# Secure Multi-Party Computation Based on Quantum Technologies

Zeinab Rahmani
Prof. Luis Barbosa
Prof. Armando Pinto

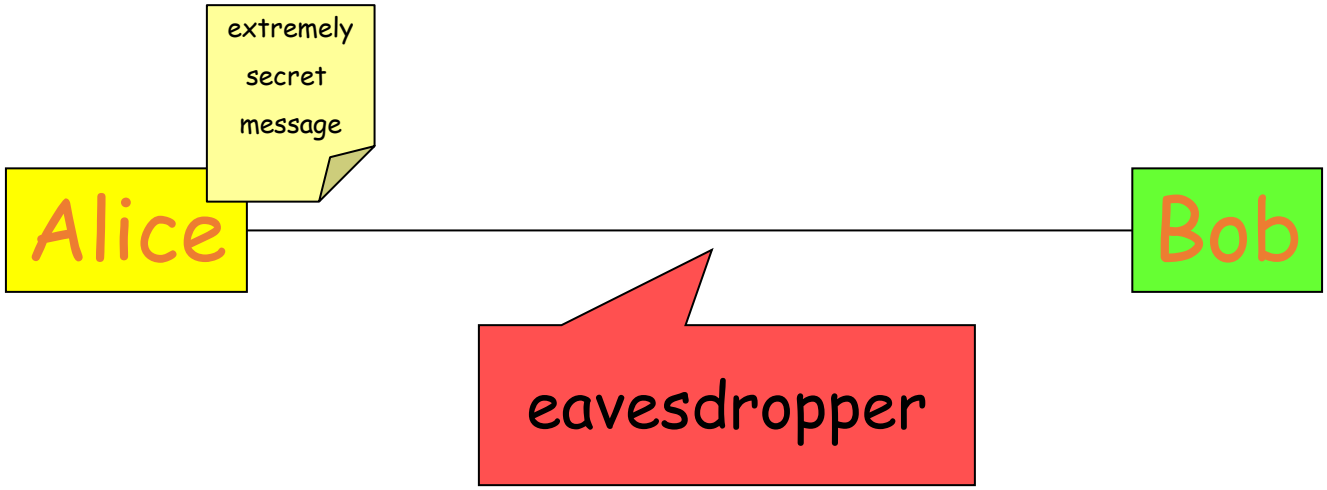
Univesity of Aveiro
2023

# Overview

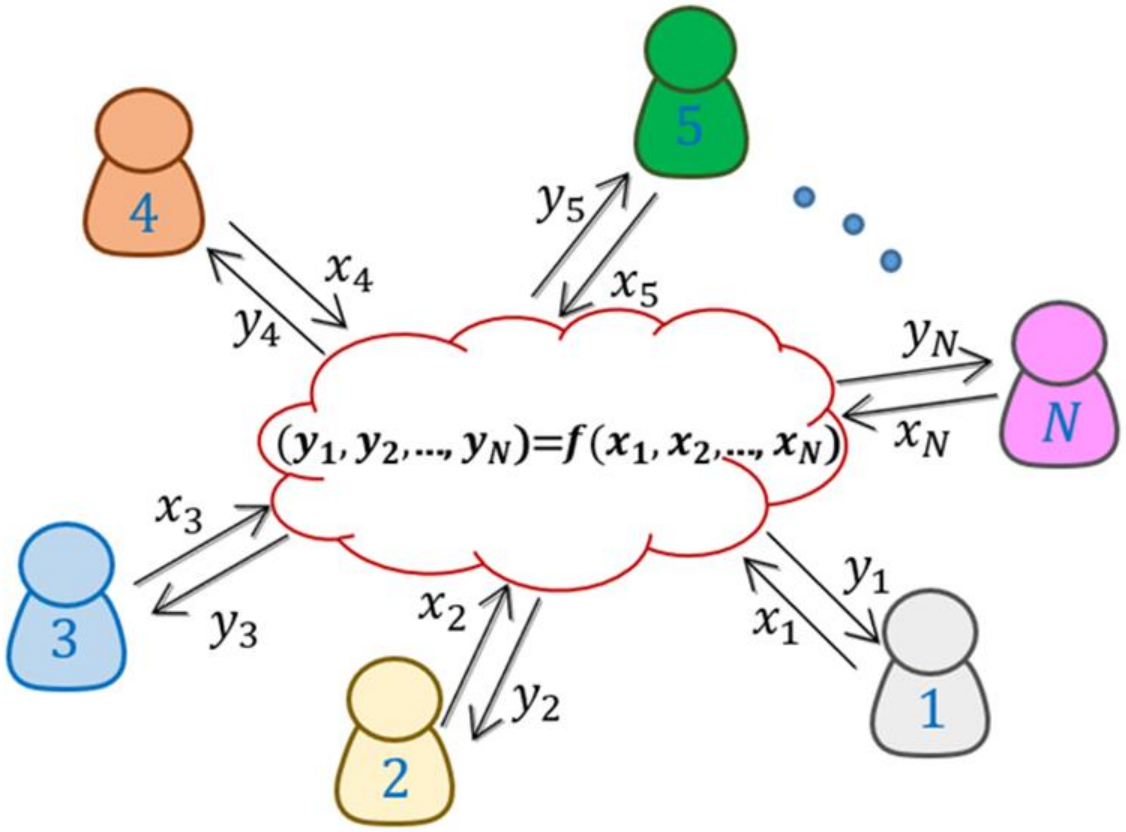
- SMC– a little bit of history
- SMC Applications
- SMC Implementation
- SMC Simulation – QisKit
- Starting Points

# Secure Multiparty Computation

## is the science of "secret collaboration"

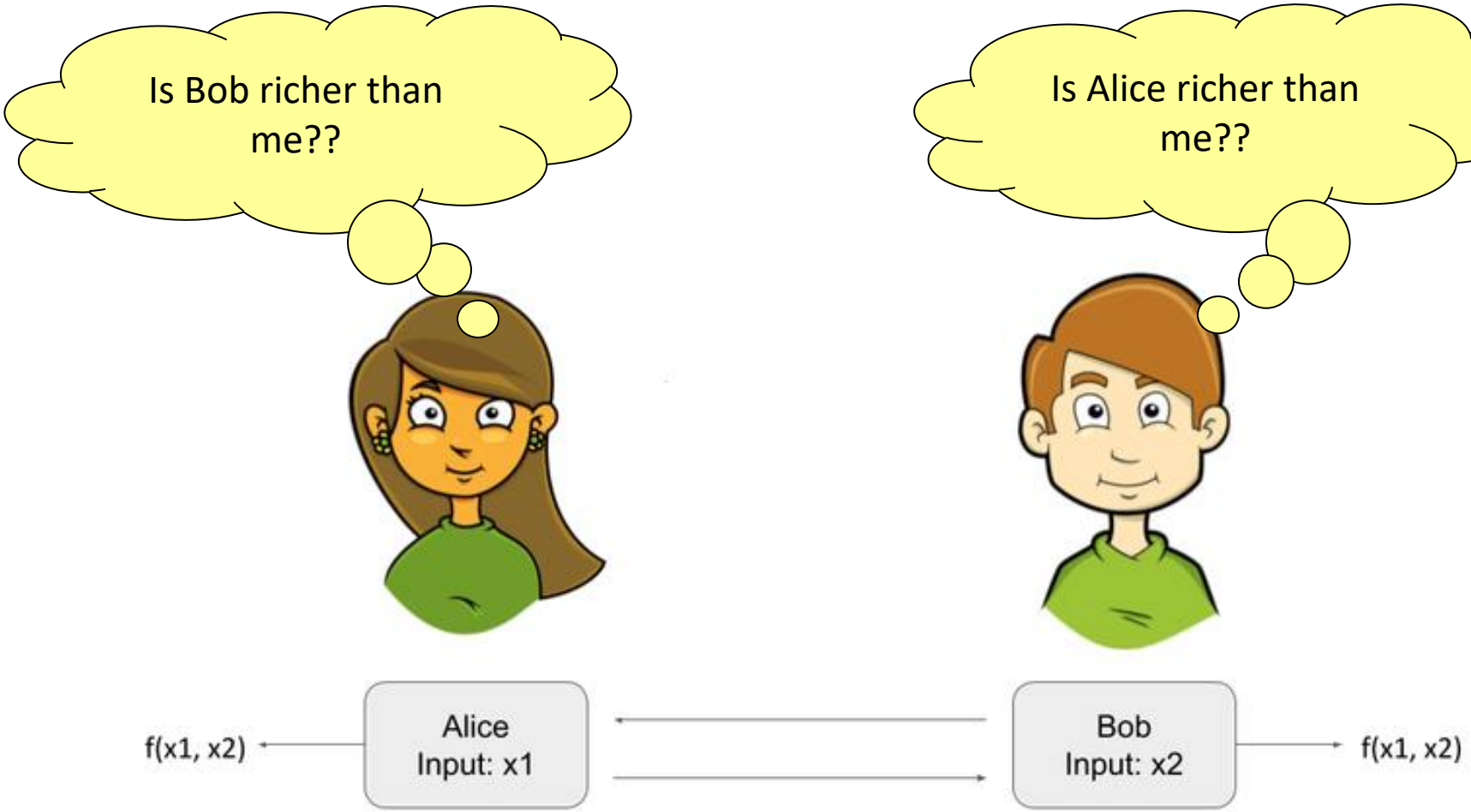


# Secure Multi-Party Computation



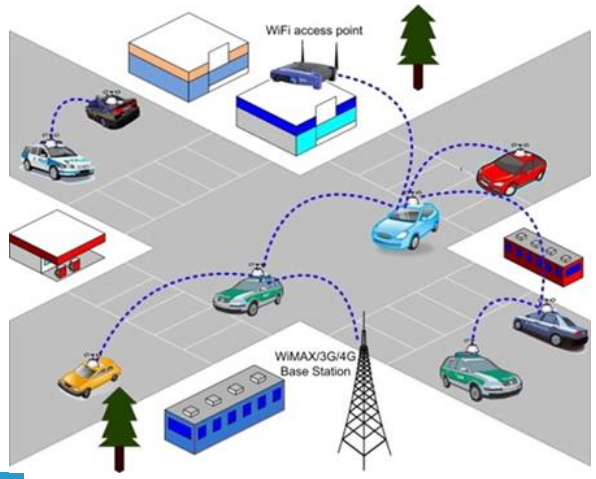

N parties compute a function with their inputs in a way that nobody has access to other' inputs.

# Yao's Millionaires' problem

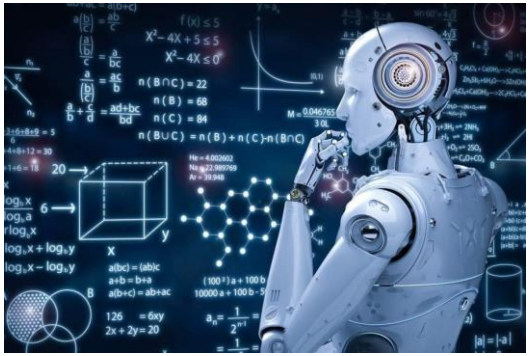


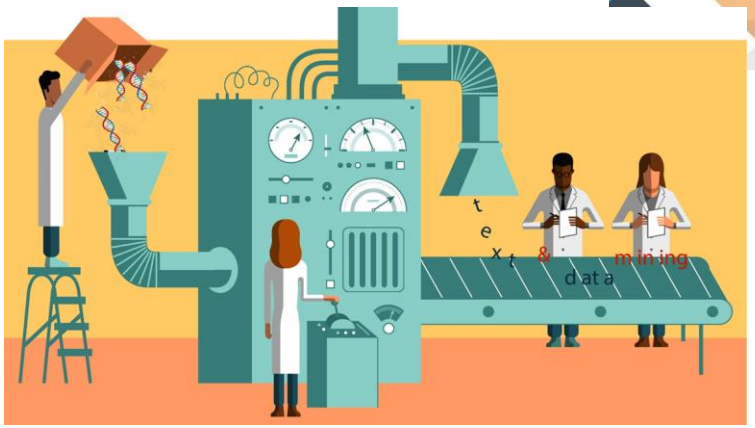
# SMC Applications

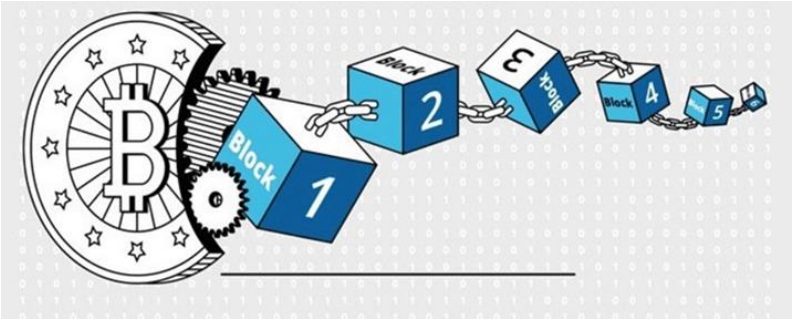
- ✓ Vehicular Network
- ✓ Data Mining
- ✓ Electronic Voting
- ✓ Block chain
- ✓ Machine Learning
- ✓ Digital Signature
- ✓ …









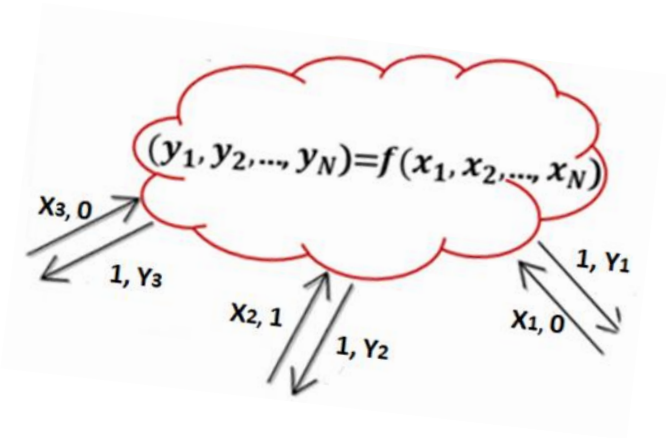
# Collision Detection in Vehicular Network

Vehicle Location

Accident Observation

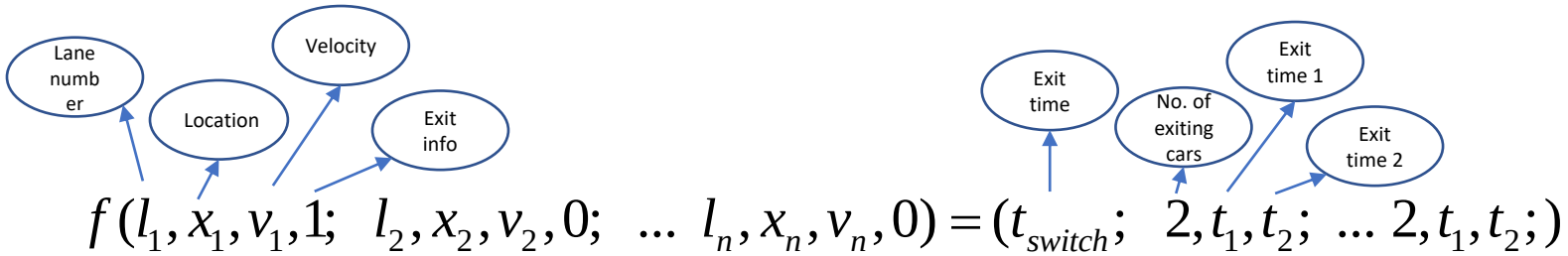
Inputs
- Party 1 -> ($X_1$ , 0)
- Party 2 -> ($X_2$, 1) =>> This party reports the accident
- Party 3 -> ($X_3$, 0)

Outputs
- Party 1 -> (1, $|X_2 - X_1|$)
- Party 2 -> (1, $|X_2 - X_2|$)
- Party 3 -> (1, $|X_2 - X_3|$)

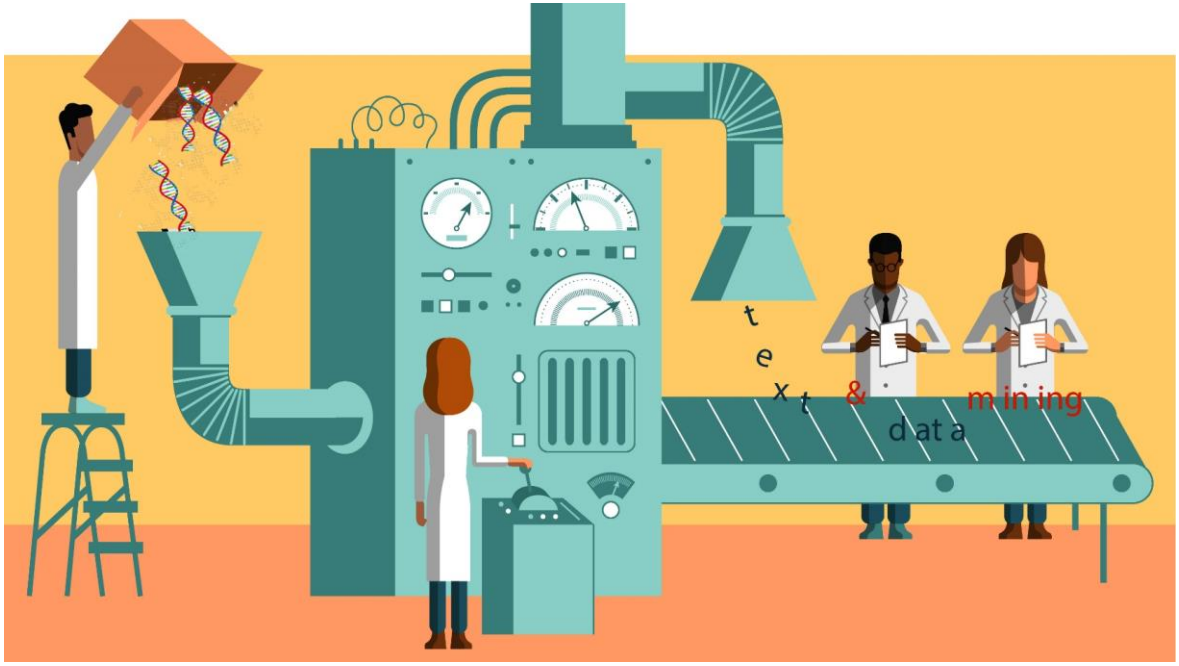


# SMC for Vehicular Network

Vehicles should be able broadcast any sort of information, such that their privacy is fully guaranteed.

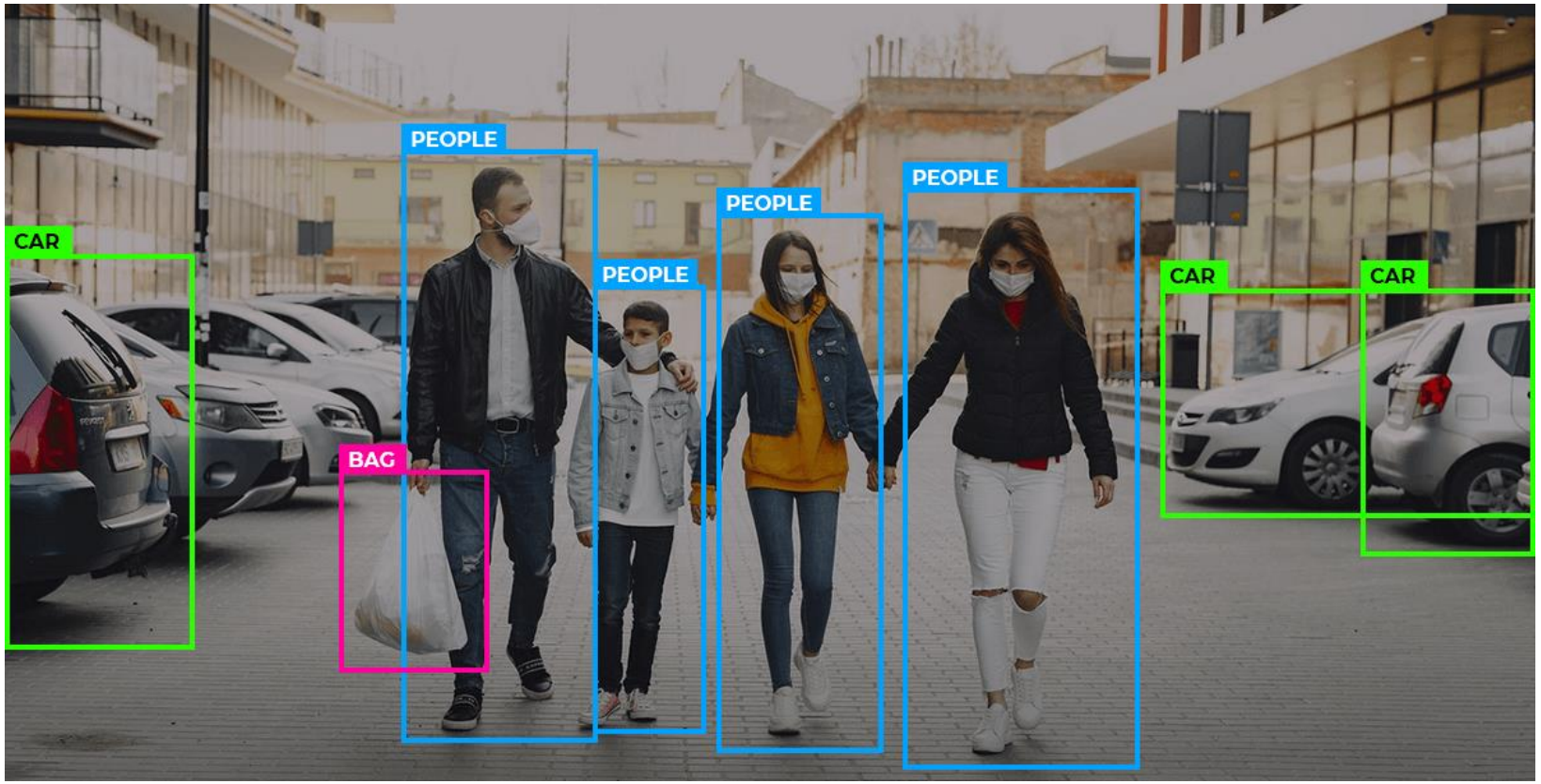



# SMC for Genome Data Mining




Genomic data mining aims to discover valuable information and relationships within the vast amount of genomic data available, providing researchers with a deeper understanding of genetics and potential applications in fields such as medicine, agriculture, and evolutionary biology.

# SMC for Machine Learning



# SMC for Machine Learning

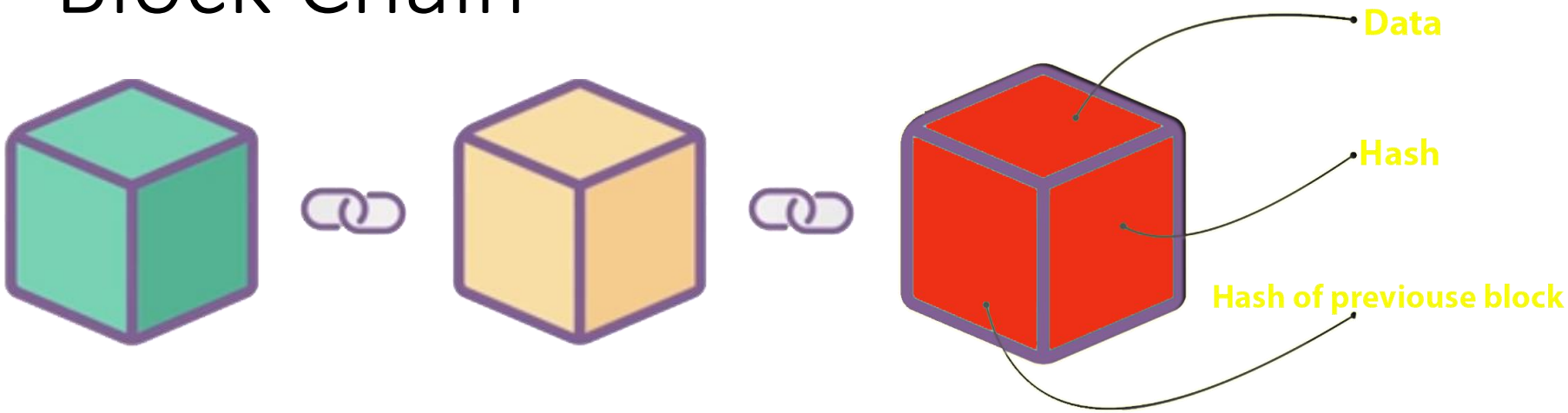
**Scenario 1:**
Multiple parties want to train a machine learning model on their combined datasets without sharing the raw data.

**Use of SMPC:**
SMPC allows these parties to jointly compute model updates without revealing their individual training samples. The model's parameters are updated collaboratively, enhancing the overall model without exposing sensitive data.

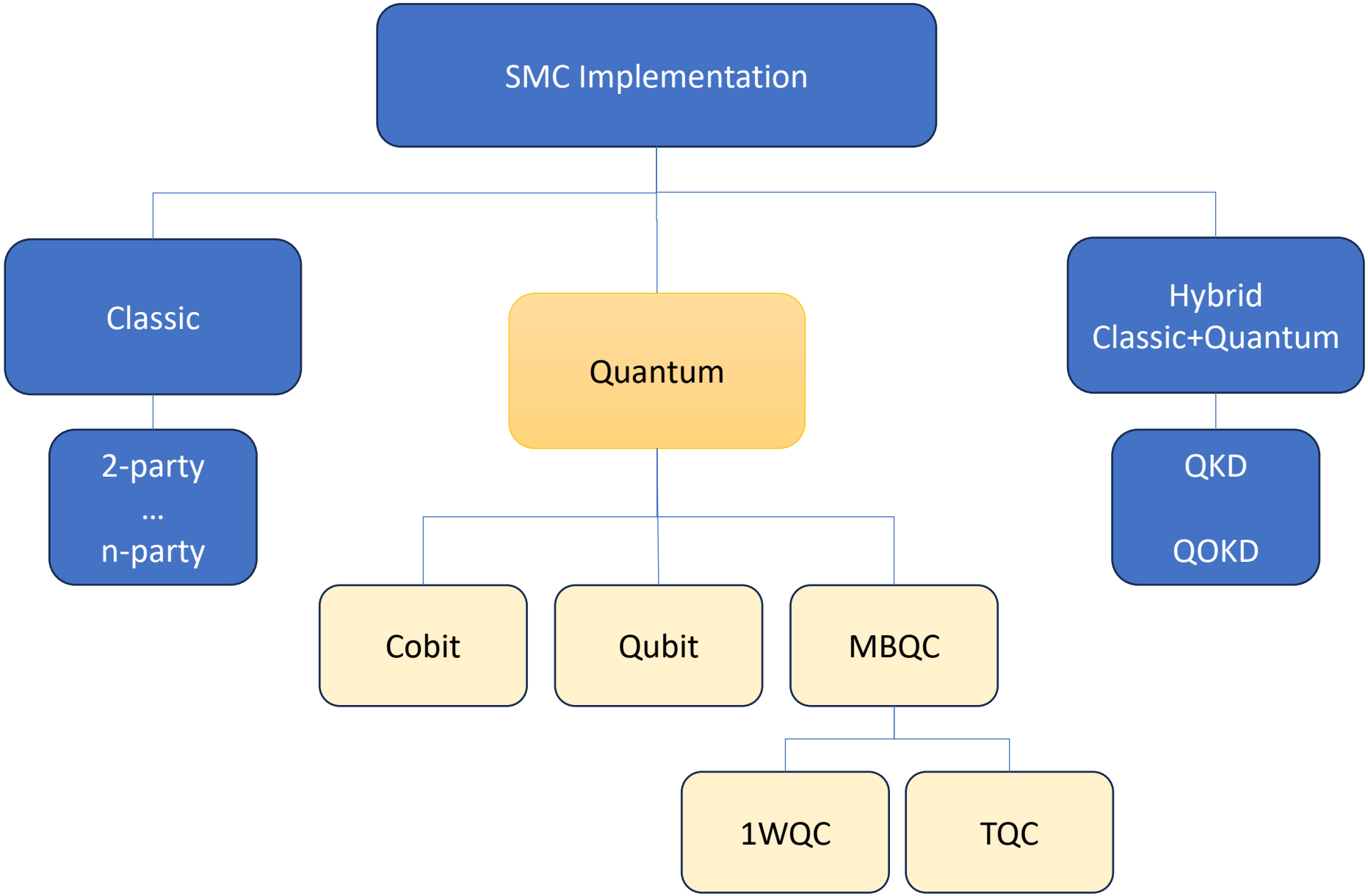
**Scenario 2:**
A model is trained on decentralized data, and parties want to make predictions on new data without sharing it.

**Use of SMPC:**
The parties can use SMPC to jointly compute predictions on new data while keeping the data itself confidential. Each party contributes to the computation without disclosing its specific input.

# Block Chain




- Distributed Ledger Technologies (DLTs)
- Open to anyone (Decentralized)
- Immune to hackers
- Data in blocks cannot be altered
- Cryptography makes information stored in blocks secure and reliable


SMC Implementation
Classic
2-party
...
n-party
Quantum
Cobit
Qubit
MBQC
1WQC
TQC
Hybrid
Classic+Quantum
QKD
QOKD

# Classical SMC

2-party protocols

...

N-party protocols

Yao's Garbled Circuits (1982): This was one of the pioneering protocols in the field. It involves a two-party computation model where one party encrypts its input, and the other party evaluates the circuit to obtain the result without learning the input.

Goldwasser-Micali-Wigderson Protocol (1987): This protocol is based on the concept of zero-knowledge proofs and is used for secure two-party computation.

Shamir's Secret Sharing (1979): Shamir's Secret Sharing is often employed as a building block for SMC.

GMW Protocol (Goldreich, Micali, and Wigderson - 1987): The GMW protocol is a general framework for secure multiparty computation.

Homomorphic Encryption: Homomorphic encryption allows computations to be performed on encrypted data.

# Problem Definition and Motivation

- To implementation the SMC technology, we need a huge number of OTs which makes the practical implementation of SMC-based application in real-life scenarios.

- Classical SMC protocols tend to require excessive computation leading to low efficiency and Low Security.

- Classical SMC which are based on public key cryptography some and computational methods such as prime numbers factorization or discrete logarithm cannot be considered secure in the presence of a quantum computer.

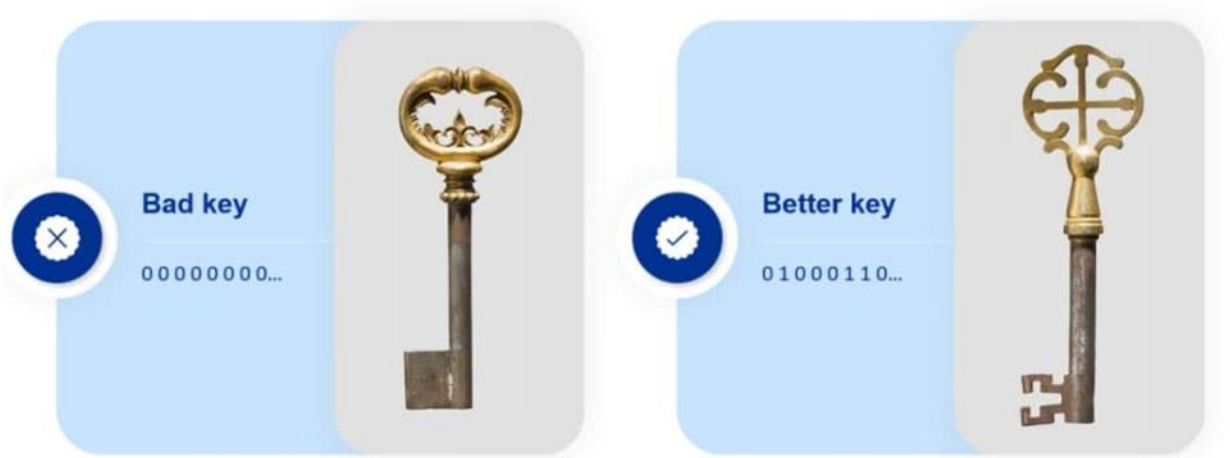
# Hybrid (Classic + Quantum)

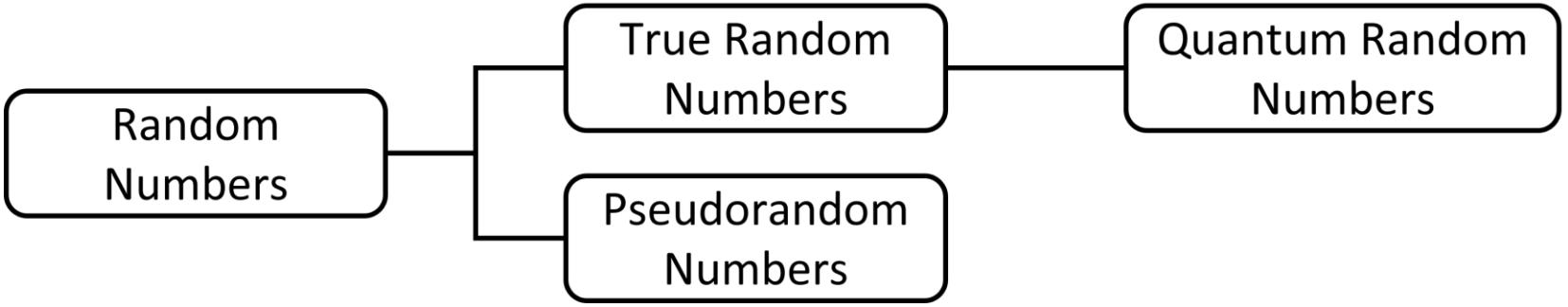
In hybris approach classical protocols are used along side with quantum technologies

**Quantum Technologies**

- ✓ Quantum Random Number Generators (QRNG)
- ✓ Quantum Key Distribution (QKD)
- ✓ Quantum Oblivious Key Distribution (QOKD)
- ✓ Quantum Oblivious Transfer (QOT)

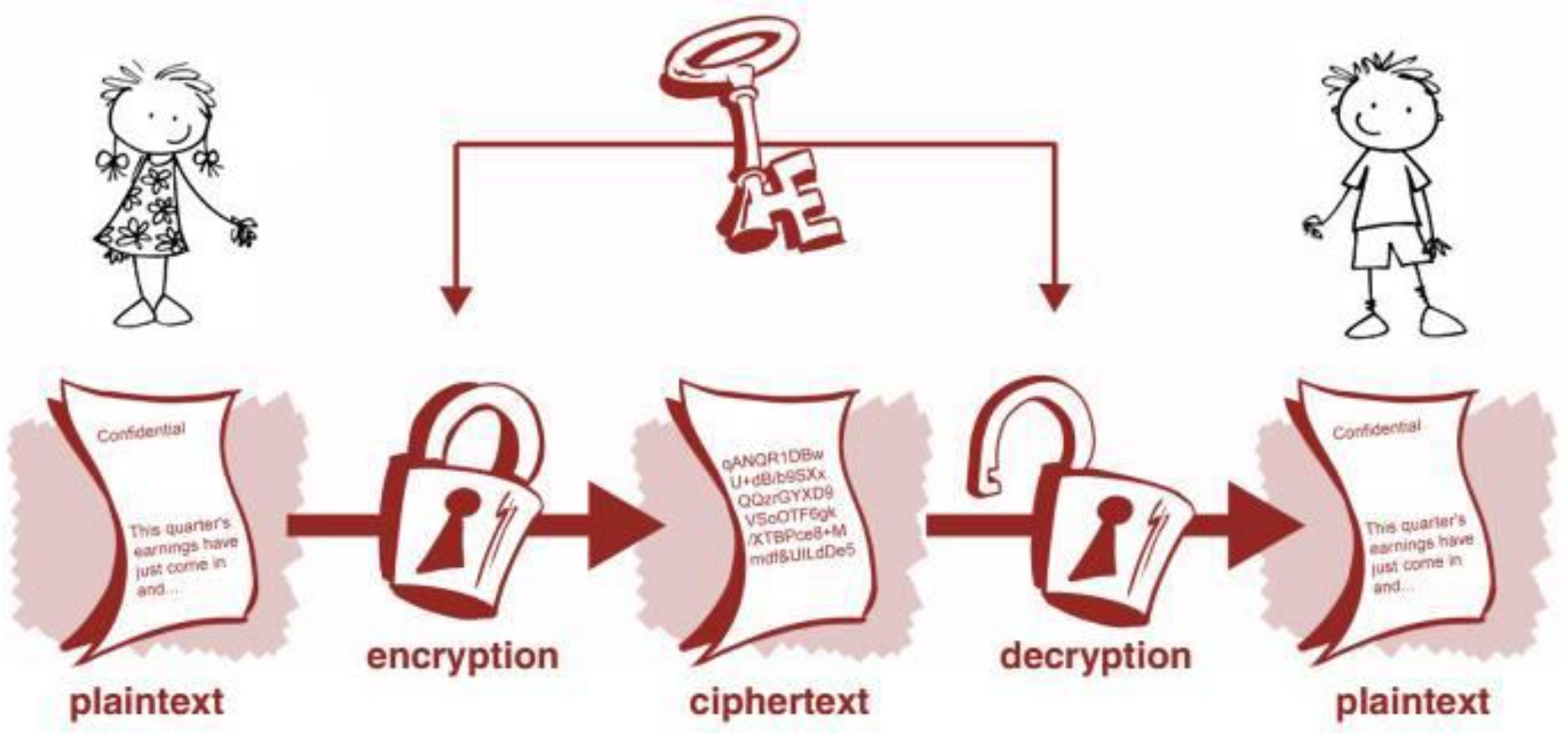
# Quantum Random Number Generators



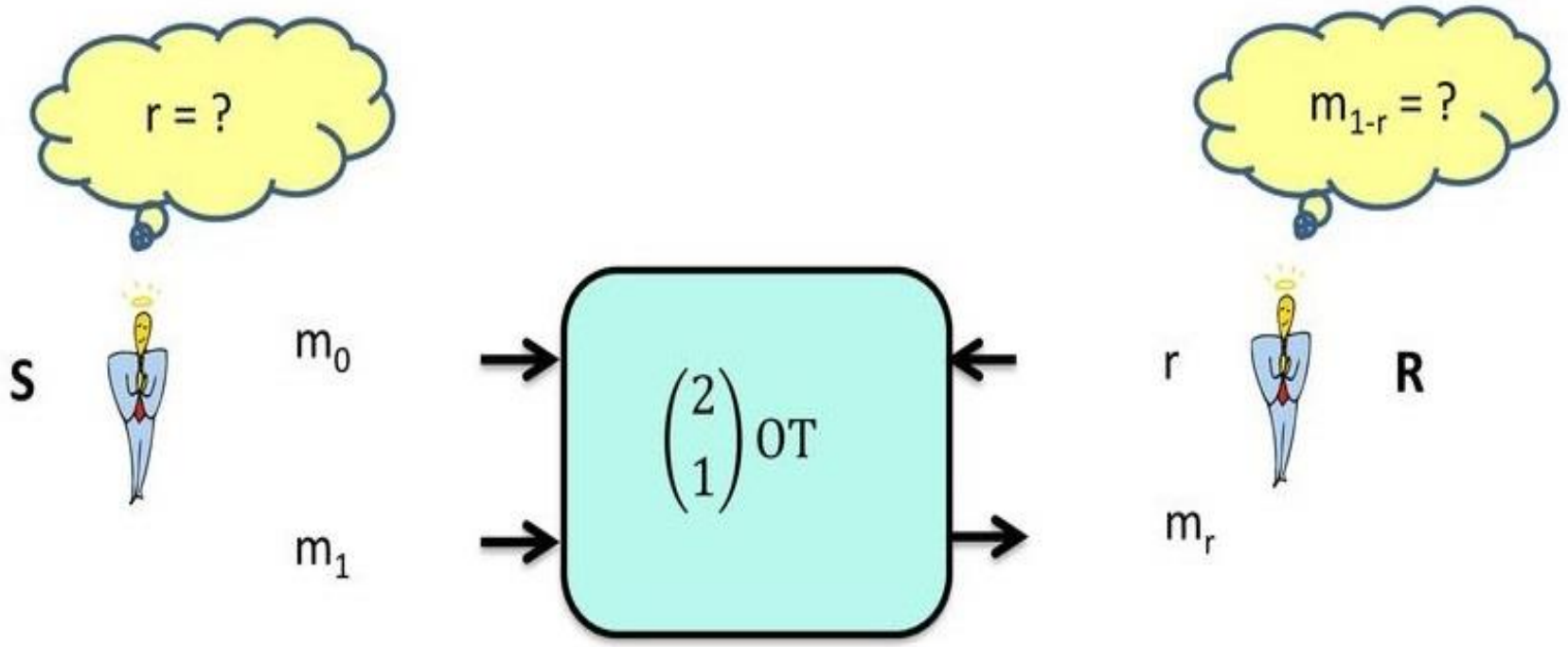




1. Pseudorandom numbers: Not truly random
2. True random numbers: Randomness is derived from the probabilistic nature of quantum physics.

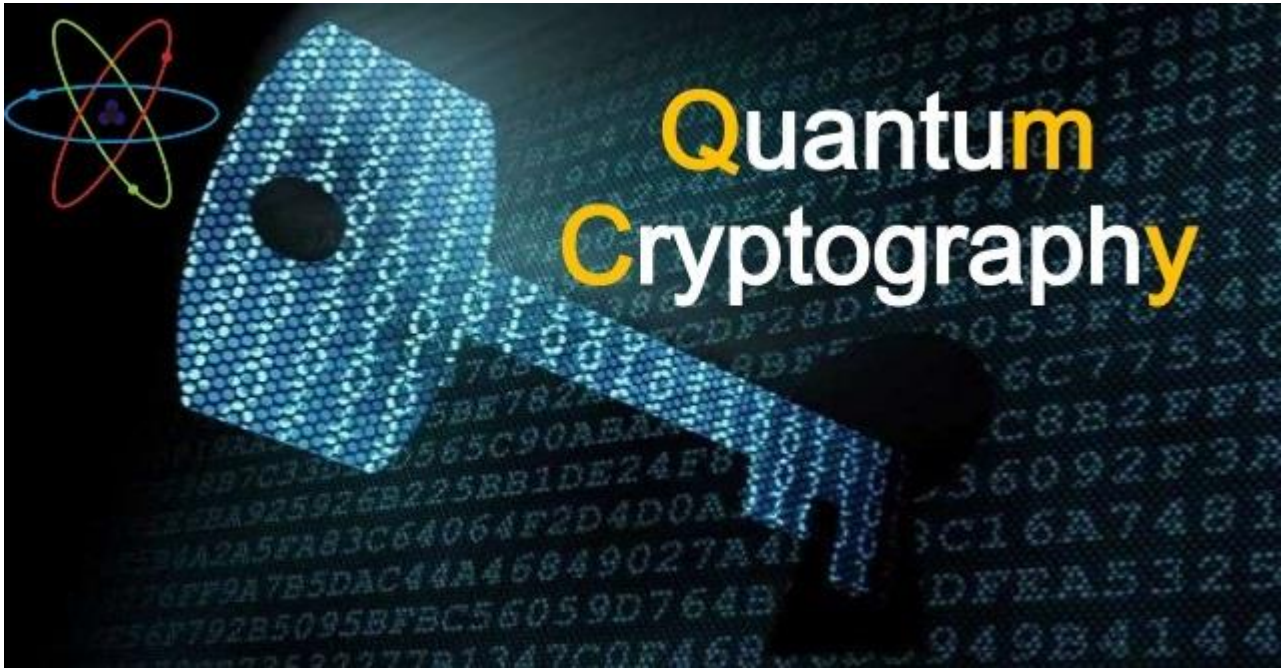
# QKD (Quantum Key Distribution)



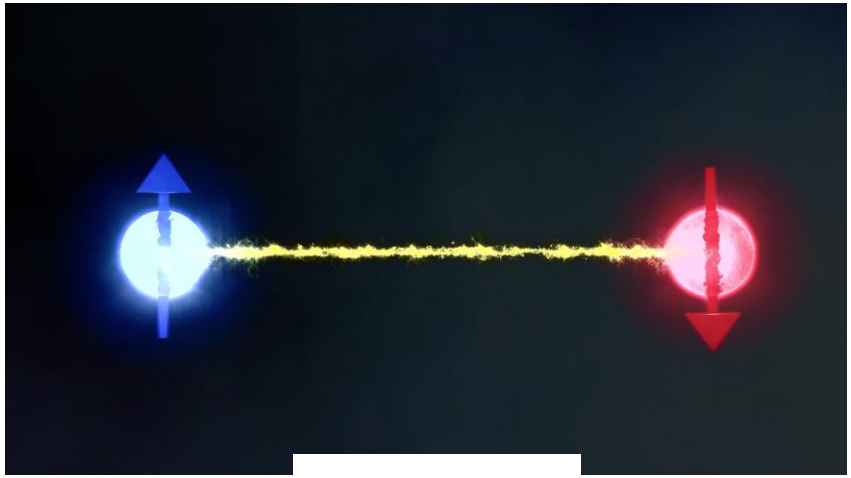
# Quantum Oblivious Transfer

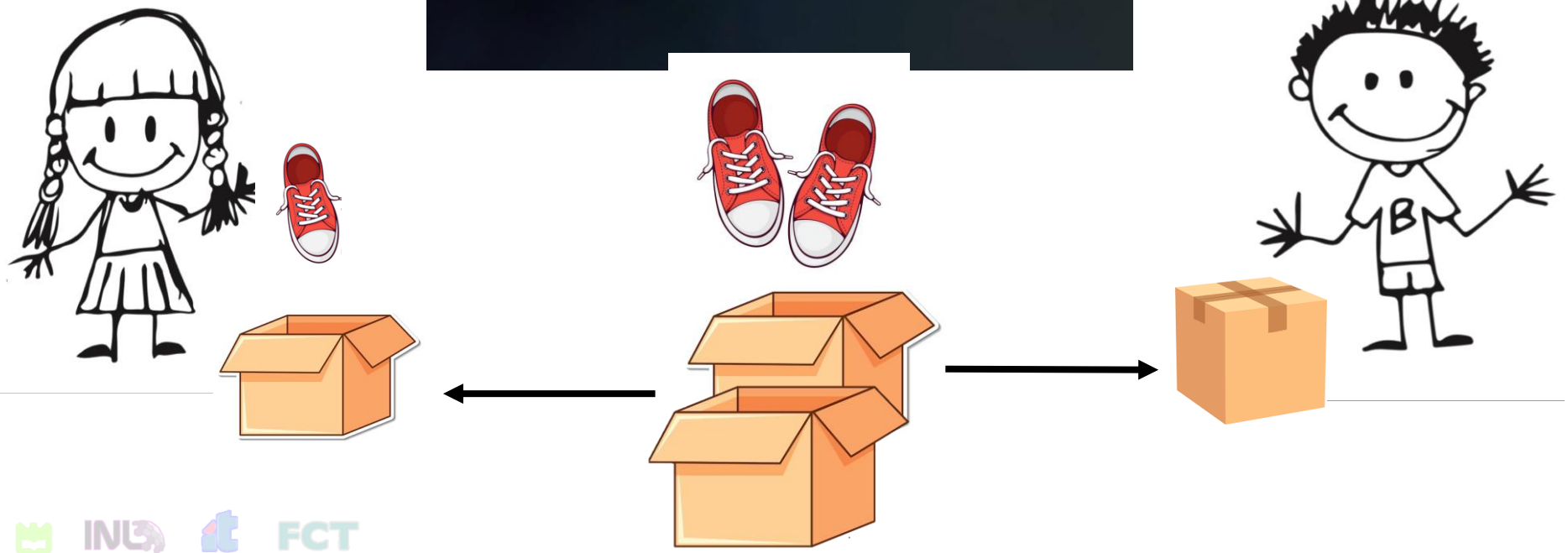


# Fully Quantum SMC



# Entanglement

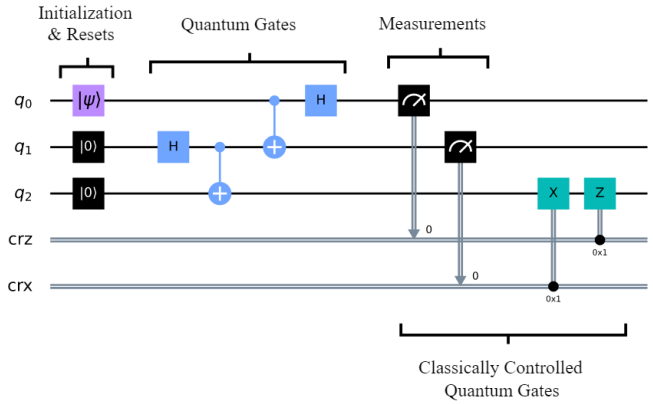




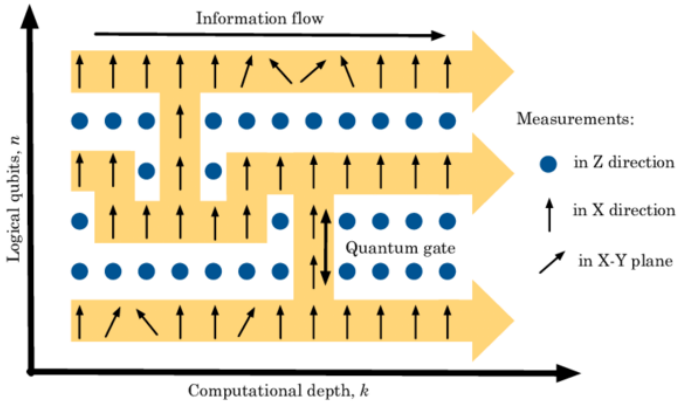
# Measurement Based Quantum Computation (MBQC)

Quantum computing:

1. Circuit model -> information is manipulated via logical gates.



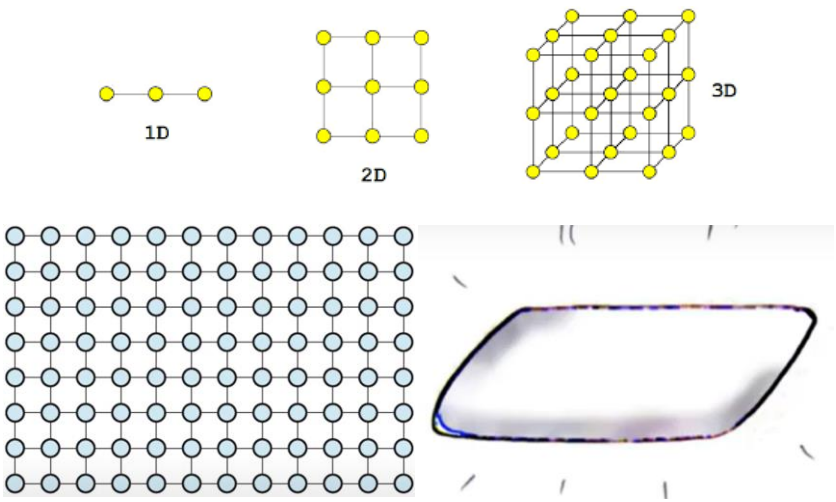

2. MBQC -> information is processed by measuring single-qubits on an entangled multi-qubit resource state.




MBQC -> preparing an entangled resource state, usually a cluster state or graph state, then performing single qubit measurements on it.

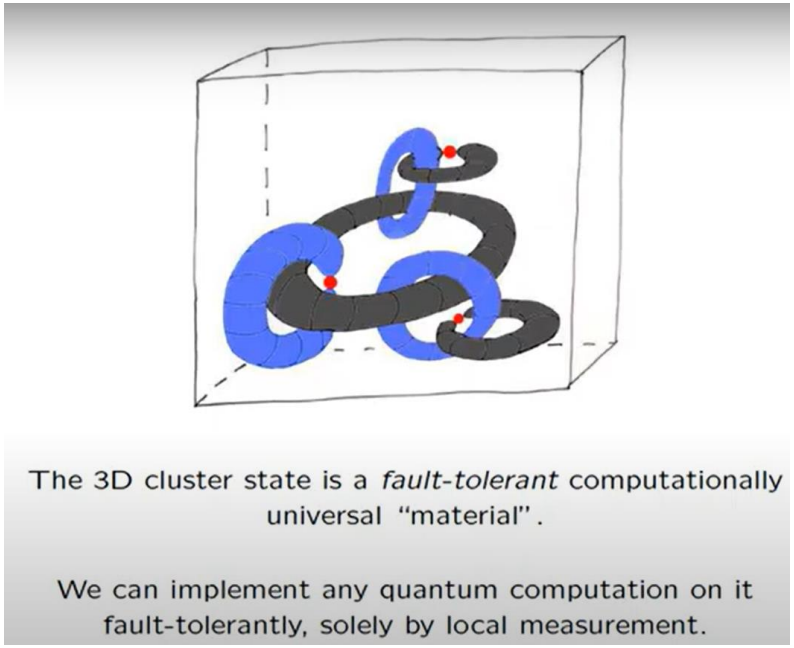
MBQC requires a lot of qubit, compared to circuit model.

# Cluster State



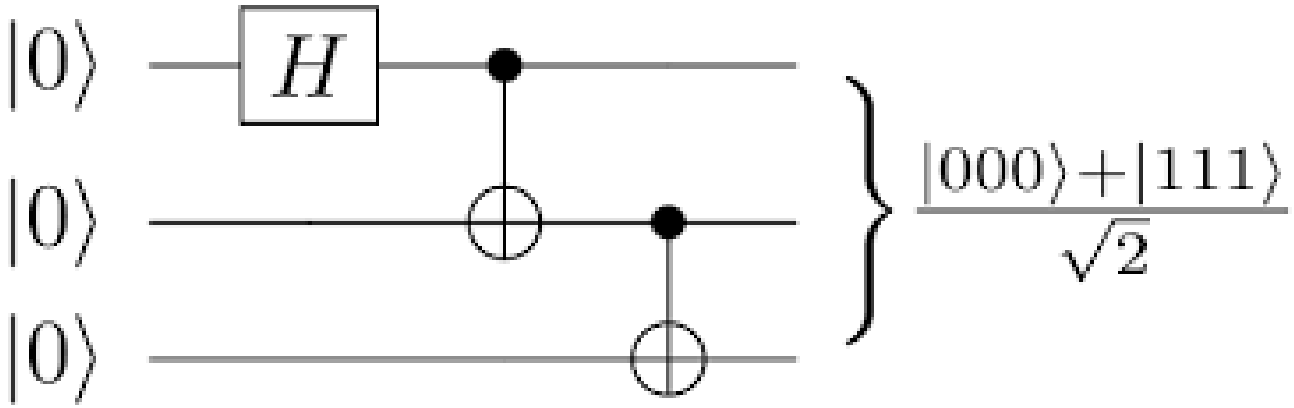

2D Cluster state -> we can implement any quantum computation on it, only by performing local measurement.

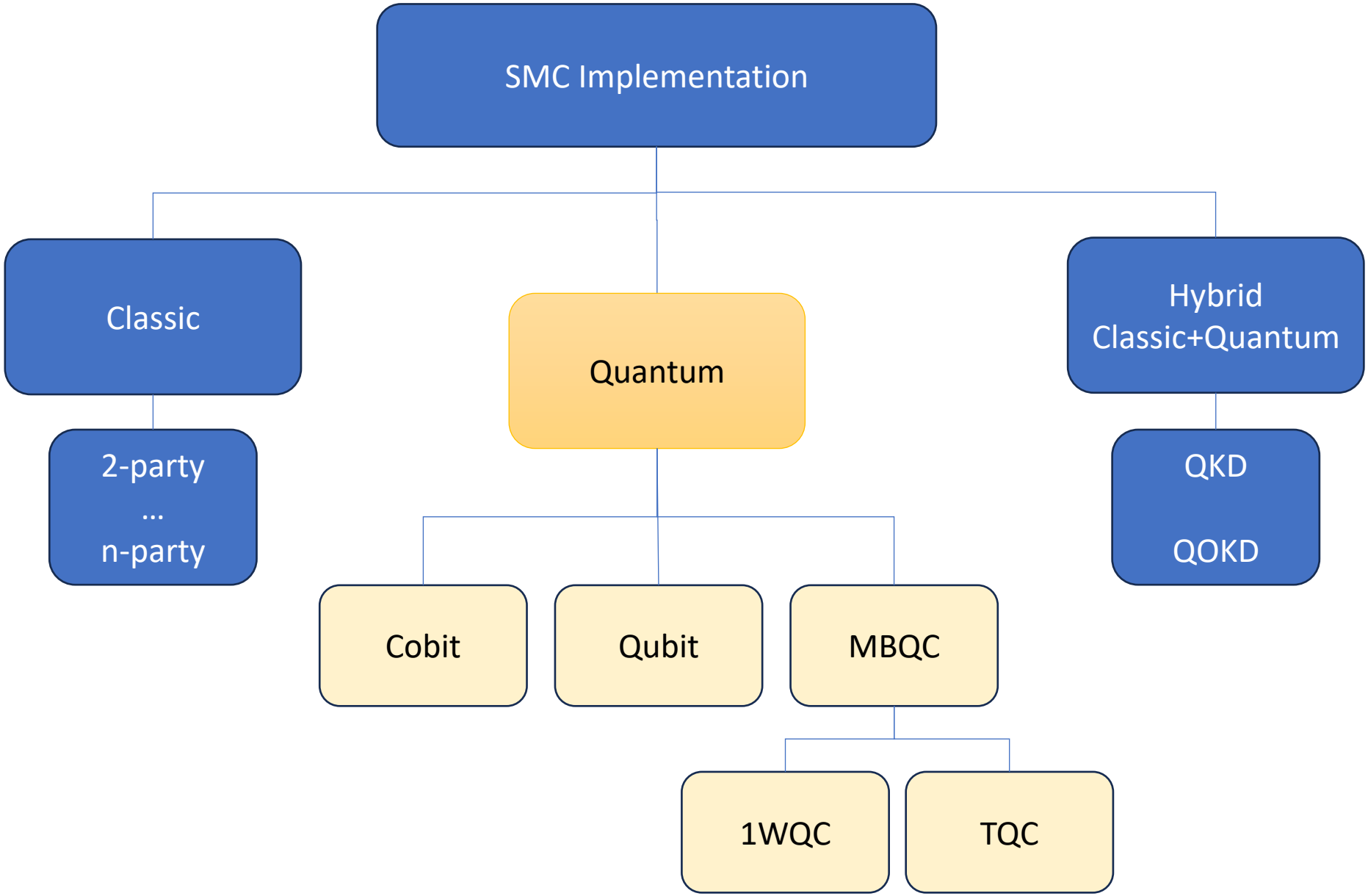
2D Cluster state -> is like a blank piece of paper -> By using measurement, we write on this paper -> we print a circuit on it by performing single-qubit measurement, as if we are drawing the circuit.




protection against noise= fault tolerance

# Greenberger-Horne-Zeilinger (GHZ) State




SMC Implementation
Classic
2-party
...
n-party
Quantum
Cobit
Qubit
MBQC
1WQC
TQC
Hybrid
Classic+Quantum
QKD
QOKD

# Paper: Secure Multi-Party Computation with a Dishonest Majority via Quantum Means

https://arxiv.org/pdf/0906.2297.pdf

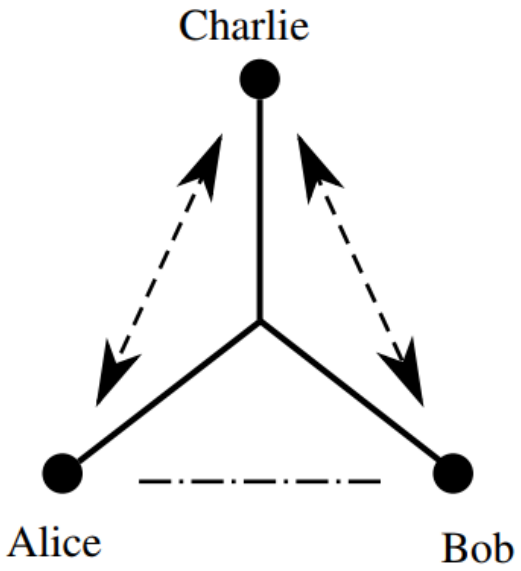
**Contribution:** Exploiting GHZ states to compute Boolean functions.

**Basic Idea:**

Inputs: a, b , a⊕b
Output -> AND(a,b)

Three qubits (GHZ) are distributed among three parties
Parties measure qubits in basis $\sigma x$ $or$ $\sigma z$

$$M_1 \oplus M_2 \oplus M_3 = \mathrm{AND(a,b)}$$



# Paper: Enhanced delegated computing using coherence

https://arxiv.org/abs/1501.06730

**Contribution:** Implementation of a NAND gate using coherence.

**Protocol**

1. Server sends cobits.
2. Client applies operations to them, dependent on some classical bits.
3. The cobits are sent back to the server.
4. Server performs a measurement on cobits.
5. The result of the measurement is the NAND operation on the client's classical bits.

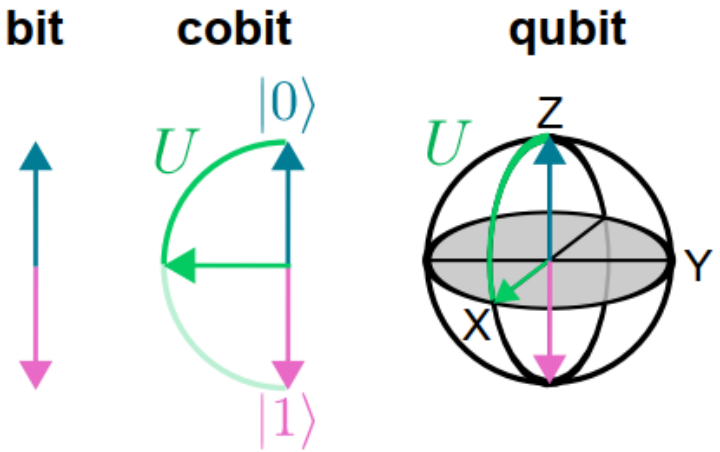
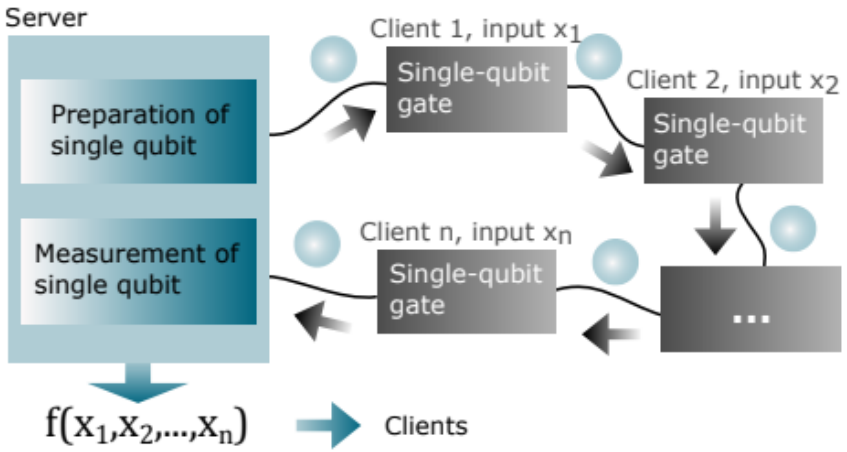



FIG. 1: Bit, cobits, and qubits. The bit is a two-level classical system, cobits are systems capable of being in a coherent superposition of two "states", and qubits are quantum systems. The operation $U$

# Paper: Classical SMC using quantum resources

https://arxiv.org/abs/1708.06144

**Contribution:** quantum can enable limited clients to securely compute non-linear functions.




## Protocol

1. Server generates a single qubit
2. Sends it to clients
3. Each client applies a rotation on the qubit (rotation is based on their input)
4. Qubit is sent back to server
5. Server performs measurement to obtain the result of the computation

- The inputs remain hidden to clients and server.
- Output remains hidden from the server.

# Paper: Secure Multiparty Computation via Measurement-Based Quantum Computing

**Contribution:** Computing Boolean function of degree 2.

**Protocol:**

1. A GHZ state is distributed among parties.
2. Parties perform rotations on their qubits considering their private input bits.
3. An additional rotation is performed to increase the security.
4. The qubits are measured, and the output of computation is obtained.

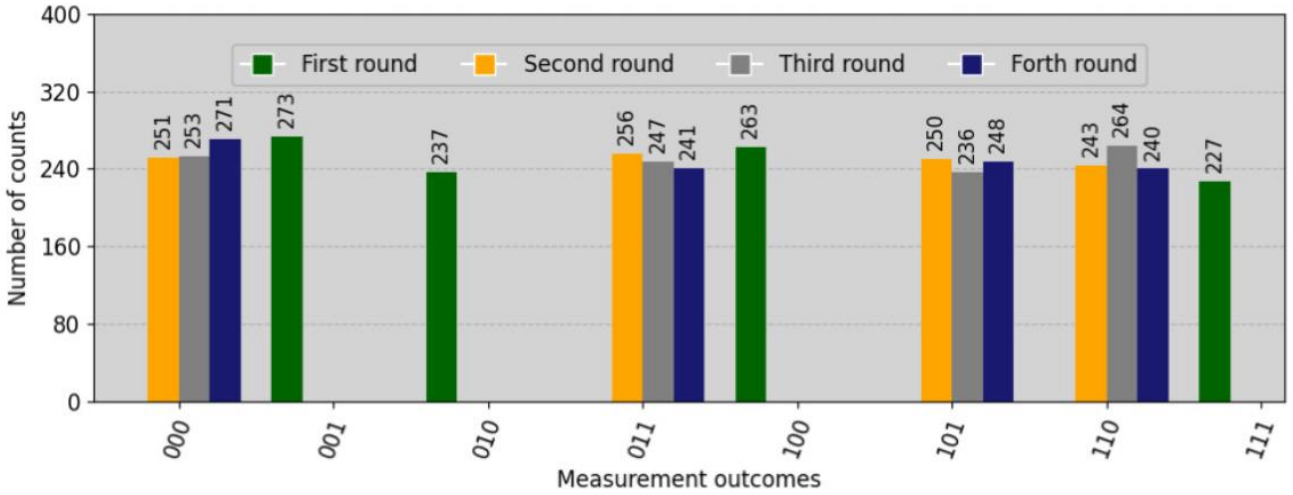
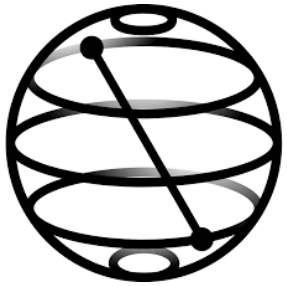



**Fig. 2** Measurement results of the quantum circuit demonstrated in Fig. 1, considering a particular scenario involving a 2-bit $OR(\vec{a}, \vec{b})$ function. The simulation are performed on *'qasm_simulator'* simulator. The circuit is run over 4 rounds with 1000 shots.

# What kind of Functions?

**Table 1** Comparison of different quantum SMC protocols. $\ell$ indicates the number of monomails. RoundCx and CommCx denote round complexity and communication complexity, respectively.

| QSMC protocols | Computed function | RoundCx | CommCx | Quantum resources |
|---|---|---|---|---|
| Ref. [31] | Boolean functions | 4 | $\mathcal{O}(2n)$ | GHZ state + Bell state (Prot. A) |
| Ref. [15] | Pairwise AND | 2 | $\mathcal{O}(2n^2)$ | Single qubit |
| Ref. [16] | N-tuple pairwise AND | 2 | $\mathcal{O}(2n^2)$ | Single qubit |
| Ref. [17] | N-variable polynomials | 3 | $\mathcal{O}(\ell n^2)$ | Single qudit (Prot. $\Gamma_1$)<br>Entangled state (Prot. $\Gamma_2$) |
| Red. [18] | Summation function | 1 | $\mathcal{O}(1)$ | Entangled state |
| This work | Boolean functions | 4 | $\mathcal{O}(2n)$ | GHZ state + Bell state |



# QisKit Implementation

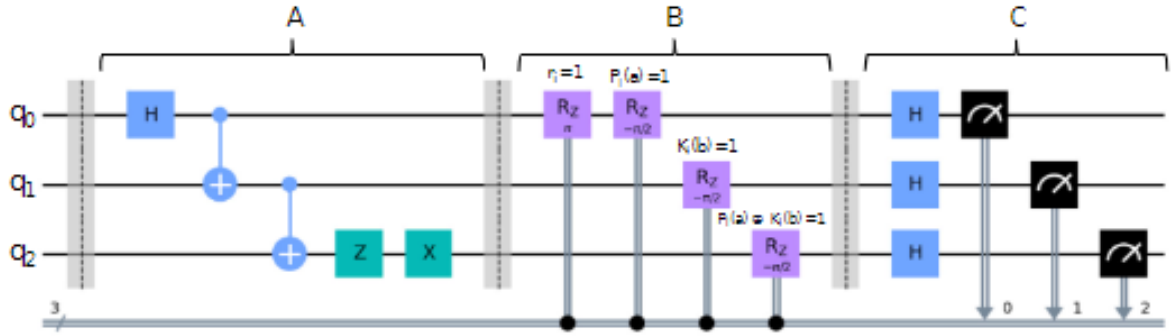



**Fig. 1** Quantum circuit for the proposed protocol. The $q_0, q_1$, and $q_2$ are three initial qubits with state $|0\rangle$. The label A represents the preparation of the GHZ state. Label B indicates the rotation of qubits with respect to the random bit $r$, $P_i$, $K_i$, $P_i \oplus K_i$. Note that the rotation gates in label B are exclusively applied when the bit values are equal to 1. Label C represents qubit measurements in the Hadamard basis. Labels 'H', '|', Z, 'X', and '$R_z$' identify the Hadamard, controlled-X, Pauli-Z, Pauli-X, and Z-rotation gates, respectively.

# SMC + Machine Learning

# 2: CrypTen: Secure Multi-Party Computation Meets Machine Learning

https://arxiv.org/abs/2109.00984

CRYPTEN: a classical software to implement SMC primitives for ML such as tensor computations, automatic differentiation, and modular neural networks.

CRYPTEN's API closely follows the API of the popular PyTorch framework for machine learning.

Security model: semi-honest

Feature Aggregation: parties hold distinct sets of features, and want to perform computations over the joint feature set without sharing data. For example, different health providers may each have part of a patient's medical history, but may wish to use the patient's entire medical history to make better predictions while still protecting patient privacy.

Data Labeling: Here, one party holds feature data while the another party holds corresponding labels, and the parties would like to learn a relationship without sharing data. For example, suppose that in previous healthcare scenario, one healthcare provider had access to health outcomes data, while other parties had access to health features. The parties may want to train a model that predicts health outcomes as a function of features, without exposing any health infomration between parties.

Dataset Augmentation: In this scenario, several parties each hold a small number of samples, but would like to use all the examples in order to improve the statistical power of a measurement or model. For example, when studying wage statistics across companies in a particular region, individual companies may not have enough data to make statistically significant hypotheses about the population. Wage data may be too sensitive to share openly, but privacy-preserving methods can be used to aggregate data across companies to make statistically significant measurements / models without exposing any individual company's data. (quantum based federated learning)

Model Hiding: In the final scenario, one party has access to a trained model, while another party would like to apply that model to its own data. However, the data and model need to be kept private. This can happen in cases where a model is proprietary, expensive to produce, and/or susceptible to white-box attacks, but has value to more than one party. Previously, this would have required the second party to send its data to the first to apply the model, but privacy-preserving techniques can be used when the data can't be exposed.

# SMC + Learning (Classical Approach)

Frameworks to implement SMC for ML:

- Crypten
- CryptFlow
- SecureNN
- SecureML
- MinioNN
- ABY3
- Delphi
- PySyft

| Framework | Malicious security | Triple generation | Supports GPUs | Supports training | General purpose† | Supports autograd |
|---|---|---|---|---|---|---|
| *Two parties* | | | | | | |
| Chameleon [62] | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Delphi [49] | ✗ | ✔ | ✗ | ✗ | ✗ | ✗ |
| EzPC [16] | ✗ | ✔ | ✗ | ✗ | ✗ | ✗ |
| Gazelle [40] | ✗ | ✔ | ✗ | ✗ | ✗ | ✗ |
| MiniONN [47] | ✗ | ✔ | ✗ | ✗ | ✗ | ✗ |
| PySyft [64] | ✗ | ✔ | ✔ | ✗ | ✗ | ✗ |
| SecureML [51] | ✗ | ✔ | ✗ | ✔ | ✗ | ✗ |
| XONN [63] | ✔ | N/A | ✗ | ✗ | ✗ | ✗ |
| *Three parties* | | | | | | |
| ABY3 [50] | ✗ | N/A | ✗ | ✔ | ✗ | ✗ |
| Astra [17] | ✗ | ✔ | ✗ | ✔ | ✗ | ✗ |
| Blaze [59] | ✗ | ✔ | ✗ | ✔ | ✗ | ✗ |
| CrypTFlow [43] | ✗ | N/A | ✗ | ✗ | ✔ | ✗ |
| CryptGPU‡ [67] | ✗ | ✗ | ✔ | ✔ | ✔ | ✔ |
| Falcon [72] | ✔ | N/A | ✗ | ✔ | ✔ | ✗ |
| SecureNN [71] | ✗ | N/A | ✗ | ✔ | ✗ | ✗ |
| *Four parties* | | | | | | |
| FLASH [11] | ✔ | N/A | ✗ | ✔ | ✗ | ✗ |
| Trident [60] | ✔ | N/A | ✗ | ✔ | ✗ | ✗ |
| *Arbitrary number of parties* | | | | | | |
| CrypTen (ours) | ✗ | ✗§ | ✔ | ✔ | ✔ | ✔ |

# SMC + Learning (Quantum Approach II)

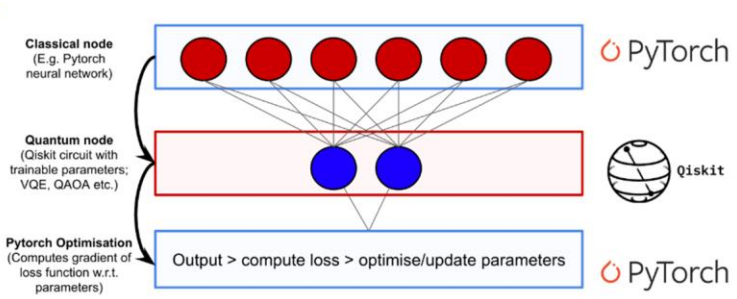
**Method:** SMC + Quantum Neural Network (QisKit)

1. Defining a SMC use case with sensitive data (patient's information in hospital)
2. Involving quantum technology in Crypten framework (Quantum Circuit of Qiskit)
3. Training the network and comparing the result with classical model.

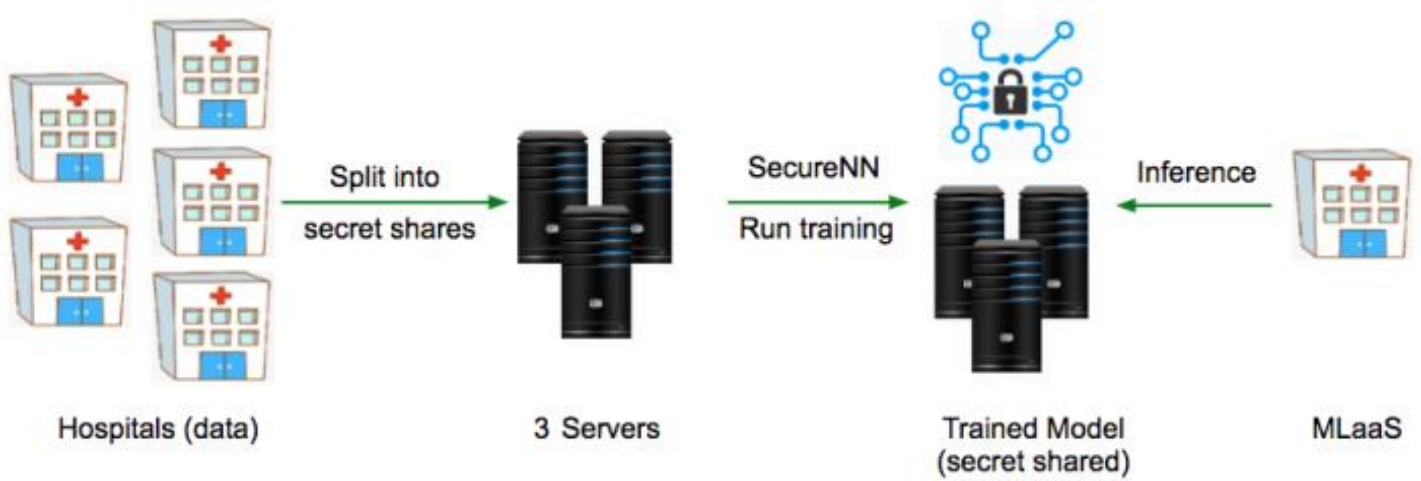
## How Does Quantum Enter the Picture?

To create a quantum-classical neural network, one can implement a hidden layer for our neural network using a parameterized quantum circuit. By "parameterized quantum circuit", we mean a quantum circuit where the rotation angles for each gate are specified by the components of a classical input vector.

https://qiskit.org/textbook/ch-machine-learning/machine-learning-qiskit-pytorch.html




Implementing SMC use case in Crypten → Integrating Qiskit to Crypten → Adding Quantum Circuit of Qiskit to Crypten

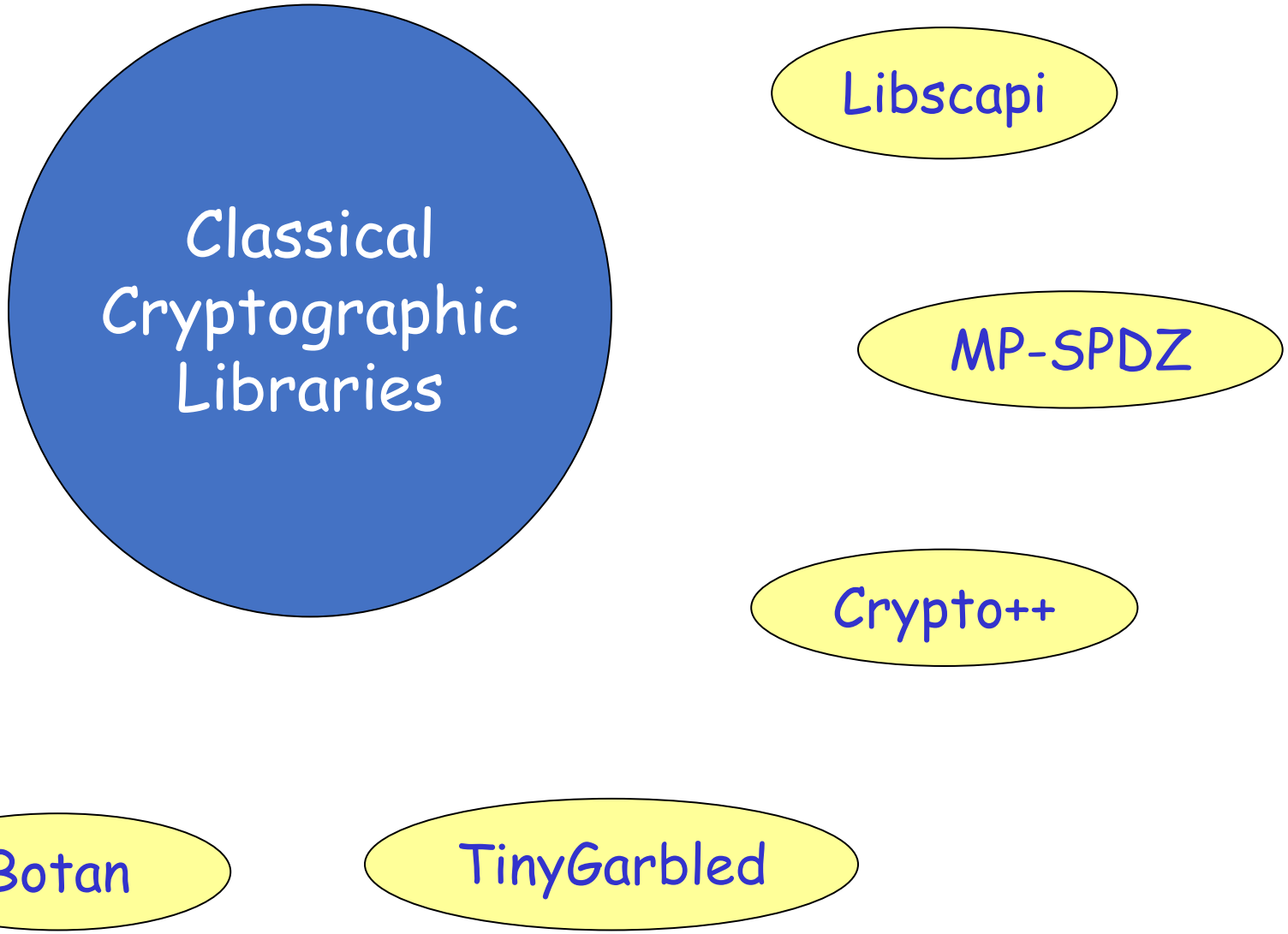



## Use case

https://petsymposium.org/popets/2019/popets-2019-0035.pdf

the servers. For example, a group of $M$ hospitals, each having sensitive patient data (such as heart rate readings, blood group, sugar levels etc.) can use the above architecture to train a model to run Machine Learning as a Service (MLaaS) and help predict some disease or irregular health behavior. The system can be set up such that the patient's sensitive input and predicted output are only revealed to the patient, and remains hidden from everyone else. The architecture is in Figure 1.

# Where to Start SMC?


Classical Cryptographic Libraries
Libscapi
MP-SPDZ
Crypto++
Botan
TinyGarbled

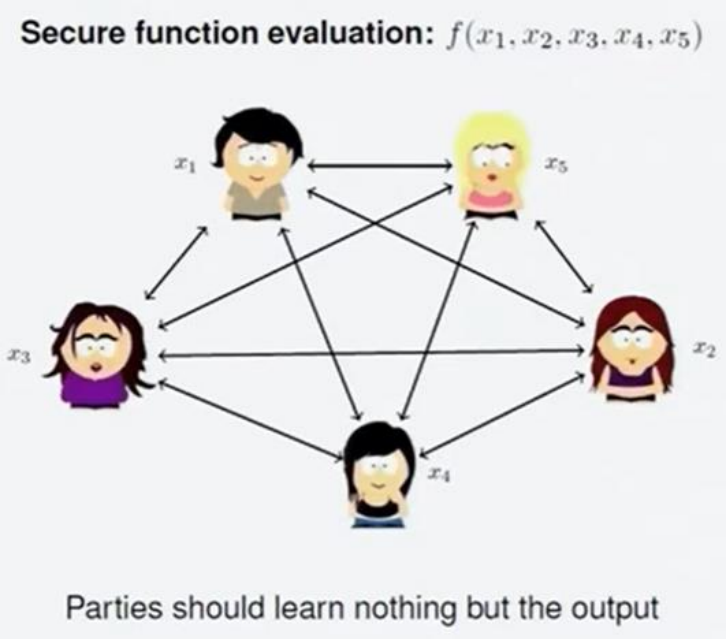
# MP-SPDZ Repository

SPDZ

BMR

BeDOZa

Overdrive




MASCOT

TinyOT

GMW

Yao Garbled Circuit

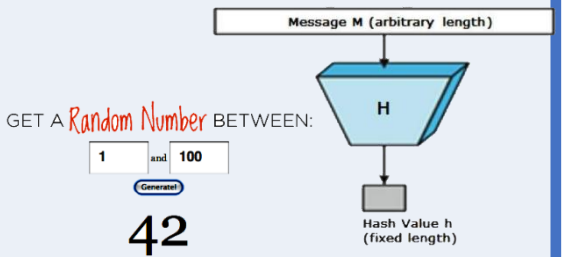
An implementatio of all these protocols are available on:
https://github.com/data61/MP-SPDZ

# LibSCAPI

- Open Source C++ Library

## Layer 1
### Basic Primitives

- ✓ Cryptographic Hash
- ✓ Pseudorandom Function
- ✓ Pseudorandom Generator
- ✓ Key Derivation Function
- ✓ Discrete Log Group

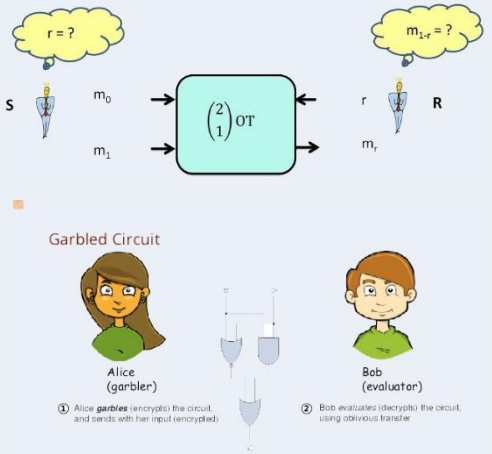



## Layer 2
### Non-Interactive Protocols

- ✓ Symmetric cryptography
- ✓ Asymmetric cryptography
- ✓ Digital signatures

## Layer 3
### Interactive Protocols

- ✓ Oblivious Transfer Protocols
- ✓ Commitment Schemes
- ✓ Circuits



# Use Case Definition

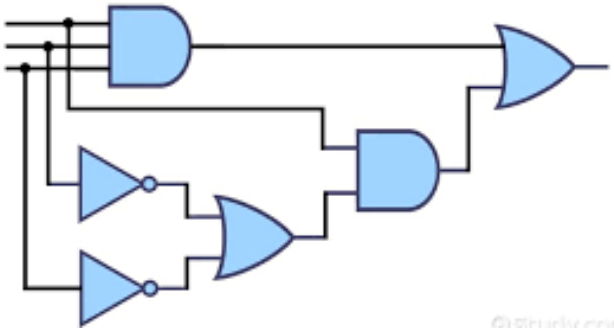
SMC for ML

SMC for Blockchain

SMC for Data Mining

...

# Circuit Definition & Implementation



- ✓ What is the function to be computed?
- ✓ What are the inputs of the function?
- ✓ What is the output of the function?

# Use Case Implementation

- ✓ QisKit
- ✓ Crypton
- ✓ SMC Libraries
- ✓ Other Platforms

# Thank You !