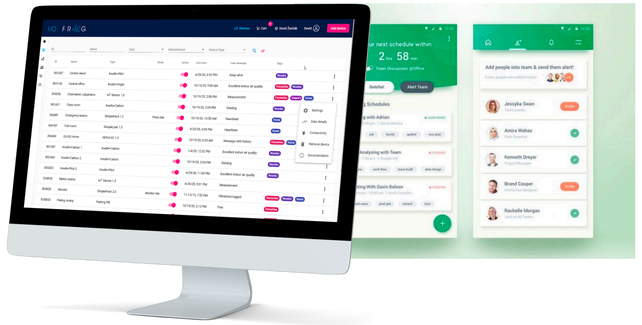
Høgskulen på Vestlandet

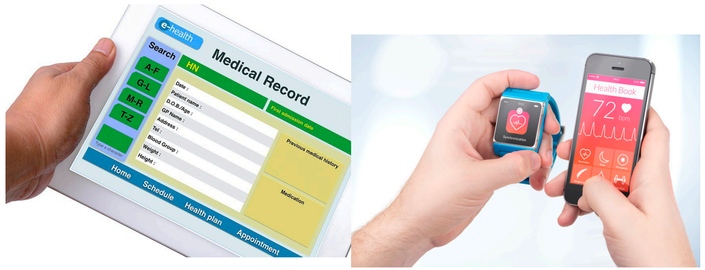
# Outline

- Motivation for Multiparty Computation (MPC)
- What is MPC Problem
- Protocols
  - Two party computation
  - Multiparty computation
- Solutions to the problem
- Approaches/stages of Secure MPC
- Real/Ideal world security definitions
- Practical Applications
- Benefits/Limitations



# Digital Revolution








- Internet of Things
- Mobile Computing
- Big data
- Cloud computing

# How do we safely store data?


**Data becomes virtually indestructible !**

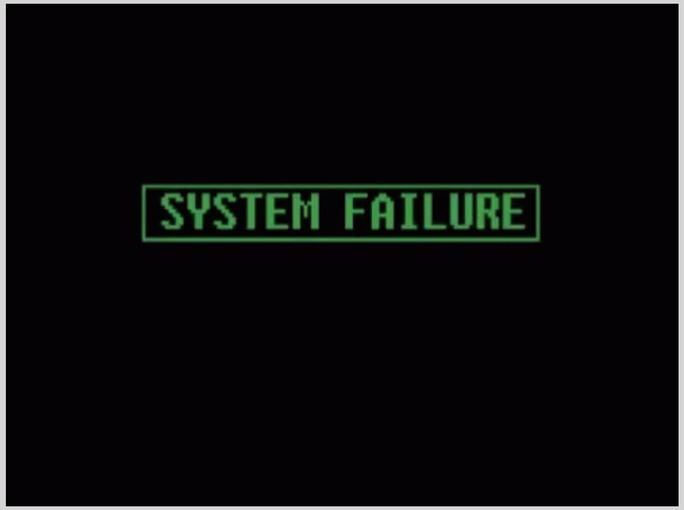
Error-Correcting Codes: [Hamming 1947; 1], [Shannon 1948; 2], How to make your data virtually indestructible, without using much storage space?



*[Yuval Ishai, Talk]*

# But what about these?






# Concept of Backup

Alibaba Cloud

box

Dropbox

iCloud

Single Point of Failure

Google Drive

OneDrive

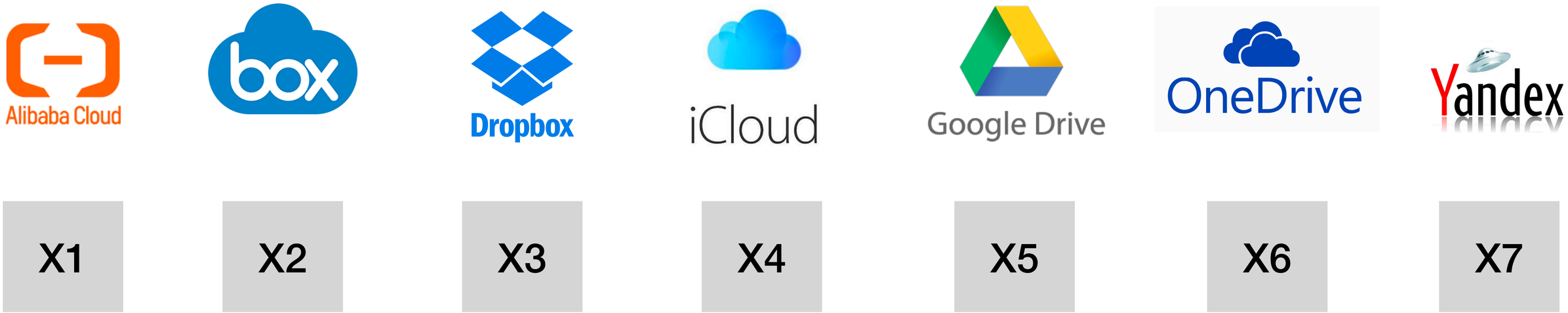
Yandex

Single Point of Failure

Data is virtually indestructible, and unleakable.

But what about confidentiality?

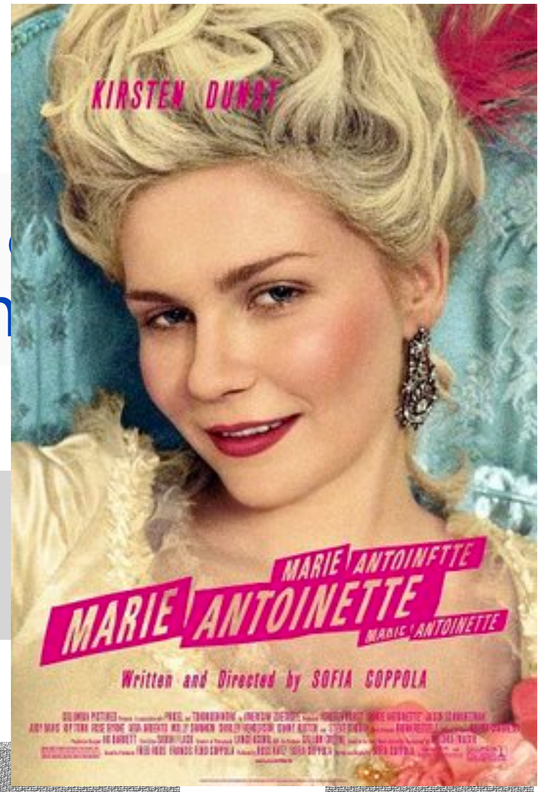
# Solution: Use Secret Sharing




- Introduced by Ali Shamir, and George Bleakley independently in 1979.
- Idea is to have a dealer distribute a **secret** S (encode/encrypted data), among more than one party.
- Why split into multiple party?  Adversary have to hack multiple party to obtain data.

# Secret Sharing

Search all pictures on VOGUE magazines??




Alibaba Cloud

box

Dropbox

iCloud

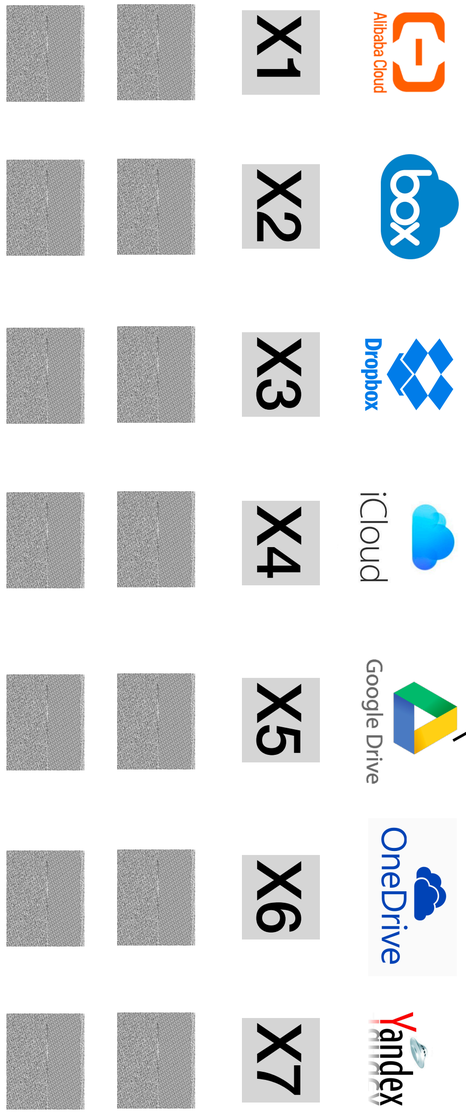
Google Drive

On

X1 X2 X3 X4 X5

Each cloud server can see part of gibberish data, making it private.

# Secret Sharing - Centralised




Single Point of Failure


Search Interface


# Decentralised server - MPC








| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **INPUT** | X1 | X2 | X3 | X4 | X5 | X6 | X7 |
| **OUTPUT** | Y1 | Y2 | Y3 | Y4 | Y5 | Y6 | Y7 |

- MPC - Multi-Party computation protocols provides decentralised alternative.
- We can keep in encrypted form in different party/clouds but still perform functionality.
- By allowing these server to interact

# Multiparty Computation (MPC)

High Level Definition:

Process **sensitive, distributed, encrypted** data *without introducing a single point of failure* and *without leaking confidentiality.*

# MPC - Goals

What we are trying to protect

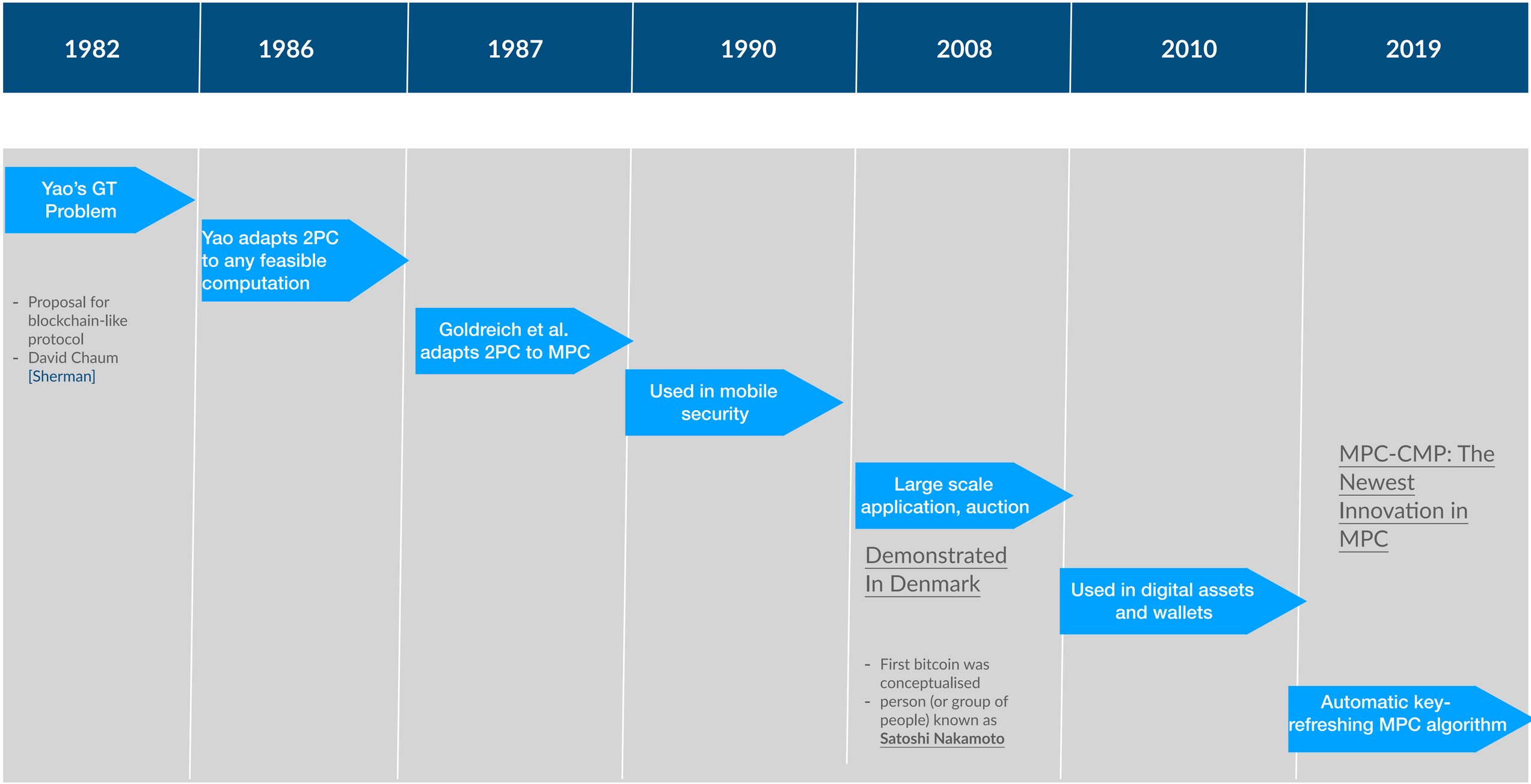
Security Goals

| | Information | Computation |
|---|---|---|
| **Integrity**<br>Correctness of data | **Error Correcting code** | **Fault-tolerant computation** |
| **Confidentiality**<br>Secrecy of data | **Secret -sharing scheme** | **Secure Multiparty Computation** |

# MPC - History 1/2

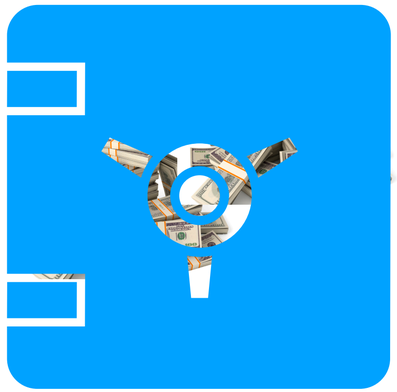
- Also referred as *secure computation*, *multiparty computation (MPC)*, *privacy preserving computation.*
- Is a subfield of cryptography.
- Formally introduced as **Secure Two-party computation** (2PC) in 1982 by Andrew Yao. So, called “The Millionaire’s Problem”.

Secure Multi-Party Computations. **From two to many millionaires** - in the presence of faithful and unfaithful computing power

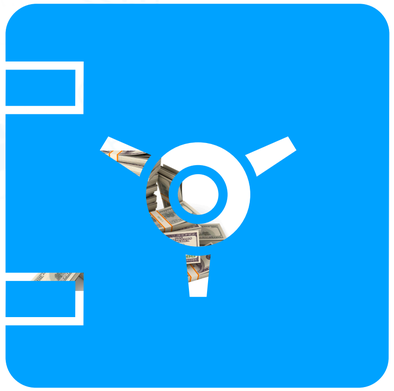

1982
1986
1987
1990
2008
2010
2019
Yao's GT Problem
- Proposal for blockchain-like protocol
- David Chaum [Sherman]
Yao adapts 2PC to any feasible computation
Goldreich et al. adapts 2PC to MPC
Used in mobile security
Large scale application, auction
Demonstrated In Denmark
- First bitcoin was conceptualised
- person (or group of people) known as Satoshi Nakamoto
Used in digital assets and wallets
MPC-CMP: The Newest Innovation in MPC
Automatic key-refreshing MPC algorithm

# The Millionaire's Problem (GT problem)

ALICE



BOB



**Goal:** To know who is richer, without revealing their actual wealth.

**Generalisation:** Given two numbers, ***a*** and ***b***, determine if $a \geq b$ is TRUE or FALSE.



*[Yao, 1982; 4]*

# The Millionaire's Problem- Solution

ALICE


Semi-trusted third party as the central server

BOB

**Generalisation:** Given two numbers, ***a*** and ***b***, determine if $a \geq b$ is TRUE or FALSE.

Example: **`a = 6, b = 2.`**

**Step 1:** Assume **`A, B`** are binary string of length ***`n`***

**`A = 110, B = 010`**

**Step 2:** Encode both string using 0-encoding, and 1 encoding as shown below:

$$\text{Let } s = s_n s_{n-1} \ldots s_1 \in \{0,1\}^n \text{ be a binary string of length } n.$$

*0-encoding* $S_s^0 = \{s_n s_{n-1} \ldots s_{i+1} 1 | s_i = 0, 1 \leq i \leq n\}$

*1-encoding* $S_s^1 = \{s_n s_{n-1} \ldots s_i | s_i = 1, 1 \leq i \leq n\}$

**`A = 110 = S¹A = {1,11}`** **`B = 010 = S⁰B = {1,011}`**

**Step 3:** **`a > b`** is valid, if they have a common element.

**`S¹A ∩ S⁰B ≠ ∅`** ⟶ **`a > b is valid`**



*Lin H.Y., et al.*

# Applications - Problems with GT solution

**Applications**

- Secret voting
- Negotiation
- Private Querying of Database
- E-commerce application/ competitors price comparison
- Secure auctions

**Problems**

- The original solution by Yao came with exponential in time and space requirements.
- Several other solutions were proposed, with improved computational performance.

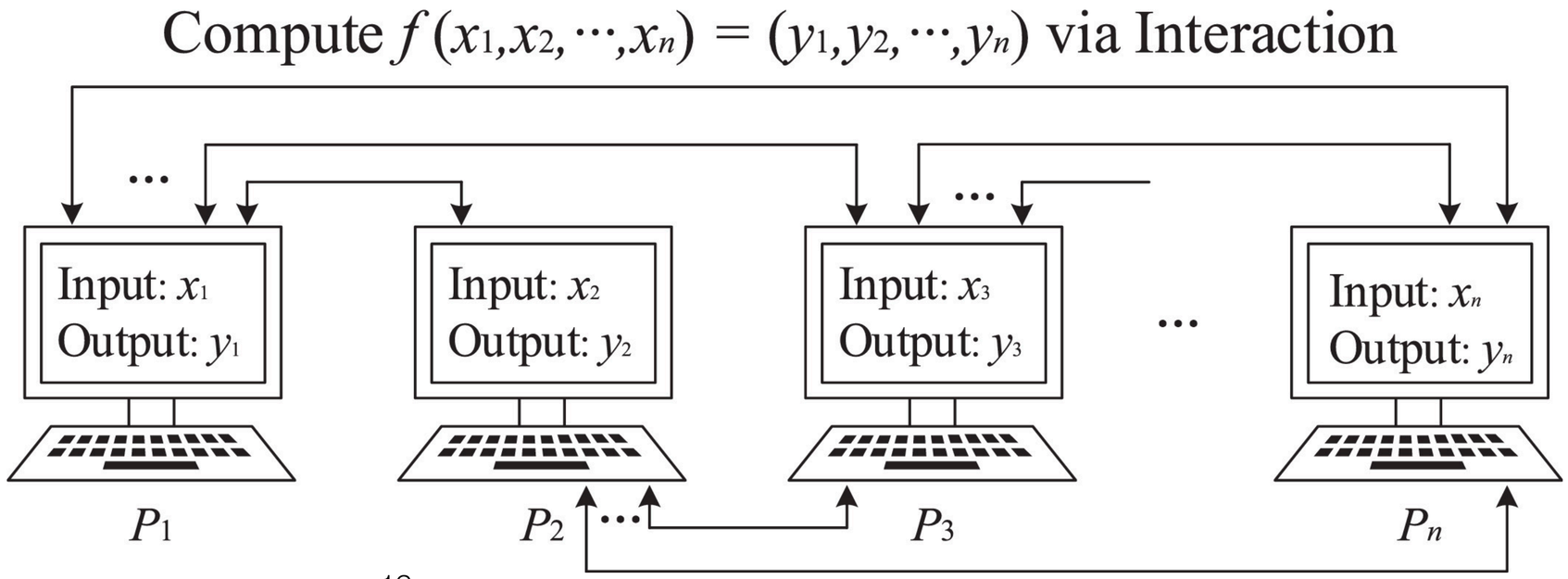
*[Yao, 1982; 4]*

*[Lin H.Y., et al.; 6 ]*

# MPC

DEFINITION

In MPC, a given number of parties $p_1, p_2, p_3, \ldots, p_N$, each have private data, respectively, $d_1, d_2, d_3, \ldots, d_N$. Parties want to compute a public function on that private data $F(d_1, d_2, d_3, \ldots, d_N)$ while keeping their inputs secret.

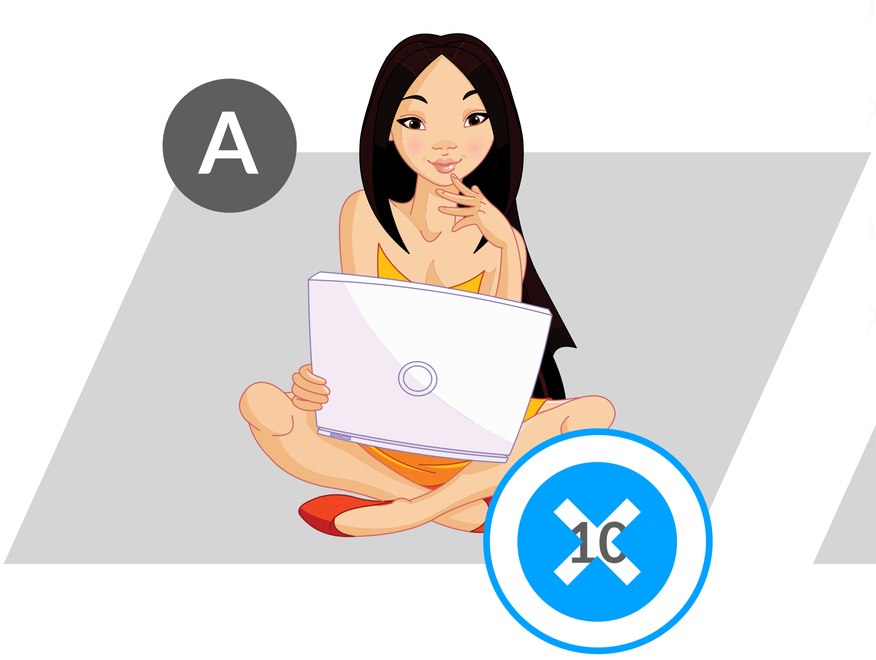


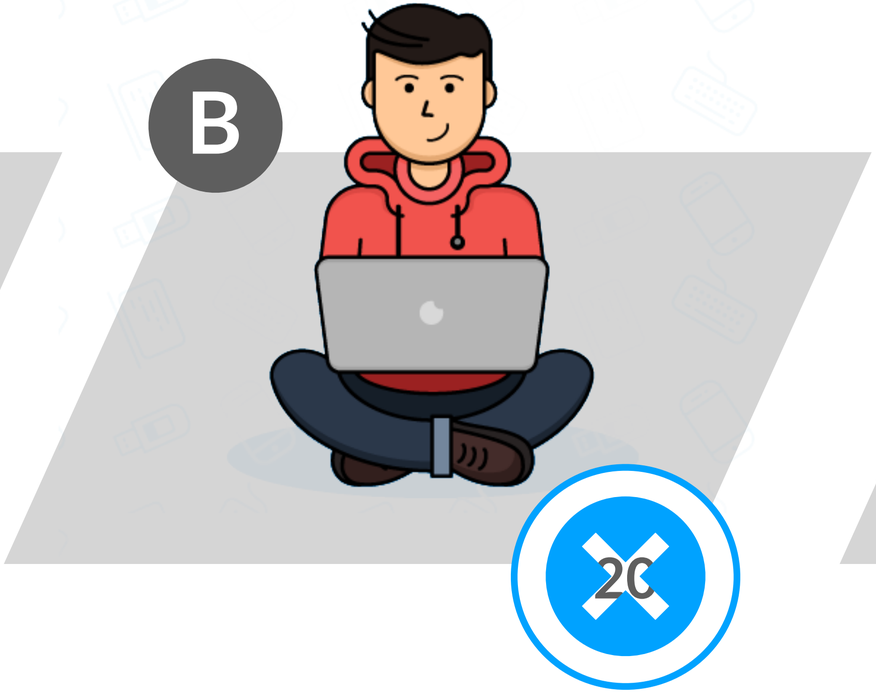

**Secret sharing VS MPC:**

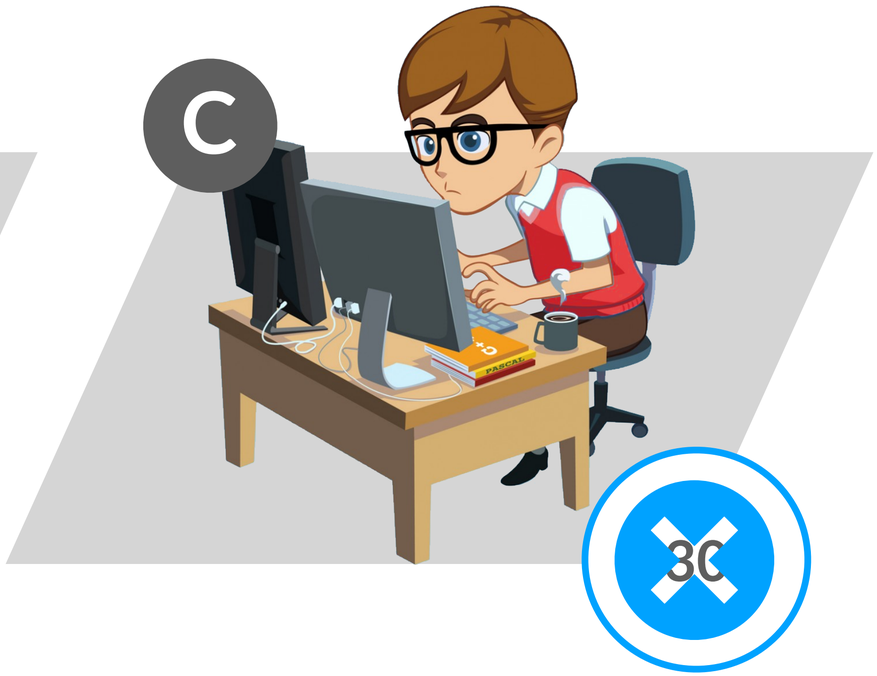
- A dealer knows the secrets of all parties.
- MPC: The secret of each player is kept secret at all times. Only the output of the function is made public.

# MPC - An Example

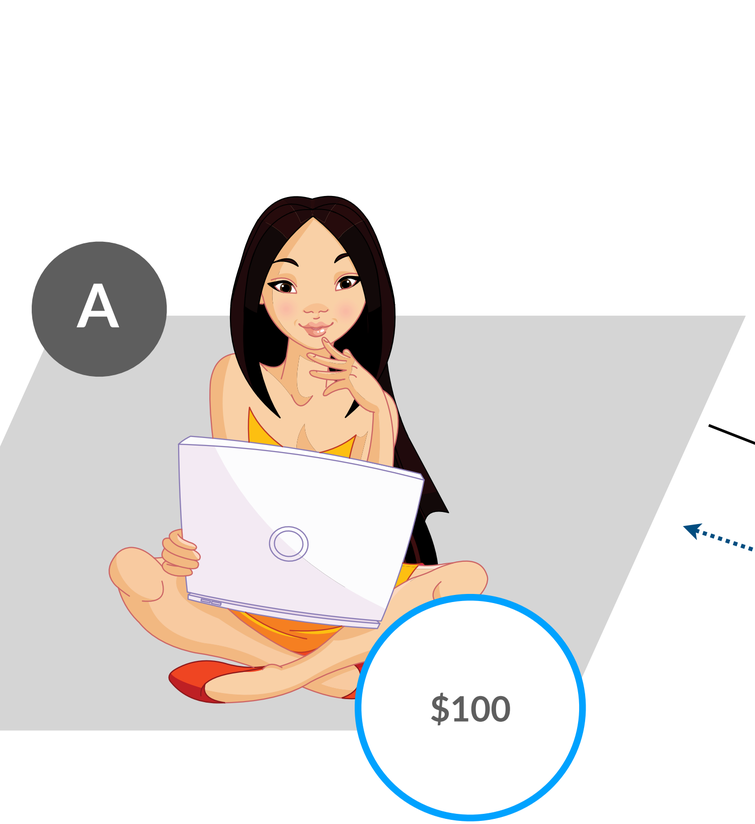
$$\text{Average Salary} = \frac{A + B + C}{3}$$



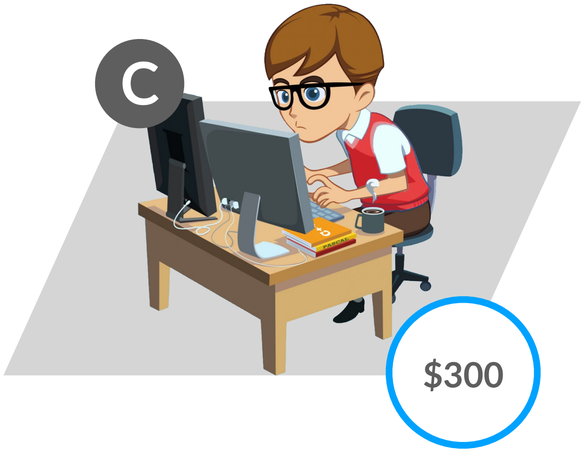



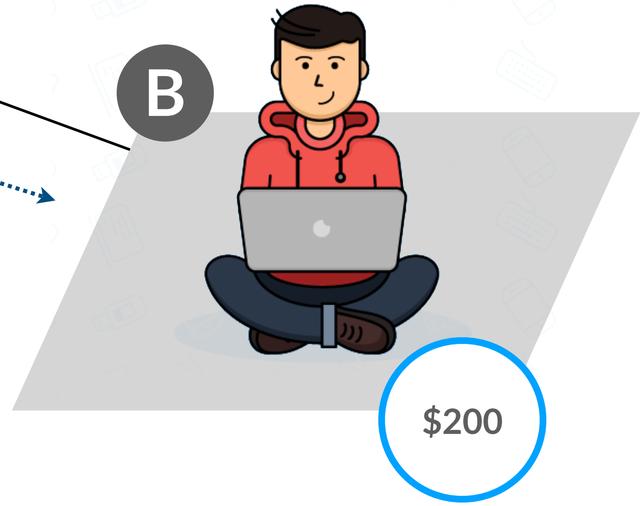



# MPC - Simplest solution








$$\frac{A+B+C}{3}$$

$\sum$ = $600 / 3

= $200

**Problem:** TTP problem (Single Point of failure), Not easy to find trusted partner, those we trust are not always trustworthy

# Use MPC protocol to compute average salary.

A = $100

$f(x)$

50

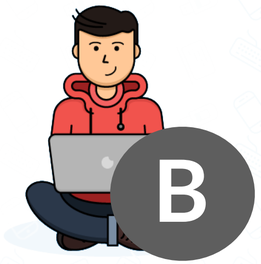
+

30

+

20

= $100




B = $200

$f(x)$

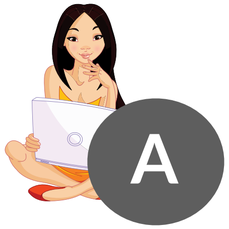
-80

+

100

+

180



C = $300

$f(x)$

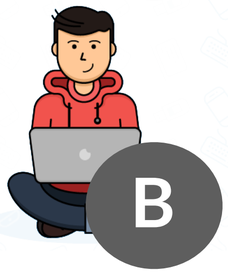
0

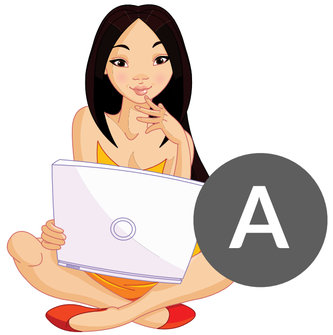
+

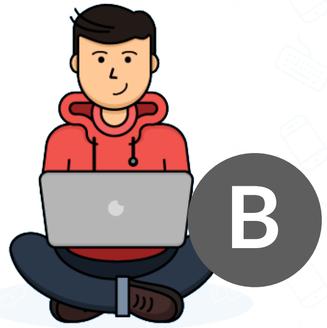
350

+

-50

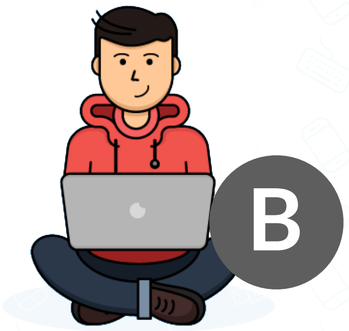






| | | | | |
|---|---|---|---|---|
| A | = $100 | 50 | 30 | 20 |
| B | = $200 | -80 | 100 | 180 |
| C | = $300 | 0 | 350 | -50 |
| Σ | | -30 | 480 | 150 |



-30

480

150

$200

$200

$200

$\sum$ = $600 / 3

= $200

Combined sum divided by total number of participants

# MPC - Threats and security Requirements

- *Privacy:* No party can obtain any information except own input and output.
- *Correctness:* Output should be correct.
- *Independence:* The input selected by participant should be honest.
- *Guarantee of outputs:* Adversary should not be able to interrupt computation.
- *Fairness:* Corrupted parties receives own output iff the honest parties receive their own output.

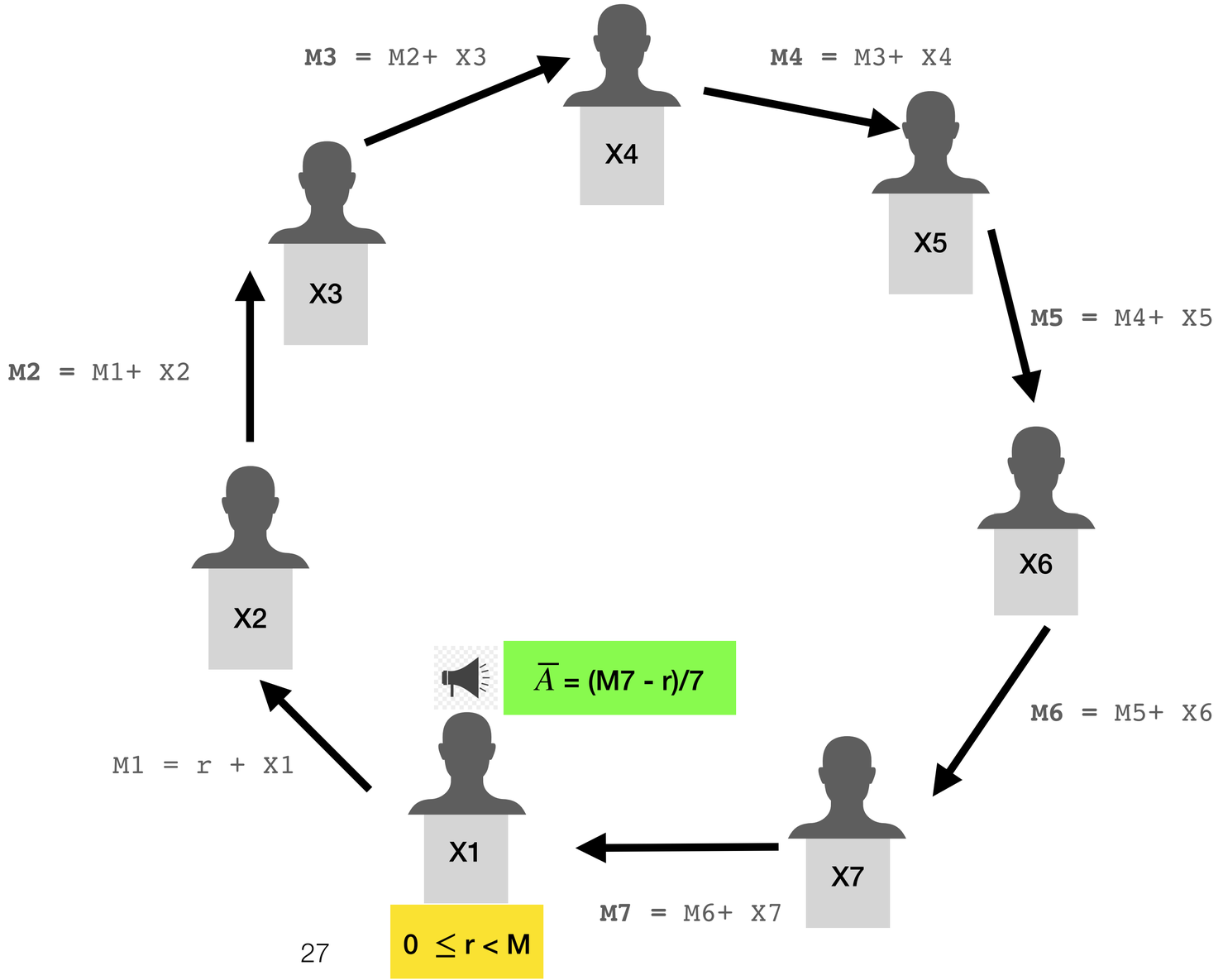
*[Zhao, Chuan, et al.; 3]*

# MPC Solution: Secure aggregation

**Problem:** Compute average without revealing inputs

**Assumptions:** Maximum Salary = M = $10^{10}$

Assuming each party sends just the mask data, and trustworthy, the algorithm works.



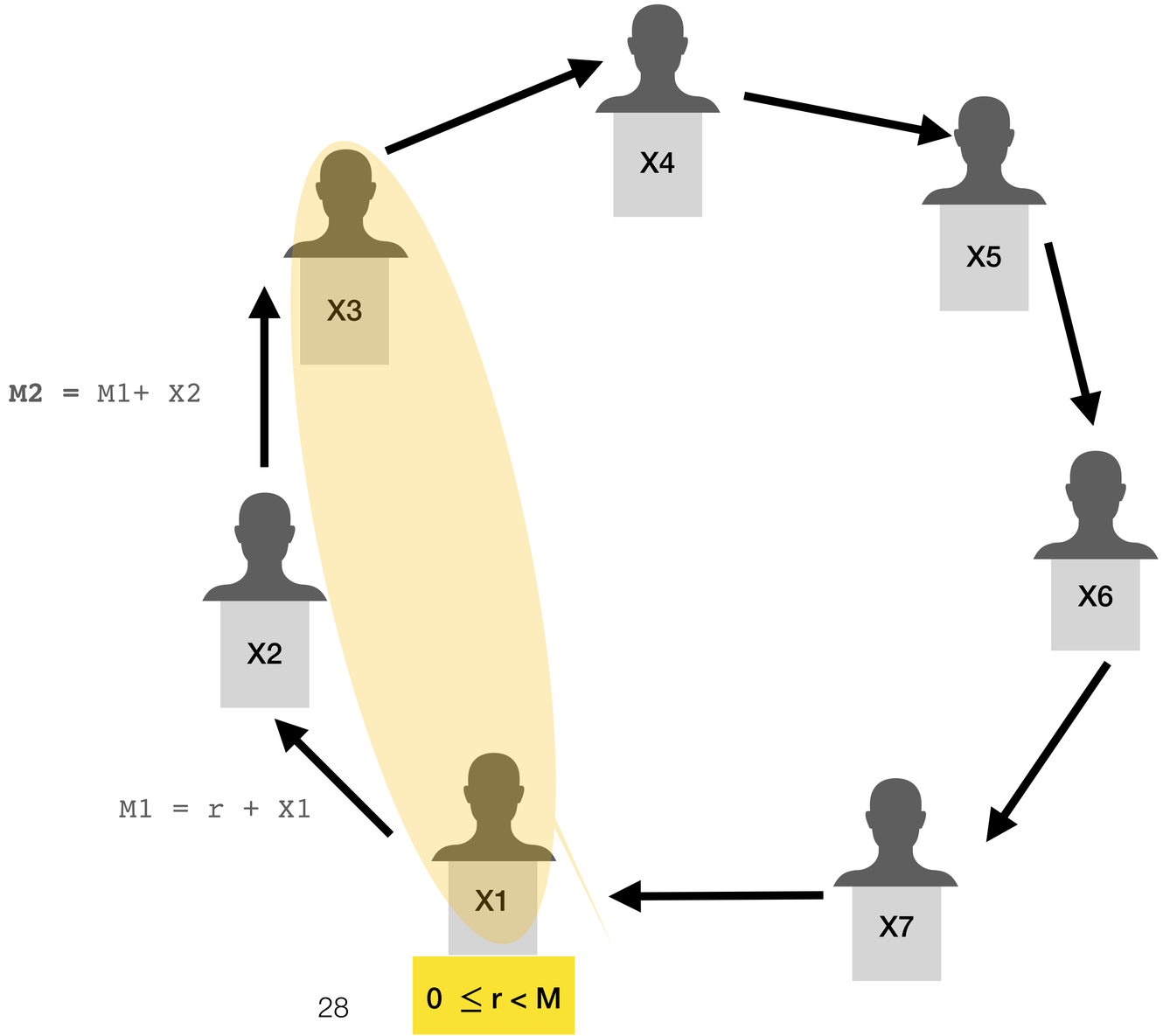
# Secure aggregation: Security concern

What if two neighbour party colludes?

They can identify the secret value of the victim party.

Maximum number of colluding party = 2

Communication overhead = 2 * n (each party sends and receives message)



# Secure aggregation: Security concern

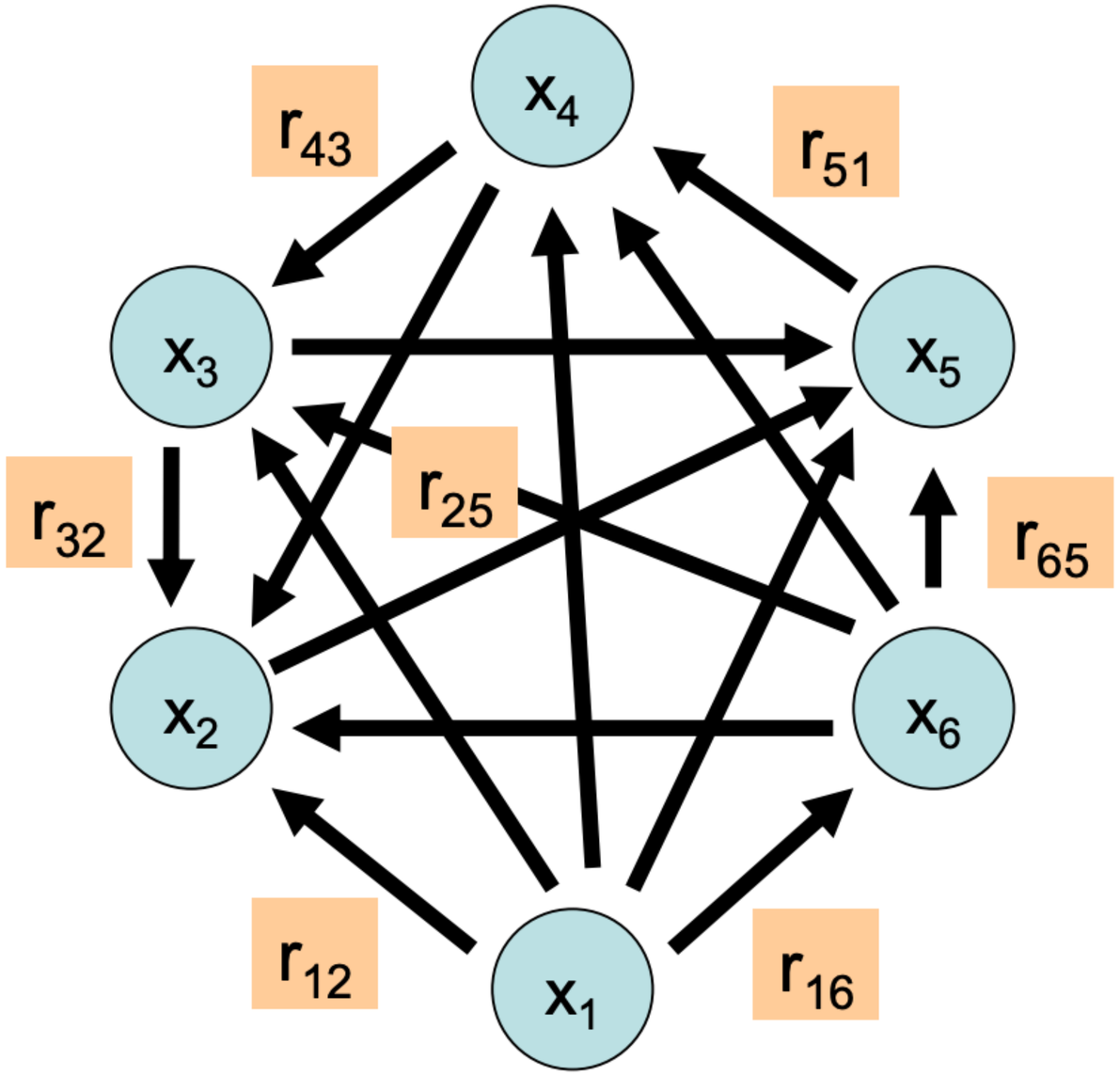
Uses elaborated network.

It has 2 rounds communication.
- 1st round: N choose 2 messages (NC2)
- 2nd round: Party computes values

At the end, masks cancels out and sum is revealed.

*Finding optimal graph for n party requires further computation.*

$Xi + inbox_i - outbox_i$




Bonawitz, Keith, et al. , Practical secure aggregation for privacy-preserving machine learning



*[Yuval Ishai, Talk]*

# More generally

- ***Aim:*** *Parties P1, P2, P3, ..PN wants to securely compute F(X1, X2, X3, …, XN)*
  - *Up to t parties can collude*
  - *Should learn (essentially) nothing but the output*
- ***Questions***
  - *For which function is MPC possible?*
  - *How efficiently?*

**Research from 1980s till now. Most of these research focus on information security and computational security.**

**Several efficiency measures:**
**Communication, rounds, computations**
**- Active area of research**

MPC-CMP: The Newest Innovation in MPC, offers the fastest transaction signing speeds of any MPC algorithm by 800%. (Fireblocks)

# Real/Ideal paradigm

## MPC - Threats and security Requirements

- *Privacy:* No party can obtain any information except own input and output.
- *Correctness:* Output should be correct.
- *Independence:* The input selected by participant should be honest.
- *Guarantee of outputs:* Adversary should not be able to interrupt computation.
- *Fairness:* Corrupted parties receives own output iff the honest parties receive their own output.

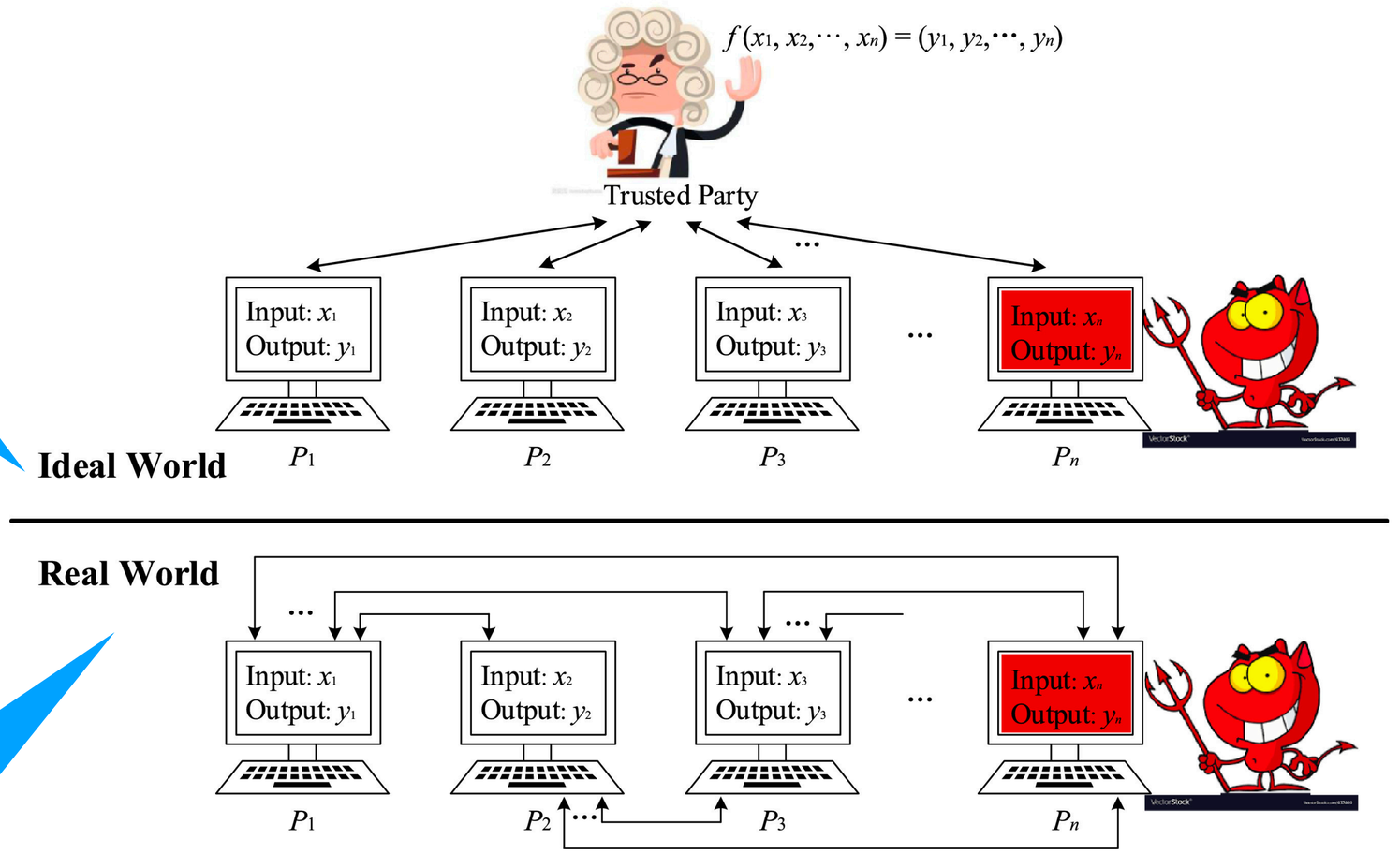
**The list does not give a definition of security, but just a requirement that a security protocol must satisfy.**

**Security definition should include all these requirements and cover all possible attacks.**

*Zhao, Chuan, et al.*

# Real/Ideal paradigm

Security definition should include all these requirements and cover all possible attacks.




In real world, no external party can be trusted. Parties execute the protocol jointly. It is secure if the protocol can simulate the result of the ideal world.

Secure MPC



*Zhao, Chuan, et al.*

# Security Model

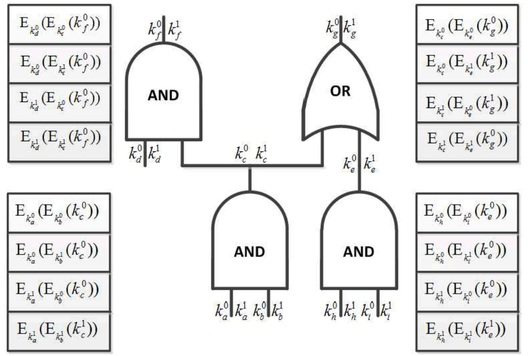
- **Secure MPC** takes security model into account.
- Security model defines the capabilities of adversary.

  - **Semi-honest adversary model**: Corrupted party must execute protocol correctly. Adversary can obtain comprehensive information, and attempts to use that information to obtain additional information. E.g. departments in an organisation.
  - **Malicious adversary model**: Corrupted party arbitrarily deviate from protocol's specifications. Preferred model for joint computation but requires high price on efficiency. E.g. Joint computations with competitors, pass wrong information to cause error.
  - **Covert adversary model**: Semi-honest is too weak, malicious adversary is too inefficient, so the third model, covert was introduced. An adversary may exhibit malicious behaviour but has a given probability of being caught cheating by honest parties. E.g. in financial and political settings



*Zhao, Chuan, et al.*

# Approaches of Secure MPC




## 1. Garbled Circuit Protocol - *Credited Yao, 82*

- The idea is any discrete, fixed size function can be converted into logic gates.
- Enables 2-party secure computation.
- Garbled circuit uses **oblivious transfer**, that can be built using asymmetric cryptography ( E.g. RSA)

### History

- Yao presented the idea in oral presentation [Yao, 5].
- Oded Goldrich documented in 2003 [Goldreich, 12].
- Term was used by Beaver, Micali, and Rogaway in STOC'90 [Beaver, 13].

A Gentle Introduction to Yao's Garbled Circuits [Sophia Yakoubov]

### Protocol

**Step 1:** The underlying function (Circuit) is known to both parties.

**Step 2:** Alice garbles (encrypts) the circuit. Alice is called ***garbler***.

**Step 3:** Alice sends the garbled circuit to Bob with her encrypted input.

**Step 4:** Bob garbles his own input. Only Alice can garble the input. So, he uses oblivious transfer to encrypt his input with the help from Alice.

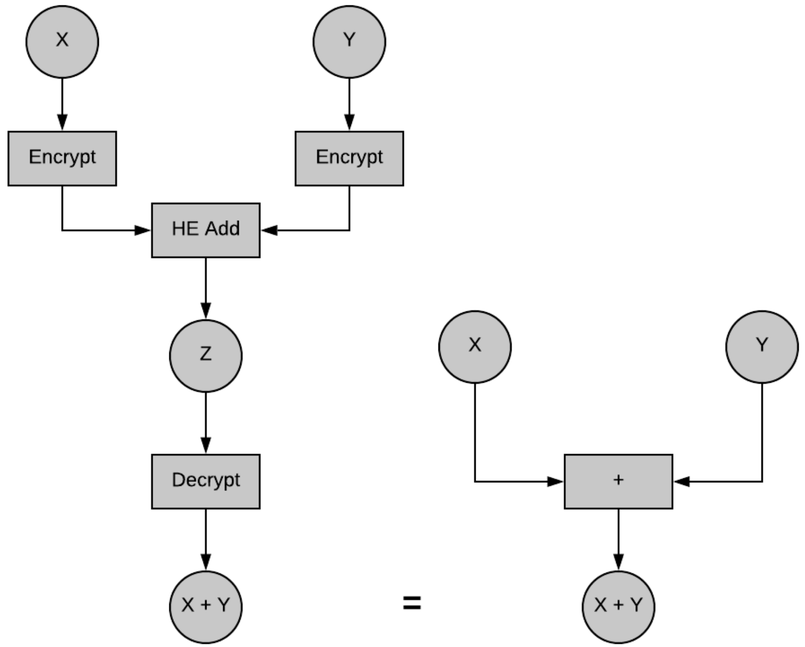
**Step 5:** Bob evaluates the circuit with his encrypted input, and circuit. Bob is called ***evaluator.***

**Step 6:** Alice and Bob communicates to learn the output.

# Approaches of Secure MPC

## 2. Homomorphic Encryption

- It is a cryptographic method that allows mathematical operations to be carryout out on cipher text (encrypted version of data), instead of actual data itself.




**Purpose** is to allow computation on encrypted data.

**Types**:

- Partially Homomorphic Encryption *(supports one type of mathematical operations - e.g. addition or multiplication)*
- Somewhat Homomorphic Encryption *( supports both types of mathematical operation unto certain complexity)*
- Fully Homomorphic Encryption *(Can use both addition and multiplication any number of times; optimisation research has been done since 2009)*



TenSEAL *Benaissa, et al.*

# FHE in user-server scenarios

Emails could be stored encrypted  so the email provider does not know the content of the messages.

Pictures could be uploaded to websites offering image processing capabilities, without the site learning about picture.

Statistics can be performed on medical data without sharing sensitive data.


Privacy preserving search engines

*Fully homomorphic encryption for machine learning, Michele Minelli.*

# Generic Stages of Secure MPC

**INPUT**

**Parties** gives input the function $F(x_1, x_2, x_3, \ldots, x_n)$. $X_{i's}$ are private values for the parties.

**COMPUTATION**

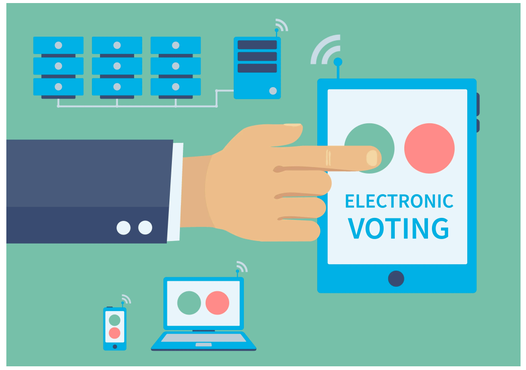
**Computation** occurs on the shared input values.

**OUTPUT**

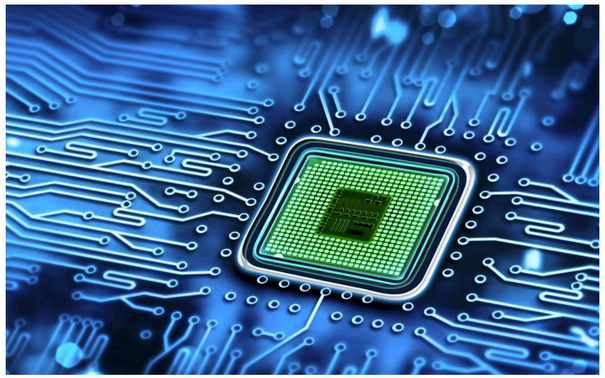
**Output** values of the function $F$ is revealed to the all/some parties.

*Mauland, Atle*

# Practical applications

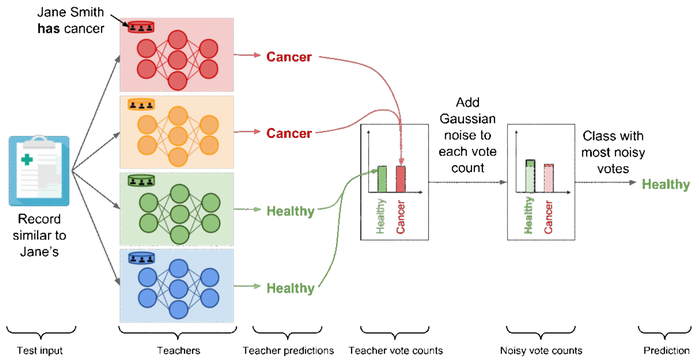



Voting/Trading/bidding/
matching/
Key management/smart
contracts/ auctions



Private circuits

*Intel SGX Explained, [Costan et al., 9]*




Privacy preserving machine
learning

Business and finance

Any are where “good guys”
trying to achieve a common
goal (computation) in
presence of “bad guys”.

*Secure multiparty computations on bitcoin [Marcin, et al., 18]*

# Secure MPC and digital assets

- A **digital asset** is anything that exists in a digital format and comes with the right to use.
- **Example:** Digital documents, audible content, motion picture, digital data, digital currencies etc.


- **Hot storage:** Private key is held online.
- **Cold storage:** Private key is held offline.
- **Hardware wallet:** Private key is held offline on a physical device.

Single point failure, security and confidentiality

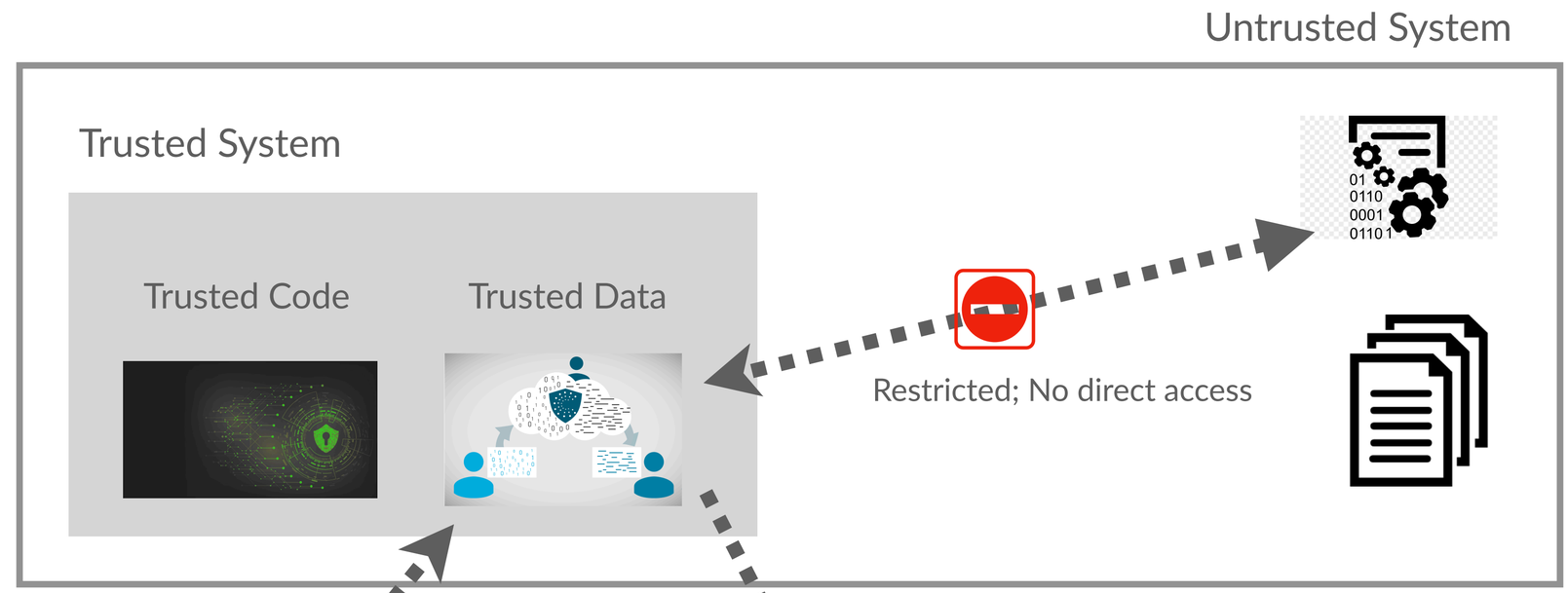
Slow transaction, possible attack spoofing, credential thefts

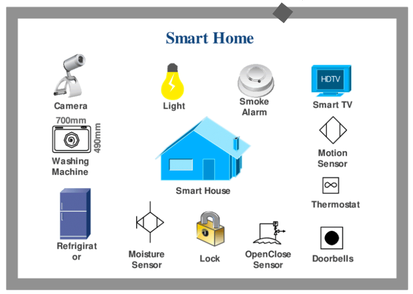
Hardware failure, chances of loosing them, speed issues

- **Secure MPC** for private key security in digital wallets.

# Trusted Execution Environment (TEE) and Secure MPC

**Real time performance** Secure MPC is essential, which is still challenging and ongoing research.




**For Example,** Intel Software Guard Extension (SGX) has trusted memory regions where code and data remains in unencrypted form. [Costan et al., 9]

**For untrusted region,** use of Secure MPC.

# Benefits of MPC

- No trusted third-parties see the data
- Eliminates tradeoff between data usability and data privacy
- GDPR and sovereign data privacy compliance
- High accuracy and precision

# Limitations of SMPC

- Computational overhead
- High communication costs between parties.
- Protocol does not provide security beyond trusted-party emulation

Marriage Proposal Problem

Men's Preference List

| Man | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|
| Victor | Bertha | Amy | Diane | Erika | Clare |
| Wyatt | Diane | Bertha | Amy | Clare | Erika |
| Xavier | Bertha | Erika | Clare | Diane | Amy |
| Yancey | Amy | Diane | Clare | Bertha | Erika |
| Zeus | Bertha | Diane | Amy | Erika | Clare |

best worst

Women's Preference List

| Woman | 1st | 2nd | 3rd | 4th | 5th |
|---|---|---|---|---|---|
| Amy | Zeus | Victor | Wyatt | Yancey | Xavier |
| Bertha | Xavier | Wyatt | Yancey | Victor | Zeus |
| Clare | Wyatt | Xavier | Yancey | Zeus | Victor |
| Diane | Victor | Zeus | Yancey | Xavier | Wyatt |
| Erika | Yancey | Wyatt | Zeus | Xavier | Victor |

best worst

No technological way to ensure that the users honestly provide their input to the trusted party. Nothing can prevent one party to feed wrong input.

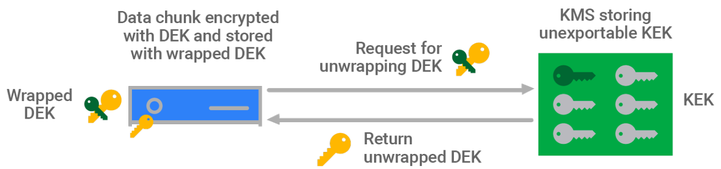
The Marriage Proposals Problem: Fair and Efficient Solution for Two-Party Computations, [Montreuil et al. , 19]

https://www.youtube.com/watch?v=Qcv1IqHWAzg

# What’s next for Secure MPC?

- MPC has quickly become the standard for securing digital assets.
- Major financial institutions –
  - Celsius (biggest US crypto lending desk), and
  - Revolut (Banking) – have announced their transition to Secure MPC.

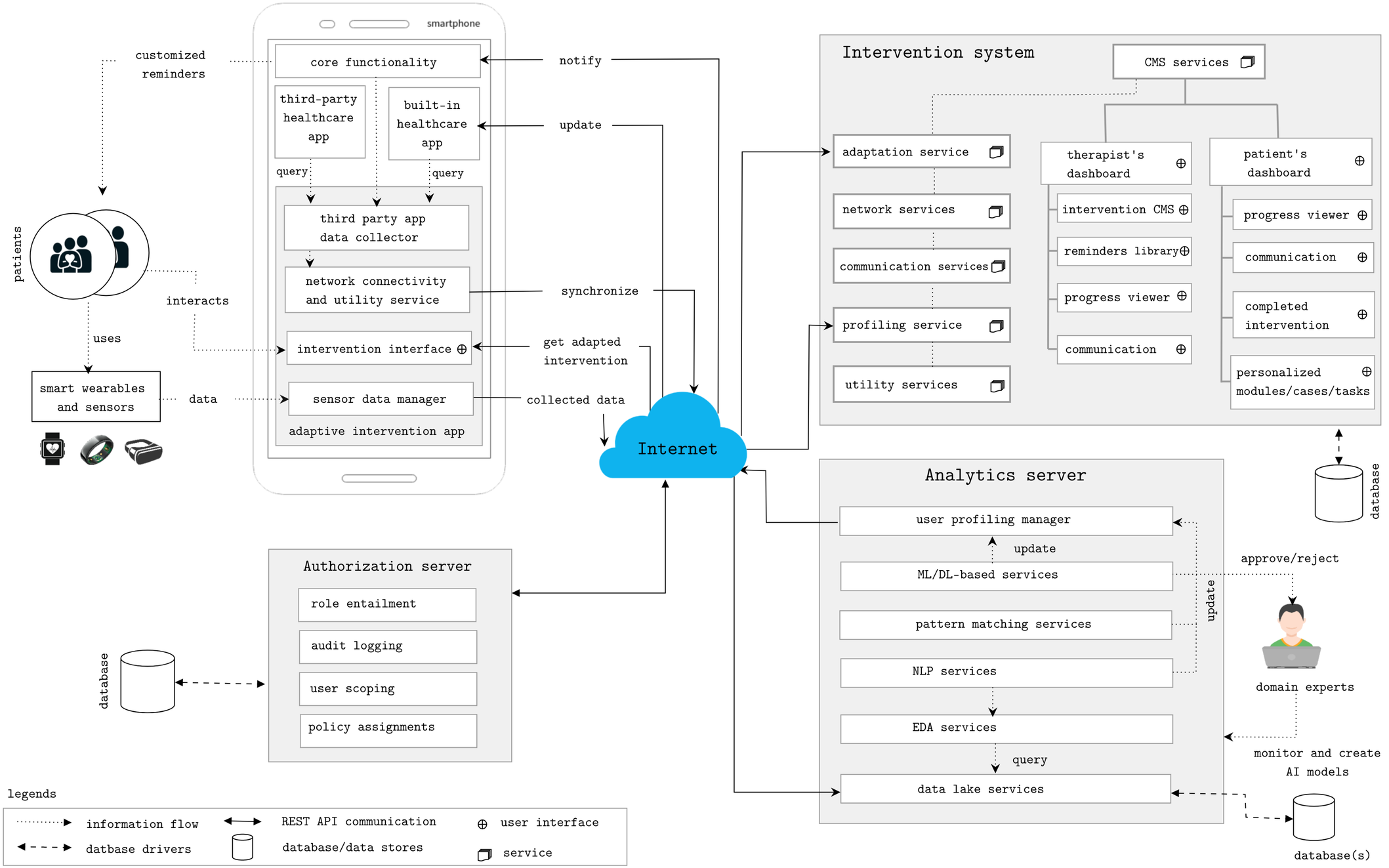
inpher






From keys to databases—real-world applications of secure multi-party computation [Archier, 20]

# SMPC/IDPT


smartphone
customized reminders
core functionality
third-party healthcare app
built-in healthcare app
query
query
third party app data collector
network connectivity and utility service
intervention interface
sensor data manager
adaptive intervention app
notify
update
synchronize
get adapted intervention
collected data
patients
interacts
uses
smart wearables and sensors
data
Internet
Intervention system
CMS services
adaptation service
network services
communication services
profiling service
utility services
therapist's dashboard
intervention CMS
reminders library
progress viewer
communication
patient's dashboard
progress viewer
communication
completed intervention
personalized modules/cases/tasks
database
Analytics server
user profiling manager
update
ML/DL-based services
pattern matching services
NLP services
EDA services
query
data lake services
update
approve/reject
domain experts
monitor and create AI models
database(s)
Authorization server
role entailment
audit logging
user scoping
policy assignments
database
legends
information flow
datbase drivers
REST API communication
database/data stores
user interface
service

1. Hamming, R. W. (1950). Error detecting and error correcting codes. The Bell system technical journal, 29(2), 147-160.

2. Shannon, Claude Elwood. "A mathematical theory of communication." The Bell system technical journal 27.3 (1948): 379-423.

3. Zhao, Chuan, et al. "Secure multi-party computation: theory, practice and applications." Information Sciences 476 (2019): 357-372.

4. Yao, Andrew Chi-Chih (1986). "How to generate and exchange secrets". 27th Annual Symposium on Foundations of Computer Science (SFCS 1986). Foundations of Computer Science, 1986., 27th Annual Symposium on. pp. 162–167. doi:10.1109/SFCS.1986.25. ISBN 978-0-8186-0740-0.

5. Andrew C. Yao. Protocols for secure computations. In SFCS '82: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.

6. Lin, Hsiao-Ying, and Wen-Guey Tzeng. "An efficient solution to the millionaires' problem based on homomorphic encryption." International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2005.

7. Kumar, Ashish, and Anupam Gupta. "Some efficient solutions to yao's millionaire problem." arXiv preprint arXiv:1310.8063 (2013).

8. https://inpher.io/technology/what-is-secure-multiparty-computation/

9. Costan, Victor, and Srinivas Devadas. "Intel sgx explained." IACR Cryptol. ePrint Arch. 2016.86 (2016): 1-118.

10. Bonawitz, Keith, et al. "Practical secure aggregation for privacy-preserving machine learning." proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017.

11. Mauland, Atle. Realizing distributed RSA using secure multiparty computations. MS thesis. Institutt for telematikk, 2009.

12. Goldreich, Oded (2003). "Cryptography and Cryptographic Protocols". Distributed Computing - Papers in Celebration of the 20th Anniversary of PODC. 16 (2–3): 177–199.

13. Beaver, Donald; Micali, Silvio; Rogaway, Phillip (1990). The Round Complexity of Secure Protocols. Proceeding STOC '90 Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing. pp. 503–513.

14. A. Benaissa, B. Retiat, B. Cebere, A.E. Belfedhal, "TenSEAL: A Library for Encrypted Tensor Operations Using Homomorphic Encryption", ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML 2021).

15. Michele Minelli. Fully homomorphic encryption for machine learning. Cryptography and Security [cs.CR]. Université Paris sciences et lettres, 2018. English. ffNNT : 2018PSLEE056ff. fftel-01918263v2f

16. https://www.youtube.com/watch?v=xwxkp4fMWsk

17. Sherman, Alan T.; Javani, Farid; Zhang, Haibin; Golaszewski, Enis (January 2019). "On the Origins and Variations of Blockchain Technologies". IEEE Security Privacy. 17 (1): 72–77.

18. Andrychowicz, Marcin, et al. "Secure multiparty computations on bitcoin." 2014 IEEE Symposium on Security and Privacy. IEEE, 2014.

19. Montreuil A., Patarin J. (2004) The Marriage Proposals Problem: Fair and Efficient Solution for Two-Party Computations. In: Canteaut A., Viswanathan K. (eds) Progress in Cryptology - INDOCRYPT 2004. INDOCRYPT 2004. Lecture Notes in Computer Science, vol 3348. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-30556-9_4

20. Archer, David W., et al. "From keys to databases—real-world applications of secure multi-party computation." The Computer Journal 61.12 (2018): 1749-1771.

# Thank you !

Questions?

**A homomorphism is a map between two algebraic structures of the same type (that is of the same name), that preserves the operations of the structures.**

## Additive Homomorphism

f(x) = x + x

For, a = 1, b = 2,

f(a) + f(b) = f (a + b)

=> f(1) + f (2) = f (1 + 2)

## Multiplicative Homomorphism

f(Sx) = Sf(x) where S = Scalar

f(x) = 2x

For, S = 4

f(4x) = 2 (4x) = 8x = 4 (2x) = 4 f(x)